ÖZET

Yüksek Lisans Tezi

ESNEK AKIŞ TİPİ ÇİZELGELEME PROBLEMLERİNİN VERİ MADENCİLİĞİ VE GENETİK ALGORİTMA KULLANILARAK ÇÖZÜLMESİ

Gülşad CERAN

Selçuk Üniversitesi Fen Bilimleri Enstitüsü

Endüstri Mühendisliği Ana Bilim Dalı

Danışman: Yrd.Doç.Dr. Orhan ENGİN

2006, 134 Sayfa

Jüri:

Prof.Dr. Ahmet PEKER

Yrd.Doç.Dr. M.Atilla ARICIOĞLU

Yrd.Doç.Dr. Orhan ENGİN

Bu çalışmada, polinomiyal olmayan (Non-Polinomial-NP)-Zor sınıfı esnek akış tipi çizelgeleme problemlerinin çözümü ve çözüm aşamalarının analizi (büyük miktarda veri içinde gömülü olan anlamlı ve kullanışlı bilgilerin çıkarılması) amaçlanmıştır. Üretim sistemlerinde toplam akış zamanını (maksimum tamamlanma zamanı) minimize edecek çözümler (iş akış sıraları) elde etmek için Genetik Algoritmalar ve bu çözümler arasındaki ilişkileri ortaya çıkarmak için veri madenciliği kullanılmıştır. Bu bağlamda genetik algoritma tekniklerini kullanan bir program yazılmış, veri madenciliği için de verilerin hazırlanması ve analiz kısmı için bu işlemlere yönelik hazır veri madenciliği programları kullanılmıştır. Esnek akış tipi problemleri çözmek için geliştirilen genetik algoritma, kısa sürede etkili sonuçlar vermiştir. Doç.Dr. Ceyda Oğuz'un ve Carlier-Neron'un esnek akış tipi problemleri üzerine çalışılmıştır.

Anahtar Kelimeler: Veri Madenciliği, Esnek Akış Tipi Çizelgeleme, Genetik Algoritmalar

ABSTRACT
Master Thesis

SOLVING HYBRID FLOW SHOP SCHEDULING PROBLEMS BY USING DATA MINING AND GENETIC ALGORITHM

Gülşad CERAN
Selçuk University
Graduate School of Natural and Applied Sciences
Department of Industrial Engineering
Supervisor: Assist.Prof.Dr. Orhan ENGİN
2006, 134 Page

Jury:
Prof.Dr. Ahmet PEKER
Assist.Prof.Dr. M.Atilla ARICIOĞLU
Assist.Prof.Dr. Orhan ENGİN

In this study, it was aimed to solve problems of NP-Hard class hybrid flow shop scheduling problems and analyze the solutions (obtain meaningful and useful knowledge from hugeamount of data). In production systems, in order to obtain the minimized solutions of total flowing time, genetic algorithms were used, and in order to find relations between these solutions data mining was used. As a result, a program using genetic algorithm tecnics was designed, and for data mining to prepare and analyze data, present data mining programs interesting these solutions were used. For solving the hybrid flow shop scheduling problems, an efficient genetic algorithm is proposed. We used Ceyda Oğuz hybrid flow shop scheduling with multiprocessor task problems and Neron-Carlier hybrid flow shop scheduling problem from literature.

.

Keywords: Data Mining, Hybrid Flow Shop Scheduling, Genetic Algorithm

ÖNSÖZ

Bu çalışmanın ortaya çıkması sürecinde yardım ve desteğini hiçbir zaman esirgemeyen, danışman hocam sayın Yrd.Doç.Dr. Orhan ENGİN'e, yardımlarını esirgemeyen sayın Endüstri Mühendisi Esra AKAY'a ve sayın Endüstri Mühendisi M. Kerim YILMAZ'a, kıyaslama problemlerini ve çözümlerini gönderdiği için sayın Doç.Dr. Ceyda Oğuz'a ve hiçbir zaman desteğini esirgemeyen aileme sonsuz teşekkürlerimi sunarım.

Gülşad CERAN

Haziran, 2006

İÇİNDEKİLER

ŞEKİL LİSTESİ

TABLO LİSTESİ

EKLER LİSTESİ

KISALTMALAR

GA Genetik Algoritmalar

C_{max} Tamamlanma zamanı

YBS Yapay Bağışıklık Sistemleri

LB Lower Band (Alt Sınır) Literatür problemleri için bilinen en iyi çözüm

VM Veri Madenciliği

1. GİRİŞ

Günümüzün rekabete dayalı ortamında, işletmeler en az miktarda kaynak kullanarak, müşteri gereksinimlerine en hızlı yanıt verebilecek tekniklerle en kaliteli ürün ve hizmet üretmenin peşindedirler. Bu yüzden, hızlı değişen müşteri talepleri karşısında özellikle üretim planlarını en çabuk oluşturan işletmeler rekabette bir adım öne geçmektetir. Her geçen gün artan rekabet koşullarında firmalar ayakta kalabilmek için, etkili yöntemler seçmek için işletmeler optimizasyon tekniklerinden yararlanmak ve sahip oldukları bilgiyi doğru kullanmak zorundalar. Sürekli değişen rekabet ortamında hızlı karar vermek ve uygulamak gerekmektedir. Bu süreci hızlandıracak yöntemlerden biri veri madenciliğidir.

Veri madenciliği, işletmelerin yoğun rekabet koşullarında varolabilmeleri için geliştirilen çözümlerden biridir.Veri madenciliği; veri içerisinden alınan bilgiye ulaşma işidir. Madencilik teriminin kullanılma sebebi, büyük bir veri yığını arasından uygun olanı arama ve seçme işleminin maden arama işine benzetilmesidir (İnan, 2003).

Veri Madenciliği, karar verme mekanizmalarına yeni bilgiler üreterek işletmelerin etkin kararlar almasını sağlamaktadır. Bunun için doğru soruların doğru bir şekilde yanıtlanması gerekmektedir.

Veri madenciliğinin kullandığı tekniklerden biri genetik algoritmalardır. Genetik algoritmalar, çok degişkenli sistemlerde oldukça etkili ve gelişimini hala sürdüren bir optimizasyon tekniğidir.

Genetik algoritmalar, evrimsel hesaplama tekniğinin bir parçasını oluşturmakta ve geleneksel yöntemlerle, çözümü zor veya hemen hemen imkansız olan problemlerin

çözümünde kullanılmaktadır. Genetik algoritmalar, deneysel çalışmalarda optimizasyon aşamasında, endüstriyel uygulamalarda ve sınıflandırmalarda uygulama alanı bulunmaktadır. Mühendislik alanında en çok optimizasyon amaçlı olarak kullanılmakta ve diğer klasik yöntemlere göre daha iyi sonuç vermektedir (Bolat ve diğerleri, 2004).

Bu çalışmada; makinelerdeki iş sıralarını karakterize etmek için tüm operasyonları etkileyecek karar kuralları belirlemekle ilgilenilmiştir. İkinci bölümde, tez kapsamında ele alınan permutasyon akış tipi ve esnek akış tipi problemler incelenmiştir. Üçüncü bölümde genetik algoritmalar hakkında genel bilgiler, çalışma prensibi, kullandığı genetik operatörler hakkında bilgiler verilmiştir. Dördüncü bölümde, veri madenciliği hakkında genel bilgiler, kullandığı teknikler ve modeller hakkında bilgiler verilmiştir. Beşinci bölümde, literatür taraması yapılmıştır. Bu bölümde, esnek akış tipi problemler ve veri madenciliğinin çizelgeleme problemlerinde uygulanmasına yönelik literatür taraması bulunmaktadır. Altınca bölümde parametre optimizasyonu yapılmıştır. Yedinci bölümde, esnek akış tipi çizelgeleme problemlerinin genetik algoritmalar kullanarak veri madenciliğiyle çözümü gerçekleştirilmiştir. Sonuç bölümünde ise; çalışma ile ilgili eleştiriler yapılmış ve bilime olabilecek katkısında bahsedilmiştir.

2. ESNEK AKIŞ TİPİ ÇİZELGELEME PROBLEMLERİ

2.1. Çizelgeleme

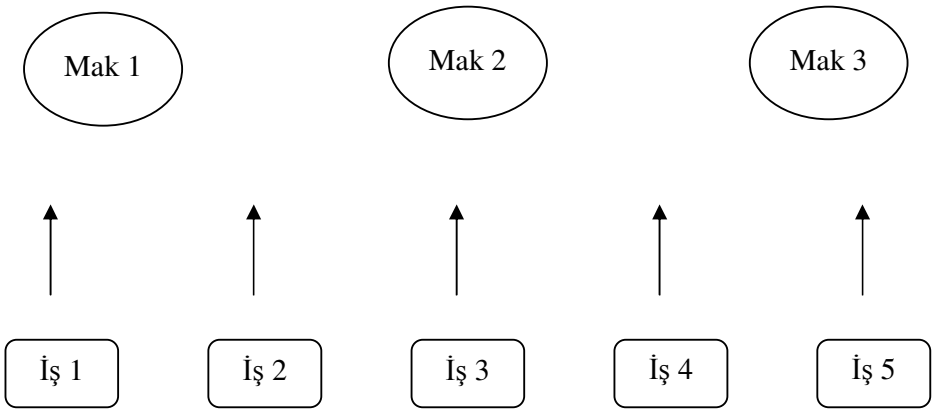
Çizelgeleme fonksiyonu, matematiksel teknikler veya sezgisel yöntemler kullanarak sınırlı kaynakların ilgili görevlere tahsis edilmesidir. Kaynakların uygun olarak atanması ile firmanın amaç ve hedeflerine en iyi şekilde ulaşması sağlanır. Çizelgeleme literatürü; parametrelerin belirgin olduğu durumdan, belirsiz olduğu duruma, tek makineliden çok makineliye, geliş sürecinin durağandan, dinamiğe değiştiği çeşitli problem yapılarını kapsar. Birden fazla ölçütün bulunduğu çizelgeleme çalışmaları son dönemlerde oldukça artmıştır. Ancak bu tür problemlerin çözümü tek ölçütlü problemler kadar kolay değildir. Çünkü birbirleri ile çelişen amaçların aynı anda en iyilendiğinden tek bir çizelgeyi oluşturmak oldukça zordur. Çizelgeleme problemleri tümleşik eniyileme problemleri yani çok amaçlı karar verme problemi olduğu için en iyi çözümlerini bulmak oldukça zordur.

Başka bir kaynakta ise; genel anlamda kesikli-parça imalatının olduğu ortamlarda var olan üretim çizelgeleme, zaman düzleminde yapılması gereken görevler için bir veya daha fazla amacı eniyileyecek şekilde kıt kaynakların tahsis edilmesi olarak tanımlanmıştır. Bu nedenle imalat endüstrilerinde çok önemli role sahip bir karar verme prosesidir. Kaynakların uygun olarak atanması ile firmanın amaç ve hedeflerini en iyi şekilde ulaşması sağlanır. Çizelgeleme probleminde makine kapasitesi kısıtları, teknolojik kısıtlar olmak üzere iki tür olurluluk kısıtı vardır. Çizelgeleme probleminin çözümü bu iki tip kısıtın birbirine bağlı ve uygun çözümüdür. Eğer elde edilen sonuç bize yerine getirilecek her bir görev için hangi kaynağın tahsis edileceğini ve her bir görevin ne zaman

yerine getirileceğini gösteriyorsa işimize yarayacaktır. Dolayısıyla, geleneksel olarak, çoğu çizelgeleme problemi kısıtlara bağlı optimizasyon problemi olarak görülmektedir (Cowling ve diğerleri, 2002).

2.1.1. Paralel Makine Problemleri

Paralel makine problemleri, m tane eş makinenin paralel olarak yerleştiği sistemlerdir. Her iş yalnız bir operasyona sahiptir ve bu m makinenin herhangi birinde işlenebilir. Genellikle makine sayısı, iş sayısından az olmaktadır. Örnek bir paralel makine sistemi Şekil 1'de gösterilmiştir (Döyen, 2004).

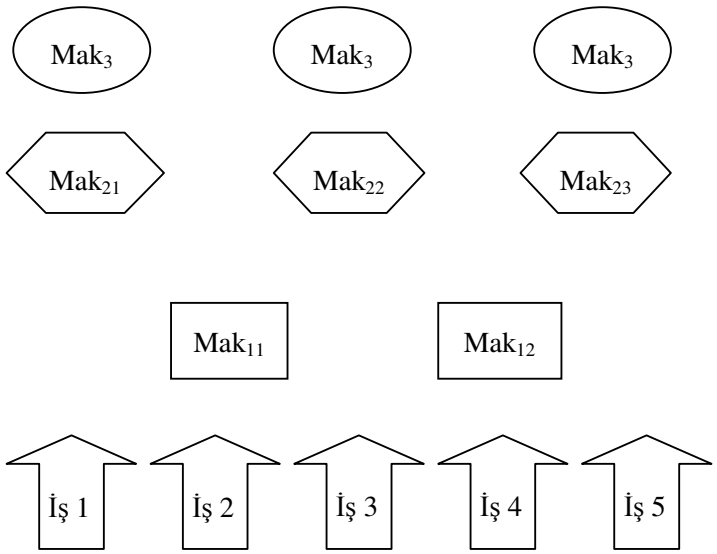


Şekil 1. Paralel makine sistem modeli.

2.1.2. Esnek Akış Tipi Problemler

Esnek (hibrid) akış tipi (hybrid flow shop) çizelgeleme problemi; akış tipi çizlgeleme problemleri ile paralel makine problemlerinin özellikleri birleştirmektedir (Şerifoğlu, 2004). Esnek akış tipi sistemde, makineler s tane seri kademeye

yerleştirilmişlerdir. $l = 1,2,...,s$ olmak üzere bir l kademesinde, m_1 tane eş makine bulunmaktadır. $j = 1,2,...,n$ olmak üzere bir j işi, her bir kademedeki makinelerden herhangi birinde yada daha fazlasında işlem görmelidir. Farklı kademelerde j işinin işlem süreleri $P_{1j}, P_{2j}..., P_{sj}$ ile gösterilir. İşlerin önceliği yoktur, örneğin, bir makinede bir operasyon başladıktan sonra başka bir operasyonun aynı makinede işlem görmeye başlayabilmesi için öncekinin mutlaka bitirilmesi gerekir. Her makinede belli bir anda en fazla bir iş işlem görebilir. Her kademede işlem görmeyi bekleyen işler için ayrılan stok alanı kısıtsız kabul edilir. Amaç, genellikle en son işin sistemden ayrılma zamanını ($C_{\max}$), minimize etmektir. Esnek akış tipi problemler NP-Zor'dur (Gupta, 1988). Şekil 2'de bir esnek akış tipi sistemin yapısı verilmiştir. Sistemde 5 iş, her birindeki makine sayıları sırasıyla 2, 3, 3 olan 3 kademede çizelgelenecektir (Döyen, 2004).



Şekil 2. Esnek akış tipi sistem modeli.

3. GENETİK ALGORİTMALAR

3.1. Genetik Algoritmaların Tarihçesi

Darwin'in evrim kuramında etkilenerek canlılarda yaşanan genetik süreci bilgisayar ortamında gerçekleştirmeyi düşünen John Holland, tek bir mekanik yapının öğrenme yeteneğini geliştirmek yerine böyle yapılarda oluşan bir topluluğun çoğalma, çiftleşme, mutasyon, vb. genetik süreçlerden geçerek başarılı (öğrenebilen) yeni bireyler oluşturabildiğini görmüş ve geliştirdiği yöntemin adı Genetik Algoritmalar olarak yerleşmiştir. 1985 yılında David E. Goldberg gaz boru hatlarının denetimi üzerine yaptığı doktora tezi ile genetik algoritmaların pratik kullanımının da olabilirliğini kanıtlamıştır (Goldberg, 1989). 1992 yılında John Koza genetik algoritmayı kullanarak genetik programlamayı geliştirmiştir (Koza 1992).

3.2. Genetik Algoritma Kavramı

Günümüzde çözülmesi zor ya da olanaksız olan çok sayıda problem vardır. Bu problemlerin çözülmesinde geliştirilmiş bir matematiksel fonksiyon olmadığı gibi olası çözümlerin hesaplanması ve en iyi çözümün seçilmesi de çok zaman almaktadır. Olası tüm çözümlerin değil de salt bazı seçilmiş çözümlerin denenmesi yöntemiyle, beklenen optimum sonucun elde edilmesini sağlayacak algoritmalar araştırılmış ve pek çok yöntem geliştirilmiştir. Geliştirilen yöntemlerden biri de Genetik Algoritmalardır (Baskak ve diğerleri, 2004).

GA'lar doğada geçerli olan en iyinin yaşaması kuralına dayanarak sürekli iyileşen çözümler üretir. Bunun için "iyi"nin ne olduğunu belirleyen bir uygunluk fonksiyonu ve yeni çözümler üretmek için yeniden kopyalama, değiştirme gibi operatörleri kullanır. GA'ların bir diğer önemli özelliği de bir grup çözümle uğraşmasıdır. Bu sâyede çok sayıda çözümün içinden iyileri seçilip kötüleri elenebilir.

3.3. Genetik Algoritmanın Çalışma Prensibi

Genetik Algoritmalar (GA) ya da daha geniş kapsamıyla Evrimsel Algoritmalar (EA), doğadaki evrimsel süreçleri model olarak kullanan bilgisayara dayalı problem çözme teknikleridir. Geleneksel programlama teknikleriyle çözülmesi güç olan, özellikle sınıflandırma ve çok boyutlu optimizasyon problemleri, bunların yardımıyla daha kolay ve hızlı olarak çözülebilmektedir. Algoritma, belirli tek bir problemi çözecek davranışın, varolan veya sonradan tanımlanan veri modeline dayandırılarak adım adım ortaya konulması ve bunun bilgisayar ortamında herhangi bir programlama diliyle kodlanmasıdır (Çölesen, 2002).

Bir problemin GA ile çözümünde izlenecek işlem adımları aşağıda verilmektedir.

1. Kullanıcının önceden tanımladığı kurallara göre genellikle rassal bir çözüm grubu seçilir veya kullanıcı kendisi ilk çözüm grubunu belirleyebilir. İlk çözüm grubuna başlangıç popülasyonu denir.

2. Her bir kromozom için bir uygunluk değeri hesaplanır; bulunan uygunluk değerleri dizilerin çözüm kalitesini gösterir. Popülasyonda yer alan en iyi uygunluk değerine sahip olan birey (kromozom), bir sonraki yeni nesile (popülasyon) doğrudan değiştirilmeden aktarılır.

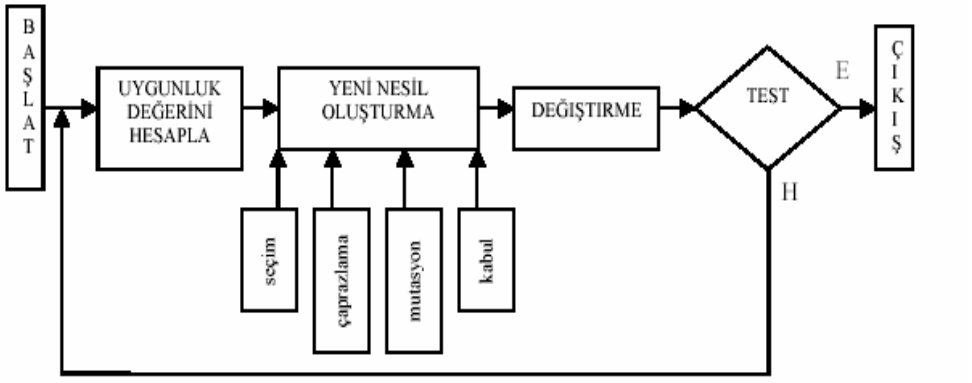
3. İki grup dizi (kromozom), belirli bir seçim yöntemine göre (uygunluk değerlerine göre hesaplanmış olasılık değerlerine göre) rassal olarak seçilirler.

4. Seçilen iki kromozom için rassal olarak genetik operatörler kullanılarak çaprazlama işlemi gerçekleştirilir. Sonuçta yeni popülasyonda yer alacak iki yeni birey (kromozom) oluşur. Çaprazlama, yeni popülasyonda yer alacak birey sayısına ulaşılana dek sürer.

5. Yeni popülasyonda ki bireyler, rassal olarak mutasyon işleminden geçerler.

6. Önceden belirlenen nesil sayısı boyunca yukarıdaki işlemler sürdürülür. Eğer en büyük nesil sayısına ulaşılmamışsa adım-2'ye dönülür. İterasyon, en büyük nesil sayısına ulaşınca işlem bitirilir. Uygunluk değeri en yüksek olan kromozom (çözüm) seçilir.

Şekil 3'de genetik algoritmaların genel akışı gösterilmiştir (Bolat ve diğerleri, 2004).



Şekil 3. Genetik algoritmaların genel akış şeması

GA bireylerin uygunluk ve iyiliklerine göre ayrılıp fark edilmesine gerek duyar. Bu sâyede üstün ve başarılı olanlar bir sonraki neslin bireylerini oluşturur. Çaprazlama, iki kromozomun bir araya gelerek genetik bilgi değişimi yapmasıdır. Mutasyon ise bir kromozomun taşıdığı genetik bilgide bir nedene bağlı olmaksızın değişme olmasıdır.

Mutasyon aramada kısır döngüye girilmemesini sağlamak, toplulukta çözümü olmayan birbirine benzer bireylerden kurtulmak ve yeni alt optimal çözümler bulunmasını sağlamak için kullanılır (Holland, 1975).

3.3.1. Başlangıç Popülasyonunun Oluşturulması

GA'ı diğer metotlardan ayıran önemli bir özellik ise, noktadan noktaya değil, noktaların oluşturduğu bir yığın içinde aramayı gerçekleştirmesidir (Goldberg, 1989). Bu nedenle, bir GA' nın ilk adımı başlangıç yığının oluşturulmasıdır. Literatürde, başlangıç yığını en basit şekilde rastsal olarak oluşturulmaktadır. Ancak, özellikle kısıtlı optimizasyon problemlerinde başlangıç yığınının rastsal oluşturulması sonucunda uygun olmayan çözümler ortaya çıkabilmektedir. Bu durumu ortadan kaldırmak için bir yol, problem için geliştirilmiş olan sezgisel metotlardan faydalanmaktır.

3.3.2. Değerlendirme Fonksiyonu

Her iterasyonda, yığındaki dizilerin bir değerlendirme fonksiyonu yardımıyla uygunluk değerleri hesaplanır. Uygunluk değeri, bir sonraki yığını oluşturacak yeni aday çözümlerin elde edilmesi için mevcut yığından hangi aday çözümlerin kullanılacağının belirlenmesinde rol oynamaktadır. GA da kullanılan değerlendirme fonksiyonu problemin amaç fonksiyonudur. Esnek akış tipi çizelgeleme problemlerinde amaç fonksiyonu minimum tamamlanma zamanıdır. Bu değer Gant yardımı ile hesaplanır.

3.3.3. Üretim Operatörü

Başlangıç yığını oluşturulduktan sonra algoritmanın her iterasyonunda, yeni yığının dizileri bir olasılıklı seçim süreci ile mevcut yığının dizileri arasından seçilir. Yüksek uygunluk değerine sahip diziler, yeni dizilerin (yeni çözümlerin) elde edilmesinde yüksek olasılığa sahiptir . Bu operatör, doğal seçimi yapay olarak gerçekleştirmektedir. Doğal yığınların uygunluğu, bireyin büyümesi ve çoğalmasında engellere karşı koyma yeteneği ile belirlenir. Amaç fonksiyonunun bir dizinin yaşaması ya da elenmesinde son karar verici olarak kullanımı ile doğal seçim yapay olarak gerçekleştirilir. Üretim operatörü teknikleri:

- **Rulet Çemberi** :

Seçim yöntemleri içinde en basit ve en kolay uygulanabilir nitelikteki yöntem, rulet-çemberi yöntemidir. Rulet-çemberi yöntemi stokastik bir yöntemdir. Tüm fertler bir çizgi üzerinde birbirine bitişik bölümler şeklinde dizilirler. Bununla birlikte her bir ferde ilişkin bölümün uzunluğu, onun uygunluk değeri kadar olur. Rasgele sayı üretilir ve rasgele sayı hangi bölüm içine denk gelirse, o bölümün ait olduğu fert seçilir. İşlem eşleşecek popülasyonun gerekli adedine ulaşılıncaya dek sürdürülür. Bu teknik, üzerinde bölümleri olan rulet-çemberine benzediği için bu şekilde adlandırılmıştır. Bu teknikte, ebeveynler uygunluk değerlerine göre seçilirler. Kromozomlar ne kadar iyiyse, seçilme şansları o kadar yüksektir. Bir kromozom birden fazla seçilebilir (Baskak ve diğerleri, 2004).

- **Turnuva Seçim Mekanizması**

Yığından rastsal olarak bir grup dizi seçilir. Bu grup içindeki en iyi uygunluk değerine sahip dizi yeni yığına kopyalanır. Bu işlem kullanıcı tarafından önceden

kararlaştırılan çevrim sayısı kadar tekrarlanır (Bolat ve diğerleri, 2004). Genellikle, grup genişliği ikidir. Ancak, bu sayının arttırılması da mümkündür (Goldberg, 1989).

3.3.4. Çaprazlama Operatörü

Çaprazlama oranı, fertlerin eşleştiklerinde çaprazlama yapıp yapmayacaklarına ilişkin olasılığı ifade eden orandır. Eğer eşleşme sonucunda çaprazlama da oluşursa, yeni ve genellikle ebeveynlerinden farklı bireyler elde edilmiş olur. Eğer çaprazlama gerçekleşmezse, bu takdirde ebeveynlerinin kopyası olan yeni fertler oluşur (Baskak ve diğerleri, 2004).

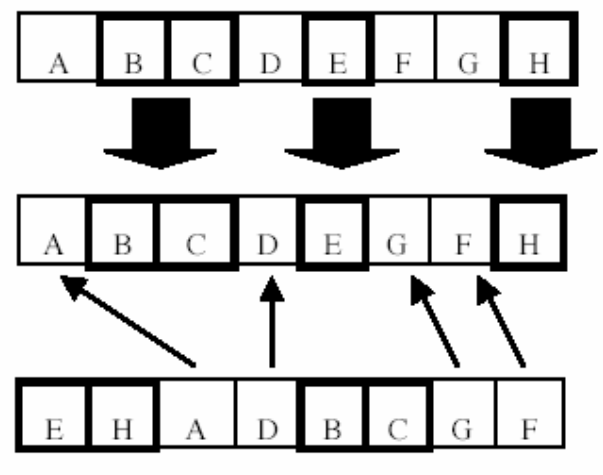
Çaprazlama sözcüğü, iki kromozomun (çözümün) birbirleri arasında gen alışverişinde bulunup iki yeni kromozom oluşturmasıdır. İkili yöntemle kodlanmış değişkenlerin yaptıkları üreme faaliyeti, kromozomların çaprazlamasına benzemesi dolayısıyla böyle adlandırılmaktadır. Ancak, gerçek değerlerin kullanılmak zorunda oldukları spesifik problemlerde, klâsik çaprazlama yöntemi yerine daha farklı yöntemler kullanılmaktadır. Seçim yöntemi ile yapay seçim sonucunda elde edilen yeni popülasyon dizisinden rassal olarak iki kromozom seçilir ve karşılıklı çaprazlama işlemine tâbi tutulur (Holland, 1975).

Akış tipi çizelgeleme problemlerine uygun altı farklı çaprazlama yöntemi test edilmiştir. Bunlar; Pozisyona dayalı, Sıraya dayalı, Kısmi planlı, Dairesel, Doğrusal ve Sıralı Çaprazlama yöntemleridir.

- Pozisyona Dayalı Çaprazlama Yöntemi (PBX)

Bu yöntemde rastlantısal olarak seçilmiş pozisyondaki işler, bir ebeveynden çocuğa kalıtsallaştırılır. Diğer işler diğer ebeveynde bulundukları sıra ile yerleştirilir. Öncelikle pozisyondaki sayılar, [1, n] rastlantısal tamsayılar şeklinde düzenlenir, daha sonra bu

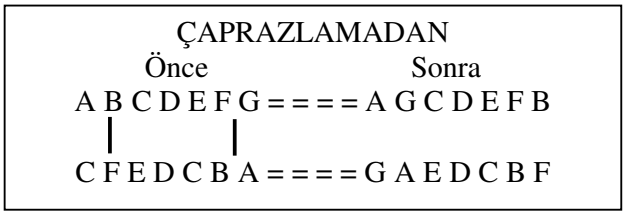
pozisyonlar rastlantısal olarak seçilir. Her pozisyonun çaprazlama olasılığı %50 dir (Murata ve diğerleri, 1996). Şekil 4'de sekiz iş içeren iki kromozomdan yapılan pozisyona dayalı çaprazlama örneği görülmektedir.



Şekil 4. Pozisyona dayalı çaprazlama**.**

- Sıraya Dayalı Çaprazlama Yöntemi (OBX)

Bu yöntemde bir grup nokta rasgele seçilir. Birinci kromozomun seçilen noktalara karşılık gelen karakterleri aynen yerlerini korur. İkinci kromozomun seçilen noktalara ait karakterleri birinci kromozomun aynı noktalarındaki karakterlerin arkasına getirilir. Geriye kalan boş pozisyonlara ikinci kromozomdan aktarılan yeni karakterler de göz önünde bulundurularak ilk kromozomun kullanılmayan karakterleri sıra ile (soldan sağa) yerleştirilerek yeni bir kromozom elde edilir (Cheng ve diğerleri, 1999). Bu tür çaprazlama, kromozomu oluşturan karakterlerin sayı ve sıralarının önem taşıdığı durumlarda kullanılır. Bu çaprazlama işlemine ait birer çaprazlama örneği Şekil 5'de verilmiştir.



Şekil 5. Sıraya dayalı çaprazlama.

- Kısmi Planlı Çaprazlama (PMX)

Goldberg tarafından geliştirilen bu çaprazlama ilk olarak gezgin satıcı probleminde (TSP) kullanılmıştır. Bu yöntemde iki ayrı iş sırasında rastlantısal olarak aralıklar belirlenir ve bu aralıkta yer alan işlerin yeri karşılıklı olarak değiştirilir (Goldberg, 1989). Bu yöntem aşağıda bir örnek üzerinde açıklanmaktadır:

- Dairesel Çaprazlama (CX)

Davis, Goldberg ve Lingle tarafından geliştirilmiş bir yöntemdir. Bu yöntemde ilk kromozomdan en baştaki gen seçilir ve bu gen yeni diziye yerleştirilir. Bu gene karşılık gelen ikinci kromozomdaki gen belirlenir bu değer de yeni kromozom üzerine yerleştirilerek dairesel bir şekilde bütün genler belirlenir (Goldberg, 1989).

- Doğrusal Sıralı Çaprazlama (LOX)

Falkenauer ve Bouffouix tarafından geliştirilmiştir. Dairesel çaprazlamanın bir varyantıdır. İşlem adımları (Cheng ve diğerleri, 1999) aşağıda verilmektedir:

1. Mevcut popülasyon içerisinden rastsal olarak iki ebeveyn seç,

2. Seçilen bu iki dizi (kromozom) üzerinde rastsal olarak iki alt dizi seç,

3. P1 dizisinden seçilen alt diziyi kromozomdan kopar ve boş kalan yerleri belirle, benzer şekilde P2 dizisinde de aynı işlemleri gerçekleştir,

4. Birinci alt diziyi P1.e ve ikinci alt diziyi P2.ye yerleştir.

- Sıralı Çaprazlama (OX)

Bu yöntem de, Davis,Goldberg ve Lingle tarafından geliştirilmiştir (Goldberg, 1989). Bu yöntemde, kromozom havuzundan rastsal olarak iki kromozom seçilir. Bu kromozomlar üzerinde yine rastlantısal olarak iki ayrı kesim noktası belirlenir. Bu kesim noktaları arasındaki kromozom sayısının her iki kromozomda da aynı olmasına dikkat edilir. Kesim noktaları arasındaki kromozomlar karşılıklı olarak yer değiştirilir. Kesim bölgesi dışında yer alan genler içerisinde tekrarlı genler oluşursa bunlar yerine sıra ile soldan sağa doğru kromozomda bulunmayan genler yazılır.

3.3.5. Mutasyon Operatörü

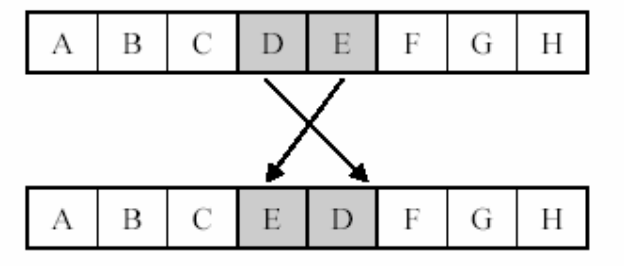
Mutasyon, bireyin kromozomunu oluşturan dizideki tek bir elemanın değerinin rasgele olarak değiştirilmesidir. Mutasyon, çözümün alt optimal noktalara takılmasını önleyen ve çok düşük olasılık değeri ile uygulanan operatördür (Akbaş ve Oğuz, 2001) .

- Ters Mutasyon

Bir kromozomda rassal olarak iki pozisyon seçilir, bu iki pozisyondaki alt diziler ters çevrilir (Murata ve diğ., 1996b).

- Komşu İki Geni Değiştirme

Rassal olarak seçilen iki komşu gen değiştirilebilir (Murata ve diğ., 1996b). Şekil 6' da komşu iki gen değiştirme mutasyon işlemi gösterilmiştir (Engin, 2001).

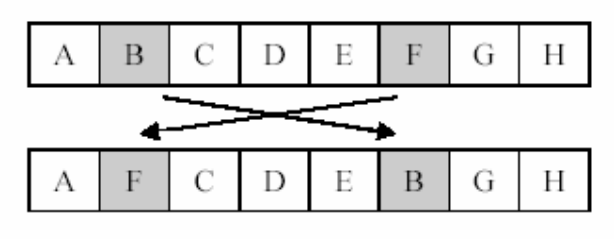


Şekil 6. Komşu iki geni değiştirme.

- Keyfi İki Geni Değiştirme

Rastsal seçilen iki gen değiştirilebilir. Özel bir durum olarak, değiştirilebilen iki komşu gen bu mutasyon içerir (Murata ve diğ., 1996b).

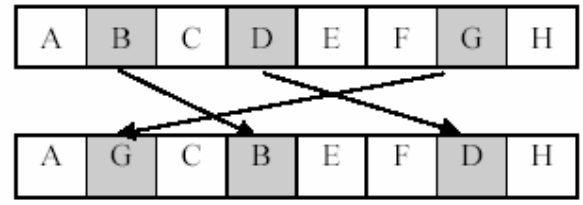
Şekil 7'de keyfi iki geni değiştirme mutasyon işlemi gösterilmiştir (Engin, 2001):



Şekil 7. Keyfi iki geni değiştirme.

- Keyfi Üç Geni Değiştirme

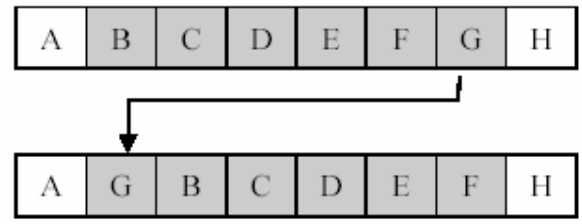
Rassal olarak seçilen üç gen keyfi olarak değiştirilir (Murata ve diğ., 1996b). Şekil 8'de keyfi üç işi değiştirme mutasyon işlemi gösterilmiştir (Engin, 2001):



Şekil 8. Keyfi Üç Geni Değiştirme

- Araya Yerleştirme

Bu mutasyon işleminde, bir operasyondaki bir gen kaydırılır ve diğer bir pozisyona konulur. Kaydırma noktası rastsal olarak seçilir. Bu mutasyon, komşu iki gen değiştirmenin özel bir durumudur. Keyfi üç geni değiştirmeyle bir kesişime de sahiptir (Murata ve diğ., 1996b). Şekil 9'da araya yerleştirme işlemi gösterilmiştir (Engin, 2001):



Şekil 9. Araya yerleştirme.Çaprazlama Oranı

Çaprazlamanın amacı, mevcut iyi kromozomların özelliklerini birleştirerek daha uygun kromozomlar oluşturmaktır. Çaprazlamanın artması, yapı bloklarının artmasına neden olmakta fakat aynı zamanda bazı iyi kromozomlarında bozulma olasılığını da artırmaktadır (Emel ve Taşkın, 2002) .

3.3.6. Mutasyon Oranı

Mutasyonun amacı popülasyondaki genetik çeşitliliği korumaktır. Mutasyon belirli bir olasılıkla ($P(m)$) bir kromozomdaki gende meydana gelebilir. Eğer mutasyon olasılığı artarsa, genetik arama rastsal bir aramaya dönüşür (Yeniay, 2001).

Mutasyon oranı, tıpkı çaprazlama oranında olduğu gibi mutasyonun olasılığını ifade eden bir orandır. Yine rasgele yöntemlerle kromozomun mutasyona uğrayıp uğramayacağı belirlenir ve buna göre mutasyon gerçekleştirilir. Mutasyon oranı genellikle çok düşük (0,01) olduğundan mutasyon işlemi fertlerde az görülür. GA'da önemli rol oynayan süreçlerden biri de mutasyon operatörüdür. Yapay sistemlerde mutasyon işlemi, kromozomlarda anî olarak oluşan değişimlerdir. Mutasyon sırasında kromozomdaki gen sayısı değişmez, sabit kalır. Mutasyon işlemi bir tek kromozom üzerinde yapılır. Mutasyon oranına göre, mutasyona uğratılacak sayıdaki diziler popülasyondan rassal olarak seçilir ve belirlenen mutasyon yöntemine göre değişime uğratılır (Baskak ve diğerleri, 2004).

GA' nın kombinatoriyel optimizasyon problemlerinde etkin bir şekilde kullanılabilmesi için GA'da kullanılan tüm parametrelerin optimize edilmesi gereği açıktır. Bu yolla çözüm kalitesi ve performansı önemli ölçüde iyileştirilebilir (Engin ve Fığlalı, 2002).

4. VERİ MADENCİLİĞİ

1960'larda veri toplama ve veri tabanlarının oluşturulması; 1970'lerde ilişkisel veri modeli, ilişkisel veritabanı yönetim sistemlerinin oluşturulması; 1980'ler ileri veri modelleri ve uygulama kaynaklı veri tabanı yönetim sistemlerinin geliştirilmesi; 1999-2000'lerde çoklu ortam veritabanları ve web veritabanları, veri ambarlama ve veri madenciliğinin oluşması sürecini göstermektedir.

Veri madenciliği (veri tabanlarında bilgi bulma); sıra dışı ama önemli, önceden bilinmeyen ve yarar sağlayacak bilgilerin bulunmasıdır.

Veri madenciliği adı geniş veri tabanlarında değerli iş bilgisini araştırma ve bir dağın değerli bir maden için kazılması arasındaki benzerlikten türemiştir (Ciocoiu, 2001).

Konunun önde gelen uzmanlarından Piatetsky-Shapiro veri madenciliğini, verilerden daha önceden bilinmeyen, zımni, muhtemelen faydalı enformasyonun monoton olmayan bir süreçte çıkartılması işlemi olarak tanımlamaktadır (Akpınar, 2000)

Gartner Group tarafından yapılan bir diğer tanımda ise veri madenciliği, istatistik ve matematik tekniklerle birlikte örüntü tanıma teknolojilerini kullanarak, depolama ortamlarında saklanmış bulunan veri yığınlarının elenmesi ile anlamlı yeni korelasyon, örüntü ve eğilimlerin keşfedilmesi sürecidir (Çerkez, 2003).

Veri madenciliği, eldeki yapısız veriden, anlamlı ve kullanışlı bilgiyi çıkarmaya yarayacak tümevarım işlemlerini formüle analiz etmeye ve uygulamaya yönelik çalışmaların bütününü içerir (Vahaplar, 2004).

Frawley'e göre veri madenciliği; eldeki verilerden üstü kapalı, çok net olmayan, önceden bilinmeyen ancak potansiyel olarak kullanışlı bilginin çıkarılmasıdır.

Veri madenciliği, büyük miktarda veri içinden gelecekle ilgili tahmin yapmamızı sağlayacak bağıntı ve kuralların bilgisayar programları kullanarak aranmasıdır (Alpaydın, 2000).

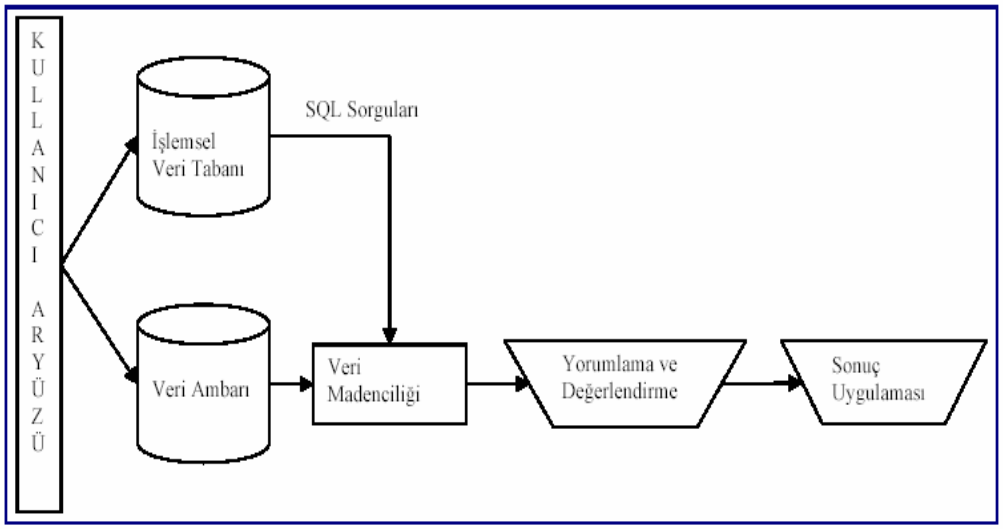
Veri madenciliği; anlamlı modeller ve kurallar ortaya çıkarmak amacıyla, otomatik ve yarı-otomatik yöntemler kullanarak büyük miktarlardaki veriyi analiz etme ve inceleme süreci olarak tanımlanabilir (Tuğ, 2004).

Geniş veri kümelerinden desenleri, değişiklikleri, düzensizlikleri ve ilişkileri çıkarmakta kullanılır. Bu sayede, web üzerinde filtrelemeler, DNA sıraları içerisinde genlerin tespiti, ekonomideki eğilim ve düzensizliklerin tespiti, elektronik alışveriş yapan müşterilerin alışkanlıkları gibi karar verme mekanizmaları için önemli bulgular elde edilebilir (Vahaplar, 2004).

Veri madenciliği, büyük hacimlerdeki verinin filtrelenmesi sürecini takiben, faydalı bilgiye ulaşarak şirketlerdeki karar vermeyi iyileştirmeye olanak sağlamaktadır (Karahoca, 2004).

Temel olarak VM, veri setleri arasındaki desenlerin yada düzenin, verilerin analizi ve yazılım tekniklerinin kullanılması ile ilgilidir. Veriler arasındaki ilişkiyi, kuralları ve özellikleri belirlemekten bilgisayar sorumludur. Amaç, daha önceden fark edilmemiş veri desenlerinin tespit edebilmektir (Vahaplar, 2004).

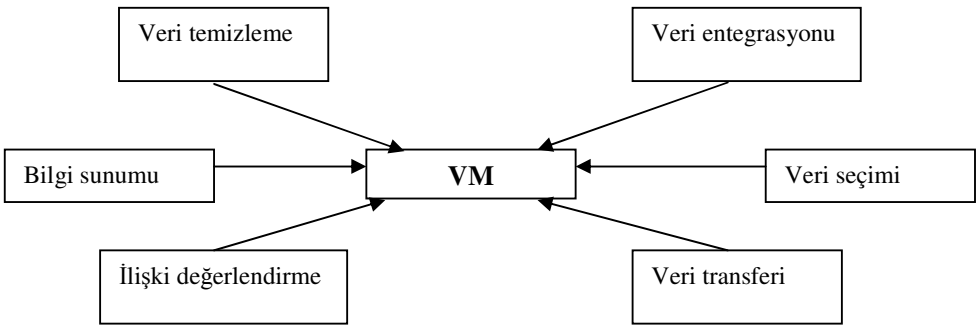
Şekil 10'da veri madenciliği ile bilgi çıkarım işleminin işleyişi görülmektedir (Tuğ, 2004).



Şekil 10. Basit bir veri madenciliği işleyiş modeli

Büyük miktarda veri içinde gömülü olan bilgilerin çıkarılması ve bu bilgilerin kuruluşa stratejik karar desteği sağlama amaçlı kullanılmasıdır.

Şekil 11'de veri madenciliği disiplini gösterilmiştir (Gaber, 1999).



Şekil 11. Veri madenciliği disiplini

4.1 Veri Madenciliği Modelleri

Veri madenciliğinde kullanılan modeller, tahmin edici ve tanımlayıcı olmak üzere iki ana başlık altında incelenmektedir (Çerkez, 2003).

- Tahmin Edici Modeller :

  Sonuçları bilinen verilerden hareket edilerek bir model geliştirilmesi ve kurulan bu modelden yararlanılarak, sonuçları bilinmeyen veri kümeleri için sonuç değerlerin tahmin edilmesi amaçlanmaktadır.

- Tanımlayıcı Modeller:

  Karar vermeye rehberlik etmede kullanılabilecek mevcut verilerdeki örüntülerin tanımlanması sağlanmaktadır.

Veri madenciliği işleminde amaç, verinin iç yapısındaki ilişkileri ortaya çıkarmak, kümelenmeleri ve bu kümelerin yapılarını bulmak, varolan verilerden yola çıkarak çeşitli öngörülerde bulunmak, kısaca verinin iç yapısını çözmektir (Tantuğ, 2002).

Veri madenciliği modelleri, gördükleri işlevlere göre aşağıdaki şekilde sıralanabilir (Çerkez, 2003).

- Sınıflandırma,
- Tahmin,
- Öngörme,
- Birliktelik Kuralları ve Ardışık Zamanlı Örüntüler,
- Kümeleme,
- Tanımlamadır.

4.1.1 Sınıflandırma

Sınıflandırma, veri madenciliği işlevleri arasında en yaygın olanıdır. Sınıflandırma, yeni bir nesnenin özellikleri araştırılarak, önceden tanımlanmış sınıflar kümesindeki uygun olanına dahil edilmesidir. Veri madenciliğinde işleme tabi nesneler, veri tabanındaki kayıtlar ile temsil edilmekte ve sınıflandırma işlemi neticesine her bir kayda belirli bir sınıf kodu atanmaktadır (Çerkez, 2003).

4.1.2 Tahmin

Sınıflandırma işlevinde bağımlı değişkenler kategorik bir değere sahip iken, tahmin işlevinde söz konusu değerler süreklilik göstermektedir. Girdi olarak kullanılan birkaç veriden tahmin işlemi sonucunda bilinmeyen ancak süreklilik arz eden değişkenler için değer üretilir (Çerkez, 2003).

Tahmin işlemi için kullanılabilecek tek yöntem yapay sinir ağları yöntemidir (Tantuğ, 2002).

4.1.3 Öngörme

Öngörme işlevini sınıflandırma ve tahmin işlevinden ayıran en önemli özelliği, kayıtların, mevcuttan öte, ileride öngörülen davranış ve değerler dışında sınıflandırılmasıdır (Çerkez, 2003). Öngörme işleminde yapılan sınıflandırmanın doğru olup olmadığını test etmenin tek yöntemi, bekle ve gör prensibidir (Tantuğ, 2002).

4.1.4 Birliktelik Kuralları ve Ardışık Zamanlı Örüntüler

Birliktelik kuralları ve ardışık zamanlı örüntüler, pazarlama amaçlı olarak pazar sepeti analizi adı altında veri madenciliğinde yaygın olarak kullanılmaktadır. Bununla birlikte bu teknikler; tıp, finans ve farklı olayların birbirleriyle ilişkili olduğunun belirlenmesi sonucunda değerli bilgi kazanımının söz konusu olduğu ortamlarda da önem taşımaktadır (Çerkez, 2003).

4.1.5 Kümeleme

Kümeleme modellerinde amaç, küme üyelerinin birbirine çok benzediği, ancak özellikleri birbirinden farklı olan kümelerin bulunması ve veri tabanındaki kayıtların bu farklı kümelere bölünmesidir. Kümeleme işlemini sınıflandırma işleminden ayıran temel özelliği, önceden tanımlanmış sınıflarla ilgili olmasıdır (Çerkez, 2003).

4.1.6 Tanımlama

Karmaşık bir veritabanında olanları tanımlayarak, verileri oluşturan müşteri, ürün ve prosesleri daha iyi anlamamıza yardımcı olur. Bir davranış ne kadar iyi tanımlanırsa, o kadar doğru açıklanabilir. İyi bir tanımlama en azından nereden başlanabileceği konusunda fikir verir (Çerkez, 2003).

4.2. Veri Madenciliği Teknikleri

Veri madenciliği teknikleri, verinin hazırlanmasından sonra, model oluşturma yada örüntü bulma için veritabanlarında kullanılan algoritmalardır. Bunlar;

- Sepet analizi,
- Bellek Tabanlı Yöntemler,
- Kümeleme (Gözetimsiz Sınıflandırma),
- İlişkisel Analiz,
- Karar Ağaçları ve Kural Türetme,
- Yapay Sinir Ağları,
- Genetik Algoritmalardır (GA).

4.2.1. Sepet Analizi

Sepet analizi tekniği, hangi ürünlerin hangi ürünle satıldığını, hangi ürünlerin promosyona girmesi gerektiğini ve benzeri bilgileri ortaya çıkarır. Sepet analizi çoğunlukla ticari anlam taşıyan verilerin olduğun ancak bu veri üzerinde hangi örüntülerin aranılacağının bilinmediği durumlarda bir başlangıç noktası olarak kullanılır (Tantuğ, 2002).

Ancak sepet analizi, geleceğe dönük tahminlerde pek başarılı değildir. Sepet analizinin temeldeki mantığında yer alan teknikler istatistik ve olasılıktan gelmektedir (Tantuğ, 2002).

4.2.2. Bellek Tabanlı Yöntemler

Gözetimli bir veri madenciliği tekniği olan bellek tabanlı yöntemler deneyimleri kullanmaktadır. Bu yöntemlerde, bilinen kayıtların bulunduğu bir veritabanı oluşturulur ve sistem yeni gelen bir kayda komşu olan diğer kayıtları belirler ve bu kayıtları kullanarak tahminde bulunur yada bir sınıflandırma işlemi gerçekler. Bellek yaklaşımlı yöntemlerin en önemli özelliği veriyi olduğu gibi kullanabilme yeteneğidir. Diğer tekniklerin aksine bellek tabanlı yöntemler, kayıtların formatı yerine sadece iki işlemin varlığıyla ilgilenir. İki kayıt arasındaki uzaklığı belirleyen bir uzaklık fonksiyonu ve komşu kayıtları işleyerek bir sonuç üreten bir kombinasyon fonksiyonudur (Tantuğ, 2002).

4.2.3. Kümeleme

Büyük ölçekli verileri homojen kümelere ayırma işlemi veri madenciliğinin en temel işlemlerinden birisidir. Verilen belli bir örüntü kümesini alt kümelere böler. Bu işlem sonucunda, aynı kümeye giren kayıtlar arasındaki benzerlik diğer kümelerdeki kayıtlara göre çok daha fazla olur (Tantuğ, 2002).

4.2.4. İlişkisel Analiz

İlişkisel analiz, matematiğin bir alt alanı olan graf teorisi tabanlıdır. İlişkisel analiz her tür sorunu çözemez yada her türlü veri üzerinde uygulanamaz. İlişkisel analiz yönteminin uygulanmasında ortaya çıkan bir başka sorun ise maddeler arsındaki ilişkilerin ortaya çıkarılmasıdır. İlişkilerin bulunması için otomatikleşmiş bazı işlemlerin gerçeklenmesi gerekebilir (Tantuğ, 2002).

4.2.5. Karar Ağaçları ve Kural Türetme

Karar ağaçları yöntemi en güçlü ve en yaygın sınıflandırma ve öngörü araçlarından birisidir. Ağaç yapılı yöntemlerin sık kullanılmasının nedeni ise yapay sinir ağlarının tersine ağaç yapılarının kuralları ifade edebilmesinden kaynaklanmaktadır (Tantuğ, 2002).

Karar ağaçları bir dizi soru sorup bunların cevapları doğrultusunda hareket ederek en kısa sürede sonuca gider. Sorulan bir soruya gelen cevap ile sorulacak diğer sorular tespit edilir. Soruların kalitesine bağlı olarak yeni kaydın sınıflandırılması az sayıda soruyla gerçekleştirilmiş olur.

4.2.6. Yapay Sinir Ağları

Yapay sinir ağları; sınıflandırma, kümeleme ve tahmin amaçları ile kolaylıkla kullanılabilecek genel amaçlı ve güçlü araçlardır (Tantuğ, 2002).

Yapay sinir ağları öncelikle sonuçları bilinen belirli bir veri kümesi üzerinde öğrenme algoritmaları çalıştırılarak eğitilir. Bu eğitim neticesinde yapay sinir ağının içerisindeki bir takım ağırlıklar belirlenir. Bu ağırlıklar kullanılarak yeni gelen veriler işlenir ve bir sonuç üretilir. Yapay sinir ağlarının en olumsuz yönü ise bu ağırlıkların neden ilgili değeri aldığının bilinmemesidir. Yada çıkan sonucun neden geçerli bir sonuç olduğu açıklanamaz (Tantuğ, 2002).

4.2.7. Genetik Algoritmaların (GA) Veri Madenciliğinde Kullanımı

Genetik algoritmalar biyolojik işlemlerden kaynağını almıştır. Yıllar boyunca genetik algoritmalar yapay sinir ağlarının eğitilmesi, bellek tabanlı yöntemlerde kombinasyon fonksiyonunun oluşturulması gibi işlerde kullanılmışlardır (Tantuğ, 2002).

Genetik algoritmalar, açıklanabilir sonuçlar üretirler. Çok değişik tiplerdeki verileri işleme özelliğine sahip olan genetik algoritmalar optimizasyon amacı ile kullanılabilirler (Tantuğ, 2002).

5. LİTERATÜR TARAMASI

5.1. Esnek Akış Tipi Çizelgeleme Problemleri Literatür Taraması

Esnek akış tipi problemler üzerine geniş bir literatür bulunmasına rağmen, araştırmaların endüstriyel uygulamaları yok denecek kadar azdır. Gerçek çizelgeleme problemlerini çözmek için genellikle öncelik kurallarına dayalı sezgiseller kullanılmıştır. Fakat bu sezgiseller optimal veya optimale yakın sonuçları garanti etmemektedir. Kullanılan sezgisellerdeki önemli bir problem ise; işler sadece ilk kademede sıralanır, aynı sıra tüm kademeler boyunca devam eder. Halbuki optimalliğe ulaşmak için her kademede işler yeniden çizelgelenmelidir (Döyen,2004).

NP-zor olarak bilinen akış tipi çizelgeleme problemlerinin GA ile çözüm kalitesi, GA' da kullanılan parametrelere bağlıdır. Genetik Algoritmada, optimum veya optimuma yakın çözüm veren parametreler, problemlerin yapısına göre değişmektedir (Engin ve Fığlalı, 2002).

Birçok araştırmacı iki aşamalı hibrid akış tipi çizelgeleme problemleri üzerinde çalışmışlardır. Sezgisel çözümler önermişlerdir. Çoğunluğu iki bölümden oluşur. Birinci bölümde genellikle Johnson kuralları kullanılarak bir sıra oluşturulur. İkinci bölümde bu sıradan, iki aşama için farklı makinelerde iş sıraları oluşturulur. K aşamalı esnek akış tipi problemler için araştırmacılar ya genel problemler için dal sınır algoritması, kısıtlayıcı çözümler kümesi için permütasyon çizelgeleme gibi klasik yöneylem araştırması yaklaşımları ya da tabu arama, tavlama benzetim yaklaşımları gibi lokal arama metotları kullanmışlardır.

Arthanari ve Ramaurthy (1971) **e**snek akış tipi çizelgeleme problemi üzerine ilk çalışma yı yapmışlardır. C_{max} minimizasyonu için bir dal-sınır yöntemi geliştirmişlerdir.

Narasimhan ve Panwalker (1984) kablo üretimiyle ilgili endüstriyel çok aşamalı, çok ürünlü akış tipi çizelgeleme problemi üzerine çalışmışlardır. Çalışma sonucunda kümülatif minimum sapma kuralı olarak bilinen sezgisel bir çözüm tekniği geliştirmişlerdir. Bu kurallar diğer standart kurallara göre daha iyi sonuçlar vermiştir.

Wittrock (1985, 1988), toplam akış zamanı ve stok alanını minimize etmek için iki farklı sezgisel (esnek akış tipi problemler için periyodik olmayan çizelgeleme algoritması) geliştirmiştir.

Kochar ve Morris (1987) sıralamaya bağlı hazırlık süreli esnek akış tipi problemler için sezgiseller geliştirmişlerdir.

Gupta (1988), esnek akış tipi çizelgeleme problemleri tamamlanma zamanı kriterine göre NP-zor olduğunu kanıtlamıştır.

Sriskandarajah ve Sethi (1989), minimum C_{max} amacı için çeşitli algoritmaların performanslarını karşılaştırmıştır.

Brah ve Hunsucker (1991), en düşük alt sınır arama tekniğinin C_{max}'ı minimize etmek için kullanmıştır.

Gupta ve Tunc (1991)**,** esnek akış tipi çizelgeleme problemlerinin C_{max}'ı minimize etmeye yönelik olurlu bir çözüm bulmak için iki tane polinomial sınırlı sezgisel algoritma geliştirmişlerdir.

Rajendran ve Chaudhuri (1992), toplam tamamlanma zamanı ve toplam akış zamanını minimize etmek için dal-sınır algoritmaları geliştirmiştir**.**

Ding ve Kittichartphayak (1994), üç tane sezgisel yöntemi incelemişler (iki tane sezgisel her aşamada bir makine olan esnek akış tipi problem için geliştirilmiş, üçüncü sezgisel de ise her aşamada bir makine olan esnek akış tipi problem için sıralı yapıcı çizelgeleme algoritması geliştirilmiş) ve hesaplama sonuçlarını karşılaştırmışlardır.

Santos ve ark. (1995), C_{max} minimizasyon amaçlı esnek akış tipi problem için global alt sınırlar önermiştir. Optimal çözüm bulunamadığında sezgisellerin kalitesini değerlendirmek için bu global alt sınırların kullanılabilineceğini göstermişlerdir.

Gupta ve ark. (1997), arama ağacının sınırlarını daraltmak için bir baskınlık kuralı oluşturmuşlardır, bir dallandırma kuralı ilave edilmiş ve başlangıç üst sınırını oluşturmak için 13 farklı sezgisel kullanılmıştır.

Riane ve ark. (1998), günlük hayattan n iş 3 aşamalı (birinci ve üçüncü aşamada bir makine, ikinci aşamada iki makine) bir esnek akış tipi problemin akış zamanını minimize etmek için sezgiseller geliştirmişlerdir.

Portmann ve ark.(1998), *k*-aşamalı hibrid flowshop çizelgeleme problemi için optimal bir metot geliştirmişlerdir. Problemin çözümünde dal sınır algoritmasını kullanmıştır. Çalışmalarında Brah ve Hunsucker dal sınır metodunu başlangıç noktası olarak belirlemişlerdir.

Brah ve Loo (1999), beş farklı sezgisel metodun performansını farklı yapıdaki problemler üzerinde test etmişlerdir. Her aşamadaki iş sayısı, makine sayısı ve paralel işlemci sayısının etkisini ve sezgisellerin performansını regrasyon analiziyle incelemişlerdir. Performans üzerinde sezgisellerin etkisi olmasına rağmen prblemin karekteristiklerinin etkisinin daha fazla olduğunu göstermişlerdir.

Moursli ve Pochet (2000), hibrid akış tipi problemler için bir etkili dal-sınır algoritması geliştirmişlerdir. Literatürdeki diğer metotlara göre daha kısa sürede daha etkili sonuçlar vermiştir.

Neron ve ark. (2000), m-makine problemleri için alt sınırı daha etkili hale getiren bir dal-sınır algoritması geliştirmişlerdir. Yeterlilik testleri ve zaman-sınırı ayarlamaları kullanımının dal-sınır algoritmalarının verimini artırdığını göstermişlerdir**.**

Negenman (2001), çizelgeleme problemleri için literatürdeki yerel arama algoritmalarını incelemiş ve bunları esnek akış tipi problemlere uyarlamıştır. Tabu arama ve tavlama benzetim algoritmalarını kullanmışlardır.

Azizoğlu ve arkadaşları (2001), toplam akış zamanını minimize etmek için esnek akış tipi çizelgeleme probleri üzerinde çalışmışlardır. Optimal çizelgeyi belirleyecek bir dal-sınır algoritması geliştirmişlerdir. Algoritmanın etkinliği, alt ve üst sınırlara ve baskın kriterlere göre değişmektedir.

Su (2002), iki aşamalı 1. aşamada çoklu işlemci ve 2. aşamada tek işlemcili sınırlı bekleme zamanlı esnek akış tipi problemi üzerine çalışılmıştır. Amaç makespan minimize etmektir. Problemin çözümü için bir sezgisel algoritma ve kıyaslama için karmaşık tamsayılı bir program geliştirilmiş.

Wardono ve Fathi (2003), aşamalar arasında sınırlı ara stok kapasitesine sahip n iş l aşamalı esnek akış tipi problemler için bir tabu arama algoritması geliştirmiştir.

Chung ve Vairaktarakis (2003), iki aşamalı esnek akış tipi problemi üzerinde çalışmışlardır. Bunun için dal sınır algoritması kullanarak bir sezgisel geliştirmişlerdir. Bu çalışma genel esnek akış tipi problemi için ilk hata sınır algoritması ve geleneksel k makine akış tipi problemi için geçerli en iyi hata sınırı geliştirmiştir.

Oğuz C. ve arkadaşları (2004), geliştirdikleri çok işlemli hibrid flowshop problemi için tabu arama yöntemiyle bir sezgisel geliştirmişlerdir.

Döyen (2004), GA'nın performansının artırılmasında, öğrenen temelli GA modellerini önermişler ve bu modeli klasik yöntemler (NEH, CDS) ile karşılaştırarak etkinliğini belirlemiştir.

Şerifoğlu (2004), her biri birden çok işlemcide işlenmesi gereken n adet iş için m katmanlı bir parelel işlemcili akış astölyesinde çizelgelenmesi problemi ele alınmıştır. Bu problemi çözmek üzere bir GA geliştirmişler ve kısa sürede etkin çözümler elde etmişlerdir. En son aşamada tüm işlemlerin tamamlandığı zamanın enküçüklenmesi amaçlanmıştır.

Wang ve arkadaşları (2005), iki aşamalı (birinci aşamada bir işin iki operasyonu arasında bekleme zamanı olmayan ve ikinci aşamada ardışık iki iş arasında makine beklemesi olamayan) esnek akış tipi çizelgeleme problemlerinin karmaşıklığını göstermişlerdir.

Oğuz C. ve arkadaşları (2005), geliştirdikleri çok işlemli hibrid flowshop problemi için genetik algoritmalar yöntemiyle bir sezgisel geliştirmişlerdir.

Allaoui ve Artiba (2006), c_{max} minimize etmek için iki aşamalı (birinci aşamada bir makine ikinci aşamada m makine) esnek akış tipi çizelgeleme problemi üzerine çalışmışlardır. Problem için bir dal sınır modeli oluşturulmuştur.

Tang ve arkadaşları (2006), n iş s aşamadan oluşan esnek akış tipi çizelgeleme problemi üzerine çalışmışlardır. C_{max}'ı minimize eden bir çizelgeleme oluşturmak için tam sayılı programlama yöntemi, Lagrangian gevşetmesi ve dinamik programlama kullanılmıştır.

Jin ve arkadaşları (2006), çok aşamalı esnek akış tipi çizelgeleme problemleri üzerine çalışmışlardır. Tavlama benzetim ve değişken derinlik araştırma yöntemleriyle bir sezgisel geliştirmişlerdir.

Tablo 1'de ve tablo 2'de esnek akış tipi problemlerle ilgili yapılan önemli çalışmalar, kullanılan çözüm metoduna göre sınıflandırılarak gösterilmiştir.

Tablo 1. Esnek Akış Tipi Çizelgeleme Problemi ile Dal Sınır Yaklaşımı Kullanılarak Yapılan Çalışmalar

DAL-SINIR YAKLAŞIMLARI	
Yazar(lar)	**Yıl**
Arthanari,T.S., Ramamurthy, K.G.	1971
Brah S.A., Hunsucker, J.L	1991
Rajendran, C., Chaudhuri, D.	1992
Santos, D.L. ve ark.	1995
Gupta, J.N.D ve ark.	1997
Portmann M.C. ve arkadaşları	1999
Moursli O., Pochet Y.	2000
Neron, E. ve ark.	2001
Azizoğlu ve ark.	2001
Chung Y. L., Vairaktarakis G.L.	2003
Allaoui ve Artiba	2006

Tablo 2.Esnek Akış Tipi Çizelgeleme Problemi ile Sezgisel Yaklaşımlar Kullanılarak Yapılan Çalışmalar

SEZGİSEL YAKLAŞIMLAR	
Yazar(lar)	**Yıl**
Narasimhan,S.L., Panwalker, S.	1984
Wittrock, R.J.	1985 1988
Kochar, S., Morris, R.J.T.	1987
Sriskandarajah, C., Sethi, S.P.	1989
Gupta, J.N.D., Tunc, E.A.	1991 1994
Ding, F.Y., Kittichartphayak, D.	1994
Riane, F. ve ark.	1998
Brah, S.A., Loo, L.L	1999
Negenman E.G.	2001
Su L.H.	2002
Wardono, B., Fathi, Y.	2003
Oğuz C. ve arkadaşları	2004
Engin O., Döyen A.	2004
Serifoğlu ve ark.	2004
Oğuz C. ve arkadaşları	2005
Wang ve ark.	2005
Tang ve arkadaşları	2006
Jin ve arkadaşları	2006

5.2. Veri Madenciliği – Çizelgeleme Problemleri Literatür Taraması

Çizelgeleme problemlerinin çözümünde birçok teknik kullanılmıştır. Bu tekniklerden bir tanesi de veri madenciliğidir. Ancak veri madenciliğinin çizelgeleme problemlerine uygulanması ile ilgili olarak çok az çalışma yapılmıştır.

Çizelgeleme problemlerinin çözümünde veri madenciliğini kullanan çalışmalar;

- D. A. Koonce ve S. C. Tsai (2000) : Bir atölye tipi çizelgelemede genetik algoritmalar ile elde edilen sonuçların modellerinin veri madenciliği kullanılarak bulunması,
- Youssef Harrath, Brigitte Chebel-Morello, Noureddine Zerhouni (2001) : Bir atölye tipi çizelgelemeyi çözümlemede genetik algoritma ve veri madenciliği,
- Youssef Harrath, Brigitte Chebel-Morello, Noureddine Zerhouni (2002): Bir atölye tipi çizelgeleme problemi için bir genetik algoritma ve veri madenciliği tabanlı meta sezgiseldir.

Çalışmalar hakkında genel bilgi Tablo 3'de gösterilmiştir.

Tablo 3. Veri madenciliğinin çizelgeleme problemlerine uygulanması

Çalışmanın Adı	Bir Atölye Tipi Çizelgelemede GA İle Elde Edilen Sonuçların VM Kullanılarak Bulunması	Bir Atölye Tipi Çizelgelemeyi Çözümlemede GA Ve VM	Bir Atölye Tipi Çizelgeleme Problemi İçin Bir GA Ve VM Tabanlı Meta Sezgisel
Yazarlar	D. A. Koonce S. C. Tsai	Youssef Harrath B. Chebel-Morello Noureddine Zerhouni	Youssef Harrath B. Chebel-Morello Noureddine Zerhouni
Yayın yılı	2000	2001	2002
Problem	6X6 Muth &Thompson Kıyaslama Problemi	6X6 Muth &Thompson Kıyaslama Problemi	6X6 Muth &Thompson Kıyaslama Problemi
Kullanılan Teknik	Genetik Algoritmalar	Genetik Algoritmalar	Genetik Algoritmalar
Başlangıç Populasyonu	500	1000	1000
Başlangıç Populasyonu Seçimi	Rassal	Rassal	Rassal
Çalıştırma Sayısı	1000	1000	1000
Çaprazlama	Sıralı Çaprazlama	Pozisyona Dayalı Çaprazlama	Pozisyona Dayalı Çaprazlama
Çaprazlama Olasılığı	$P_{cross} = 0,60$	$P_{cross} = 0,1$	$P_{cross} = 0,1$
Mutasyon	Sıra Temelli Çaprazlama	Rassal	Rassal
Mutasyon Olasılığı	$P_{mut} = 0,20$	$P_{mut} = 0,01$	$P_{mut} = 0,01$
Sonuç	Optimal çözüm tüm çalıştırmada bulunmuştur. 264 farklı çözüm V. M.'de kullanılmıştır.	%92.7 oranında optimal çözüm bulunmuştur. 106 farklı çözüm V. M.'de kullanılmıştır.	%92.7 oranında optimal çözüm bulunmuştur. 106 farklı çözüm V. M.'de kullanılmıştır.
Veri Hazırlanması	Sınıflandırma	Sınıflandırma	Chimerge Algoritması (Sınıflandırma)
V.M Algoritması	Özellik Temelli İndirgeme Algoritması	Karar Ağacı	Chimerge Algoritması

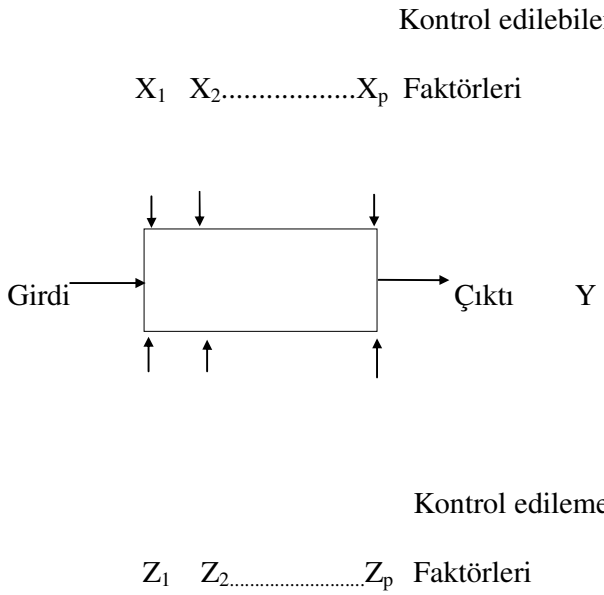
6. PARAMETRE OPTİMİZASYONU

Herhangi bir kombinatoriyel optimizasyon problemi çözülürken kısa sürede optimal veya optimal yakını çözüm elde etmek için, problem parametrelerinin uygun değerleri kullanılmalıdır. Çalışmada önerilen genetik algoritmalarda da bazı parametreler kullanmaktadır. Herhangi bir problemin çözümünde kullanılan GA algoritması için optimum veya optimuma yakın çözüm veren parametre seti, başka bir GA uygulaması için genelleştirilemez (Döyen, 2004). Bu nedenle ele aldığımız esnek akış tipi problemler için ayrı ayrı parametre optimizasyonu yapılmıştır. GA parametreleri çok adımlı bir deney tasarımı yaklaşımı ile geliştirilmiştir. Geliştirilen deney tasarımı yaklaşımı, Taguchi metoduna dayanmaktadır.

6.1. Deney Tasarımı

Deney tasarımı bir deney serisi olup, bir prosesin girdi değişkenlerinin amaçlar doğrultusunda değiştirilip, prosesin gözlenmesi ve sistem çıktısındaki değişkenliklerin belirlenmesidir. Proses, insan, makine ve metot kombinasyonundan oluşan ve girdi bileşenlerini çıktı ürününe çeviren bir sistemdir. Genel bir proses modeli, Şekil 12' de görülmektedir (Taguchi ve Phadke 1989).

Proses değişkenlerinin bir kısmı X_1, X_2,……, X_p gibi kontrol edilebilen, bir kısmı da Z_1, Z_2, ……. , Z_p gibi kontrol edilemeyen girdi faktörleri olabilir. Bu kontrol edilemeyenlere gürültü faktörleri denilmektedir.



Şekil 12. Genel bir proses modeli.

Bir deney tasarımının amaçları genel olarak şunlardır,

1) Çıktı üzerinde hangi faktörlerin en etkin olduğunu belirlemektir.

2) Çıktı değeri Y' nin nominal değerine yakın olabilmesi için, önemli olan X kontrol edilebilen girdi faktörlerinin seviyelerini belirlemektir.

3) Kontrol edilemeyen değişkenlerin (Z) etkilerini en aza indirmek için, X faktörlerinin seviyelerini belirlemektir.

Deney tasarım metotları, bir prosesin istatistiksel olarak kontrol altında tutulması için de kullanılabilir. Örneğin; kontrol şeması, prosesin kontrol dışı olduğunu gösterdiğinde, prosesin tekrar kontrol altına alınması için hangi girdi değişkenlerinin

önemli olduğunun belirlenmesi gerekir. Deney tasarımı ile proses üzerinde en etkili olan, kontrol edilebilir girdi faktörleri tespit edilebilir (Döyen, 2004).

6.2. Esnek Akış Tipi Çizelgeleme Problemleri için Parametre Optimizasyonu

Bu çalışmada kullanılan esnek akış tipi problemler Carlier ve Neron (2000)' de verilen kıyaslama problemleri ile Oğuz kıyaslama problemleridir. Carlier ve Neron problemleri *n(iş) x s(aşama)* tipi problemlerin boyutları: 10x5, 10x10, 15x5, 15x10 dır. İşlem süreleri [3,20] aralığında uniform dağılmaktadır. Problemler; iş sayısı ve kademe sayıları özelliklerine göre gruplandırılabilir. Bir problemin yapısını bu iki özellik belirlemektedir. Örnek bir problem notasyonu: "*j10c5a2*" şeklindedir. Burada; *j10*, 10 iş bulunduğunu; *c5*, 5 kademe bulunduğunu; *a*, kademelerdeki makine yerleşimi yapısını; en sondaki 2 ise örnek indisini göstermektedir. Kademelerdeki makine yerleşimleri aşağıdaki şekildedir:

a: Orta kademede 1 makine (darboğaz), diğer kademelerde 3 makine vardır.

b: İlk kademede 1 makine (darboğaz), diğerlerinde 3 makine vardır.

c: Orta kademede 2 makine(darboğaz), diğer kademelerde 3 makine vardır.

d: Her kademede 3 makine var (darboğaz olan kademe yoktur).

Ele alınan problemler, belirtilen iki özelliğe göre 4 gruba ayrılabilir. Her gruptan alınan bir örnek, diğer geri kalanları da temsil etmektedir, dolayısıyla alınan örnek problem için yapılan parametre optimizasyonu gruptaki diğer örnekler içinde geçerlidir. Bu nedenle her gruptan bir örnek problem alınmış ve bu 4 örnek için parametre optimizasyonu çalışması yapılmıştır. Her örnek için bulunan optimal parametre seti, gruptaki diğer

problemlerin çözümünde de kullanılmıştır. Ele alınan örnek problemler: *j10c10b3, j10c5a2, j15c10b1, j15c5b3*'dür.

Oğuz kıyaslama problemlerinde *n(iş) x s(aşama)* tipi problemlerdir. Örnek problem notasyonu "P50S8T05" şeklindedir. Burada P problem zorluk derecesini göstermekte olup P ve Q değerlerini almaktadır. Q tipi problemler daha zordur. 50 sayısı iş sayısını göstermektedir. Burada 100 iş H1 ile temsil edilmektedir. S8 aşama sayısını, T05 ise problem indisini göstermektedir. n=5,10,20,50,100 iş sayıları, m=2,5,8 değerleri ise aşama sayılarıdır. Çözümü gerçekleştirilecek olan problemlerin boyutları: 5x2, 10x2, 20x2, 50x2, 100x2, 5x5, 10x5, 20x5, 50x5, 100x8, 5x8, 10x8, 20x8, 50x8 ve 100x8,' dir. Her işlemci herhangi bir anda sadece bir işlemi yürütebilmektedir. İşlemciler arıza yapmazlar. Bütün işler çizelgeleme başında hazırdırlar. İşler kesintisiz işlenir. Amaç, C_{max}'ın en küçüklenmesidir.

Bu problemleri çözmek için geliştirilen GA 'da 6 parametre kullanılmaktadır. Her ayrı tip problem türü için gerekli parametreleri belirlemek amacıyla olası tüm kombinasyonlar *tam faktöriyel deney* düzeneği olarak kurulmuştur. Her faktörde birden fazla düzey bulunmaktadır. Bu düzeyler Tablo 4' de verilmiştir.

Tablo 4. Faktörler için kullanılan düzeyler.

Üreme Yöntemi	Rulet Çemberi
	Tournament
Üreme Oranı	0,1' den 0,9' a kadar
Çaprazlama Yöntemi	Pozisyona Dayalı
	Sıraya Dayalı
	Kısmı Planlı
	Dairesel
	Doğrusal Sıralı
	Sıralı
Çaprazlama Oranı	0,1' den 0,9' a kadar
Mutasyon Yöntemi	Ters Mutasyon
	Komşu İki İşi Değiştirme
	Keyfi İki İşi Değiştirme
	Keyfi Üç İşi Değiştirme
	Araya Sokma
Mutasyon Oranı	0,1' den 0,9' a kadar

Parametrelerin belirlenebilmesi için seçilen her bir problem $2\times9\times6\times9\times5\times9=43740$ defa çözülmüştür. Çözüm sonuçları için en önemli faktör C_{max} değeridir. C_{max} değerinden sonra dikkate alınan çözüm süresidir. Çözüm süreleri eşit ise iterasyon sayıları dikkate alınmaktadır. Bu çalışma için geliştirilen GA, n adet işin sıralanmasına dayalı bir kromozom yapısı kullanmaktadır. En iyi sonucu veren parametreler benzer problemlerin çözümünde kullanılacaktır. Tablo 2' de seçilen örnek problem için en iyi sonucu veren parametreler verilmiştir. Bu değerler her bir örneğin çözümünden elde edilen en iyi sonucu göstermektedir. Her bir örnek için en iyi ilk 30 sonuç Ek 1' de bulunabilir.

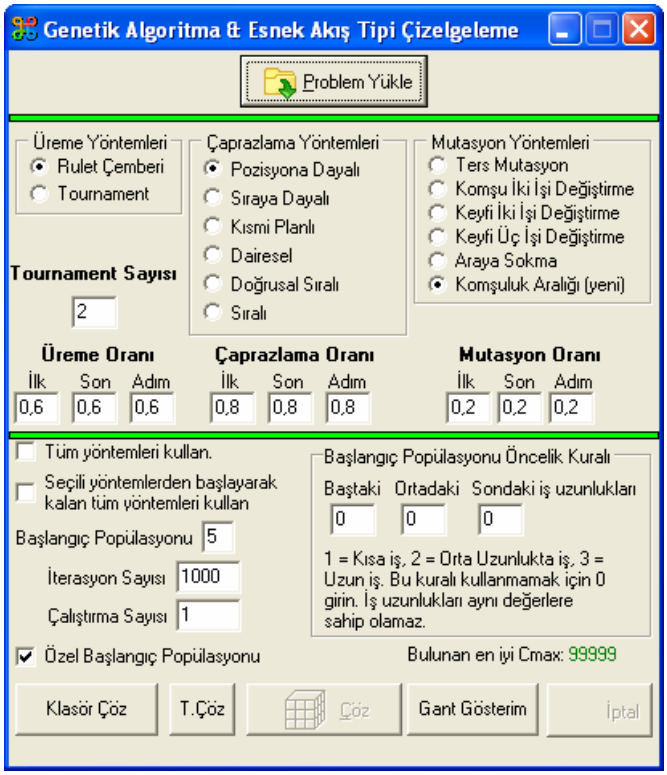
Tablo 5. Seçilen örnekler için en iyi sonucu veren parametreler.

PROBLEM	ÜREME	ÜREME ORAN	ÇAPRAZLAMA	ÇAPR. ORANI	MUTASYON	MUT. ORAN	CMAX	İTERASYON	ZAMAN
P5S2T05	Rulet Çemberi	0,2	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	267	1	00:00.000
P10S2T05	Rulet Çemberi	0,4	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	526	1	00:00.000
P20S2T05	Rulet Çemberi	0,4	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	869	1	00:00.000
P50S2T05	Rulet Çemberi	0,8	Sıraya Dayalı	0,8	Araya Sokma	0,5	1729	11	00:03.109
P1S2T05	Rulet Çemberi	0,2	Sıraya Dayalı	0,3	Keyfi Üç İşi Değiştirme	0,2	5094	1	00:00.203
P5S5T05	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Ters Mutasyon	0,2	511	1	00:00.015
P10S5T05	Rulet Çemberi	0,9	Sıraya Dayalı	0,6	Komşu İki İşi Değiştirme	0,6	591	1	00:00.063
P20S5T05	Rulet Çemberi	0,9	Dairesel	0,9	Araya Sokma	0,6	972	12	00:02.656
P50S5T05	Rulet Çemberi	0,7	Sıralı	0,4	Komşu İki İşi Değiştirme	0,1	2638	9	00:01.657
P1S5T05	Rulet Çemberi	0,5	Sıralı	0,3	Araya Sokma	0,5	4985	7	00:03.515
P5S8T05	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Komşu İki İşi Değiştirme	0,1	616	1	00:00.015
P10S8T05	Rulet Çemberi	0,2	Dairesel	0,2	Araya Sokma	0,9	767	2	00:00.172
P20S8T05	Rulet Çemberi	0,5	Sıraya Dayalı	0,7	Komşu İki İşi Değiştirme	0,2	1208	11	00:01.328
P50S8T05	Rulet Çemberi	0,7	Pozisyona Dayalı	0,3	Keyfi Üç İşi Değiştirme	0,9	2711	16	00:06.547
P1S8T05	Rulet Çemberi	0,1	Kısmi Planlı	0,9	Araya Sokma	0,9	5219	8	00:10.250
J10C5A2	Rulet Çemberi	0,4	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	88	1	00:00.000
J10C10B3	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Ters Mutasyon	0,1	169	1	00:00.000
J15C5B3	Rulet Çemberi	0,1	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	157	1	00:00.000
J15C10B1	Rulet Çemberi	0,2	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	222	1	00:00.000

7. ESNEK AKIŞ TİPİ ROBLEMLERİNİN GENETİK ALGORİTMALAR KULLANARAK VERİ MADENCİLİYLE ÇÖZÜMÜ

7.1. Veri Toplama

Esnek akış tipi çizelgeleme problemlerini, GA ile çözen program Borland Delphi 7.0 ile geliştirilmiştir. Programın kaynak kodları Ek2' de sunulmuştur. Programın örnek ekran görüntüsü Şekil 13' de görülmektedir.



Şekil 13. Programa ait ekran görüntüsü.

Hazırlanan programı Intel Pentium 4 3 GHz işlemcili, 512 Mb ram ve Microsoft Windows XP with SP1 işletim sistemi bulunan bilgisayar üzerinde çalıştırılmıştır.Test

edilen esnek akış tipi problemler; Carlier ve Neron (2000) tarafından önerilen kıyaslama problemleri ve Oğuz (2004) tarafından önerilen kıyaslama problemleridir.

Optimal çözümleri elde etmek pek mümkün olmadığından, literatürde genellikle sezgisel bir metodun çözüm kalitesi, problemler için alt-sınır değerlerine ne kadar yakın çözüm verdiğine göre değerlendirilmektedir.

Yapılan çalışmada Carlier ve Neron (2000) tarafından önerilen kıyaslama problemleri için elde edilen sonuçlar Döyen (2004) tarafından Yapay Bağışıklık Sistemleri (YBS) ile Oğuz tarafından önerilen kıyaslama problemleri (2004-2005) için elde edilen sonuçlar ile karşılaştırılmıştır.

Döyen (2004), çalışmasında algoritmanın LB sınırına ulaşana kadar ya da en fazla 1600 sn çalışmasını sağlamıştır. Bu çalışmada daha önce bulunan LB' ler göz ardı edilerek her problem 50 defa çalıştırılmıştır. Böylece belirsizlik ortamında algoritmanın ne kadar güvenilebilir olduğu ortaya konulmaya çalışılmıştır.

Yapılan testlerde GA algoritmasının çözüm kalitesi ve çözüme ulaşmada harcadığı süre (CPU zamanı) ölçülmüştür. Çözüm kalitesi, algoritmanın elde ettiği en iyi çözüm ile problemin alt sınır değeri arasındaki yüzde sapma miktarı ile ölçülmektedir. Yüzde sapma (1)' deki ifadeye göre hesaplanır.

$$\%\ Sapma = \frac{A\lg oritma\ Çözümü - LB}{LB} \quad \text{.....} 1$$

Karşılaştırma yapılan çalışmada kullanılan bilgisayar sistemi P4 1,7 Ghz, 256 Mb Ram' de yapılmıştır. Programın kullandığı ram miktarı ortalama 16 Mb olduğundan ram' in fazla bir önemi yoktur. GA ile problemlerin çözüldüğü sistem ise P4 3 Ghz işlemciye sahiptir. Beklenen yüzde sapma GA ile problemlerin çözüldüğü sistem ile karşılaştırma

yapılan bilgisayar sistemin işlencileri farkının, GA ile problemlerin çözüldüğü sistemin işsemci değerine bölümüyle ölçülmektedir.

$$\%\ Beklenen\ Sapma = \frac{3-1,7}{3} \cong \%43$$

Yapılan çalışmanın iyi sayılabilmesi için CPU zamanlarında en az %43' lük bir iyileşmenin olması beklenmelidir. Carlier ve Neron (2000) tarafından önerilen kıyaslama problemleri çözümlerinden elde edilen C_{max} ve CPU zamanları karşılaştırmalı olarak Tablo 6 ve 7'de sunulmuştur.

Tablo 6. Esnek akış tipi problemler için elde edilen

Problem	GA C_{max}	GA CPU	YBS C_{max}	YBS CPU	LB C_{max}	GA % Sapma	YBS % Sapma
j10c5a2	88	0	88	0,046	88	0	0
j10c5a3	117	0	117	0,141	117	0	0
j10c5a4	121	0,015	121	0,171	121	0	0
j10c5a5	122	0	122	0,906	122	0	0
j10c5a6	110	0,015	110	3,578	110	0	0
j10c5b1	130	0	130	0,14	130	0	0
j10c5b2	107	0	107	0,796	107	0	0
j10c5b3	109	0	109	0,516	109	0	0
j10c5b4	122	0	122	1,656	122	0	0
j10c5b5	153	0	153	0,124	153	0	0
j10c5b6	115	0	115	0,14	115	0	0
j10c5c1	69	0,031	68	31,624	68	0,014493	0
j10c5c2	74	0,016	74	4,14	74	0	0
j10c5c3	72	0,016	72		71	0,013889	0,013889
j10c5c4	66	0,031	66	2,64	66	0	0
j10c5c5	78	0,094	78	13,827	78	0	0
j10c5c6	70	0	69	12,141	69	0,014286	0
j10c5d1	66	0,046	66	4,891	66	0	0
j10c5d2	74	0,11	73	31,156	73	0,013514	0
j10c5d3	64	0,015	64	14,781	64	0	0
j10c5d4	71	0	70	4,5	70	0,014085	0
j10c5d5	68	0,031	66	1445,641	66	0,029412	0
j10c5d6	62	0,062	62	7,61	62	0	0
j10c10a1	139	0,015	139	1,187	139	0	0
j10c10a2	158	0,125	158	17,782	158	0	0
j10c10a3	148	0,047	148	0,531	148	0	0
j10c10a4	149	0,141	149	1,766	149	0	0
j10c10a5	148	0	148	0,563	148	0	0
j10c10a6	146	0,156	146	3,562	146	0	0
j10c10b1	163	0	163	0,187	163	0	0
j10c10b2	157	0,131	157	0,689	157	0	0
j10c10b3	169	0	169	0,031	169	0	0
j10c10b4	159	0,015	159	0,109	159	0	0
j10c10b5	165	0,016	165	0,03	165	0	0
j10c10b6	165	0,016	165	0,39	165	0	0
j10c10c1	116	0,062	115	---	113	0,025862	0,017391
j10c10c2	117	0,141	119	---	116	0,008547	0,02521
j10c10c3	117	0,234	116	---	98	0,162393	0,155172
j10c10c4	121	0,281	120	---	103	0,14876	0,141667
j10c10c5	126	0,721	126	---	121	0,039683	0,039683

Tablo 7. Esnek akış tipi problemler için elde edilen sonuçlar (devam)

Problem	GA C_{max}	GA CPU	YBS C_{max}	YBS CPU	LB C_{max}	GA % Sapma	YBS % Sapma
j15c5a1	178	0,031	178	1,094	178	0	0
j15c5a2	165	0,015	165	0,25	165	0	0
j15c5a3	130	0,015	130	0,249	130	0	0
j15c5a4	156	0,015	156	1,39	156	0	0
j15c5a5	164	0,046	164	0,266	164	0	0
j15c5a6	178	0,032	178	0,265	178	0	0
j15c5b1	170	0,015	170	0,218	170	0	0
j15c5b2	152	0,015	152	0,218	152	0	0
j15c5b3	157	0,015	157	0,234	157	0	0
j15c5b4	147	0,015	147	0,25	147	0	0
j15c5b5	166	0,016	166	1,968	166	0	0
j15c5b6	175	0,015	175	0,609	175	0	0
j15c5c1	86	0,031	85	774,484	85	0,011628	0
j15c5c2	91	0,156	91	---	90	0,010989	0,010989
j15c5c3	87	0,109	87	16,437	87	0	0
j15c5c4	90	0	89	316,937	89	0,011111	0
j15c5c5	76	0,063	74	---	73	0,039474	0,013514
j15c5c6	91	0,047	91	18,797	91	0	0
j15c5d1	167	0,015	167	0,234	167	0	0
j15c5d2	86	0,406	84	---	82	0,046512	0,02381
j15c5d3	84	0,015	83	---	77	0,083333	0,072289
j15c5d4	86	0,188	84	---	61	0,290698	0,27381
j15c5d5	81	0,105	80	---	67	0,17284	0,1625
j15c5d6	83	0,406	82	---	79	0,048193	0,036585
j15c10a1	236	0,015	236	0,531	236	0	0
j15c10a2	200	0,015	200	29,515	200	0	0
j15c10a3	198	0,063	198	4,312	198	0	0
j15c10a4	225	0,031	225	11,5	225	0	0
j15c10a5	182	0,016	182	1,843	182	0	0
j15c10a6	200	0,031	200	1,718	200	0	0
j15c10b1	222	0,031	222	2,656	222	0	0
j15c10b2	187	0,047	187	0,484	187	0	0
j15c10b3	222	0,015	222	0,437	222	0	0
j15c10b4	221	0,016	221	0,452	221	0	0
j15c10b5	200	0,094	200	0,468	200	0	0
j15c10b6	219	0,031	219	0,452	219	0	0

Tablo 6ve 7' de yer alan GA' ya ait CPU zamanları saniye cinsinden, YBS' ye ait olanlar ise dakika cinsindendir.

10 iş x 5 kademe boyutlu problemler için; a, b, c, d yapılarında toplam 23 problem çözülmüştür. Toplam 17 problemin hepsi için alt sınır değerler (optimal sonuçlar) elde edilmiştir. c ve d tipi problemlerden bazılarında ise Döyen'den %2,9 ile %1,3 arasında değişen oranlarda daha kötü değer bulunmuştur. 6 problem için optimal sonuç bulunamazken, Döyen sadece 1 problem için bulamamıştır. Bunlardan 1 tanesi optimal sonucu bulamamasına rağmen Döyen ile aynı değeri bulmuştur.

10 iş x 10 kademe boyutlu problemler için a, b, c yapılarında toplam 18 problem çözülmüştür. a ve b türünde 12 problem için optimal sonuçları bulunmuştur. c türündeki problemler için Döyen'e göre daha kötü sonuçlar bulunmasına rağmen bir problem için (j10c10c2) daha iyi sonuç bulunmuştur. Döyen a ve b türünde 12 problem için optimal sonuçları bulunmuştur.

15 iş x 5 kademe boyutlu problemler için a, b, c, d yapılarında toplam 24 problem çözülmüştür. 9 problem için optimal sonucu bulunamazken, Döyen sadece 7 tanesi için optimal değeri bulamamıştır.

15 iş x 10 kademe boyutlu problemlerde, a ve b tipi (toplam 12 problem) tüm problemlerde optimal sonuçlara ulaşılmıştır. Döyen a ve b türünde 12 problem için optimal sonuçları bulunmuştur.

Dikkat edileceği üzere, a ve b tipi problemlerde aynı çözümler elde edilirken, c ve d tipi problemlerde daha çok sapma görülmektedir. Makine yerleşimi yapısı problemin zorluğu üzerinde önemli bir etkiye sahiptir. Problemin büyüklüğü ise GA' yı etkilememektedir.

77 problemin tamamı göz önüne alındığında, GA algoritmasının alt sınır değerlerinden ortalama sapması % 1,23 iken; YBS yönteminin ortalama sapması % 1,01 olmuştur. Problemleri çözmek için harcanan CPU zamanlarına bakılırsa önerilen sezgisel

algoritmanın verimliliği hakkında bir fikir sahibi olunabilir. Alt sınır değerine ulaşılmış 55 problem için YBS algoritmasının harcadığı toplam süre 54,368 dk.'dır. GA yönteminin alt sınır değerlerine ulaştığı 55 problem için harcadığı toplam süre ise sadece 0,405 saniyedir. Bu açıdan bakıldığında GA, YBS' ye kötü sonuçlar vermesine rağmen çok kısa sürede sonuca ulaşılabilmiştir.

Neron ve ark. (2001) 10x5 ve 15x5 problemlerin c ve d tiplerini (toplam 24 problem) zor örnekler olarak sınıflandırmıştır. Alt sınır değere ulaşma yüzdeleri kolay örnekler için her iki yöntemle de eşittir. Zor örnekler için genetik algoritmalar daha az sayıda alt sınıra ulaşmıştır. Bu tip problemler yerel optimum noktaları sayısı ve olurlu çözüm sayısı göz önüne alınırsa daha karmaşık problemlerdir. Bu nedenle herhangi bir sezgisel arama tekniğinin lokal optimuma takılıp kalma şansı daha yüksektir. Unutulmaması gereken bir nokta da; problemler için verilen alt sınır değerleri, optimum değerlerden çok daha düşük olabilirler. Sonuçta bulunan değerler belki de optimum çözümlerdir, bunlardan daha iyi bir çözüm yoktur. Bu sorunun cevabı ancak; esnek akış tipi problemler optimal olarak çözüldüğü zaman verilebilir (Döyen, 2004).

Oğuz'un (2004) kıyaslama problemleri (toplam 240) çözülmüştür. Çözülen bu problemlerin sonuçları karşılaştırmalı olarak tablo 8, tablo 9, tablo 10, tablo 11, tablo 12 ve tablo 13'de sunulmuştur. Tablolar % sapma ve bu çalışmada elde edilen CPU zamanına göre hazırlanmıştır. Tablo 8, tablo 9, tablo 10, tablo 11, tablo 12 ve tablo 13'de yer alan CPU zamanı yapısı dakika: saniye.saniyenin 1000'de 1'i şeklindedir.

Tablo 8. P tipi 2 aşamalı Oğuz Kıyaslama Problemleri için elde edilen sonuçlar

Problem	% SAPMA			GA CPU
	Oğuz (2004)	Oğuz (2005)	GA	
P10S2T01	0,00000	0,00000	0,00000	00:00.016
P10S2T02	0,00000	0,00000	0,00000	00:00.000
P10S2T03	0,05960	0,05960	0,05960	00:00.031
P10S2T04	0,00000	0,00000	0,00000	00:00.015
P10S2T05	0,00000	0,00000	0,00000	00:00.015
P10S2T06	0,00000	0,00000	0,00000	00:00.000
P10S2T07	0,00000	0,00000	0,00000	00:00.015
P10S2T08	0,00000	0,00000	0,00000	00:00.015
P10S2T09	0,00000	0,00000	0,00000	00:00.015
P10S2T10	0,00000	0,00000	0,00000	00:00.000
P20S2T01	0,03994	0,01997	0,02611	00:07.703
P20S2T02	0,02368	0,02368	0,01639	00:11.671
P20S2T03	0,00188	0,00000	0,00000	00:00.281
P20S2T04	0,00000	0,00000	0,00000	00:00.562
P20S2T05	0,00000	0,00000	0,00000	00:00.218
P20S2T06	0,00000	0,00000	0,00000	00:02.500
P20S2T07	0,00568	0,00341	0,00341	00:05.391
P20S2T08	0,00556	0,00278	0,00278	00:02.500
P20S2T09	0,00000	0,00000	0,00000	00:00.469
P20S2T10	0,00000	0,00000	0,00000	00:01.328
P50S2T01	0,01753	0,01753	0,00605	01:35.958
P50S2T02	0,00000	0,00000	0,00000	00:01.031
P50S2T03	0,00590	0,00590	0,00000	01:25.375
P50S2T04	0,01238	0,01238	0,00681	01:58.938
P50S2T05	0,00235	0,00353	0,00235	01:41.781
P50S2T06	0,00000	0,00000	0,00000	00:01.843
P50S2T07	0,02929	0,02716	0,00692	03:06.016
P50S2T08	0,00000	0,00000	0,00000	00:02.000
P50S2T09	0,00000	0,00000	0,00000	00:01.047
P50S2T10	0,00262	0,00262	0,00052	00:14.952
P1HS2T01	0,00534	0,00534	0,00721	00:23.901
P1HS2T02	0,00000	0,00000	0,00000	00:00.219
P1HS2T03	0,00814	0,00814	0,00693	00:13.826
P1HS2T04	0,00000	0,00000	0,00000	00:03.061
P1HS2T05	0,00000	0,00000	0,00000	00:00.609
P1HS2T06	0,00000	0,00000	0,00000	00:00.391
P1HS2T07	0,01404	0,00926	0,00657	00:24.576
P1HS2T08	0,00020	0,00020	0,00000	00:00.219
P1HS2T09	0,00105	0,00105	0,00683	00:16.577
P1HS2T10	0,00097	0,00097	0,00242	00:17.652

Tablo 9. P tipi 5 aşamalı Oğuz Kıyaslama Problemleri için elde edilen sonuçlar

Problem	% SAPMA			GA CPU
	Oğuz (2004)	Oğuz (2005)	GA	
P10S5T01	0,04878	0,04530	0,04878	00:00.859
P10S5T02	0,06022	0,06022	0,06022	00:01.796
P10S5T03	0,08618	0,07540	0,07540	00:01.297
P10S5T04	0,08278	0,08609	0,11755	00:02.796
P10S5T05	0,03684	0,03684	0,03684	00:00.390
P10S5T06	0,00000	0,00000	0,00000	00:00.906
P10S5T07	0,10345	0,08210	0,08210	00:02.125
P10S5T08	0,05323	0,02742	0,02258	00:01.422
P10S5T09	0,09076	0,09076	0,10191	00:00.859
P10S5T10	0,10417	0,10417	0,12660	00:02.141
P20S5T01	0,01567	0,01567	0,01567	00:00.515
P20S5T02	0,07165	0,07165	0,07165	00:08.861
P20S5T03	0,04657	0,02561	0,03260	00:08.236
P20S5T04	0,10317	0,02268	0,00000	00:04.141
P20S5T05	0,02313	0,02313	0,01682	00:10.798
P20S5T06	0,02650	0,02650	0,01880	00:11.548
P20S5T07	0,00000	0,00000	0,00000	00:03.515
P20S5T08	0,03519	0,03519	0,03128	00:04.141
P20S5T09	0,00000	0,00000	0,00000	00:00.625
P20S5T10	0,07165	0,07165	0,06858	00:12.516
P50S5T01	0,03947	0,00911	0,00911	00:23.219
P50S5T02	0,05355	0,04539	0,01099	00:36.343
P50S5T03	0,05338	0,00998	0,00998	00:50.938
P50S5T04	0,10659	0,00618	0,02523	01:30.797
P50S5T05	0,02722	0,02577	0,00000	00:32.609
P50S5T06	0,00000	0,00000	0,00000	00:26.674
P50S5T07	0,01100	0,01100	0,00513	00:41.763
P50S5T08	0,05390	0,02447	0,01099	00:26.328
P50S5T09	0,07561	0,05096	0,00000	00:57.703
P50S5T10	0,04152	0,00271	0,00587	00:55.766
P1HS5T01	0,03493	0,03493	0,00000	01:12.234
P1HS5T02	0,01543	0,01543	0,00000	01:22.515
P1HS5T03	0,06990	0,03819	0,00674	04:24.603
P1HS5T04	0,01425	0,01425	0,01130	02:52.023
P1HS5T05	0,02347	0,02347	0,00000	00:44.922
P1HS5T06	0,17312	0,03591	0,02924	06:02.078
P1HS5T07	0,00780	0,00300	0,00000	00:41.563
P1HS5T08	0,02858	0,00673	0,00000	01:57.094
P1HS5T09	0,04264	0,01379	0,00000	01:33.297
P1HS5T10	0,03248	0,03248	0,02994	05:06.859

Tablo 10. P tipi 8 aşamalı Oğuz Kıyaslama Problemleri için elde edilen sonuçlar

Problem	% SAPMA			GA CPU
	Oğuz (2004)	Oğuz (2005)	GA	
P10S8T01	0,21268	0,21268	0,23662	00:10.157
P10S8T02	0,26179	0,26179	0,31870	00:02.344
P10S8T03	0,20162	0,20027	0,21786	00:01.469
P10S8T04	0,13091	0,13091	0,17350	00:01.938
P10S8T05	0,01902	0,01902	0,04212	00:01.109
P10S8T06	0,00409	0,00409	0,00409	00:00.796
P10S8T07	0,24757	0,24757	0,30097	00:01.454
P10S8T08	0,19479	0,19479	0,24540	00:00.782
P10S8T09	0,00830	0,00830	0,02075	00:01.141
P10S8T10	0,19589	0,19589	0,20000	00:01.953
P20S8T01	0,10280	0,08163	0,09599	00:17.453
P20S8T02	0,02344	0,00000	0,00000	00:04.547
P20S8T03	0,02573	0,02573	0,02744	00:45.999
P20S8T04	0,03363	0,01593	0,07080	00:24.078
P20S8T05	0,03152	0,03152	0,02300	00:35.016
P20S8T06	0,14361	0,08163	0,04913	01:05.594
P20S8T07	0,28308	0,23795	0,28821	00:23.406
P20S8T08	0,03412	0,03791	0,03791	00:07.297
P20S8T09	0,02344	0,00000	0,00000	00:06.156
P20S8T10	0,02573	0,04117	0,03945	00:09.000
P50S8T01	0,16810	0,02709	0,05288	03:53.755
P50S8T02	0,02750	0,02750	0,00000	04:03.593
P50S8T03	0,00936	0,00936	0,00787	05:27.433
P50S8T04	0,24979	0,04760	0,02696	04:13.578
P50S8T05	0,03639	0,03639	0,00986	03:52.078
P50S8T06	0,18438	0,01875	0,00313	04:11.019
P50S8T07	0,05436	0,05436	0,02544	02:51.125
P50S8T08	0,02434	0,02434	0,01881	03:32.422
P50S8T09	0,20177	0,04760	0,03412	04:01.833
P50S8T10	0,05371	0,05371	0,03187	03:55.359
P1HS8T01	0,07746	0,02877	0,00000	14:55.688
P1HS8T02	0,03568	0,03568	0,00157	12:48.611
P1HS8T03	0,04710	0,01987	0,00644	09:50.312
P1HS8T04	0,02149	0,02149	0,00103	18:57.406
P1HS8T05	0,04080	0,01944	0,00019	07:49.540
P1HS8T06	0,09604	0,03422	0,01104	13:42.918
P1HS8T07	0,06341	0,02426	0,00019	11:10.917
P1HS8T08	0,28169	0,07294	0,05584	19:08.864
P1HS8T09	0,08362	0,01951	0,00020	10:40.883
P1HS8T10	0,08600	0,03838	0,00185	06:43.914

Tablo 11. Q tipi 2 aşamalı Oğuz Kıyaslama Problemleri için elde edilen sonuçlar

Problem	% SAPMA			GA CPU
	Oğuz (2004)	Oğuz (2005)	GA	
Q10S2T1	0,00000	0,00000	0,00000	00:00.031
Q10S2T2	0,05012	0,05012	0,05012	00:00.016
Q10S2T3	0,09859	0,09859	0,00000	00:00.046
Q10S2T4	0,00000	0,00000	0,00000	00:00.015
Q10S2T5	0,00000	0,00000	0,19209	00:00.015
Q10S2T6	0,05817	0,05817	0,06094	00:00.016
Q10S2T7	0,00286	0,00286	0,00286	00:00.281
Q10S2T8	0,12333	0,12333	0,93333	00:00.016
Q10S2T9	0,04076	0,04076	0,04076	00:00.062
Q10S2T10	0,07331	0,07331	0,07331	00:00.016
Q20S2T1	0,07193	0,07193	0,07334	00:00.422
Q20S2T2	0,00892	0,00892	0,35414	00:01.953
Q20S2T3	0,04690	0,04690	0,47280	00:01.172
Q20S2T4	0,00000	0,00000	0,00000	00:00.172
Q20S2T5	0,00844	0,00844	0,01547	00:02.578
Q20S2T6	0,00426	0,00426	0,01420	00:01.234
Q20S2T7	0,00000	0,00000	0,00000	00:00.203
Q20S2T8	0,00493	0,00369	0,00369	00:00.156
Q20S2T9	0,00000	0,00000	0,00000	00:00.187
Q20S2T10	0,01826	0,01826	0,02283	00:01.625
Q50S2T1	0,00161	0,00161	0,00161	00:02.422
Q50S2T2	0,00329	0,00329	0,00329	00:22.719
Q50S2T3	0,01651	0,01651	0,04068	01:38.328
Q50S2T4	0,00623	0,00623	0,00249	01:16.047
Q50S2T5	0,00000	0,00000	0,00000	00:11.358
Q50S2T6	0,03114	0,02353	0,22215	03:07.343
Q50S2T7	0,00000	0,00000	0,00000	00:02.485
Q50S2T8	0,04037	0,03292	0,07267	02:26.169
Q50S2T9	0,02747	0,02747	0,02129	01:33.576
Q50S2T10	0,03338	0,02567	0,10334	01:03.782
Q1HS2T1	0,02536	0,02250	0,02214	01:26.320
Q1HS2T2	0,01904	0,01904	0,01360	01:58.655
Q1HS2T3	0,07560	0,03849	0,04364	01:51.906
Q1HS2T4	0,00000	0,00000	0,05171	02:16.049
Q1HS2T5	0,00173	0,00029	0,00029	00:24.046
Q1HS2T6	0,01704	0,00911	0,00705	02:16.500
Q1HS2T7	0,02439	0,01891	0,02165	02:33.969
Q1HS2T8	0,00266	0,00118	0,00118	01:56.125
Q1HS2T9	0,01737	0,01642	0,00000	01:55.523
Q1HS2T10	0,02134	0,01885	0,01778	01:32.966

Tablo 12. Q tipi 5 aşamalı Oğuz Kıyaslama Problemleri için elde edilen sonuçlar

Problem	% SAPMA			GA CPU
	Oğuz (2004)	Oğuz (2005)	GA	
Q10S5T1	0,10642	0,09291	0,09291	00:11.953
Q10S5T2	0,01536	0,01536	0,02560	00:06.828
Q10S5T3	0,05664	0,05664	0,09027	00:04.219
Q10S5T4	0,07743	0,06919	0,14827	00:06.578
Q10S5T5	0,04412	0,02574	0,02574	00:00.844
Q10S5T6	0,08155	0,08155	0,10485	00:01.265
Q10S5T7	0,15146	0,15146	0,15146	00:02.171
Q10S5T8	0,03811	0,03811	0,00000	00:00.688
Q10S5T9	0,01637	0,00000	0,00000	00:00.828
Q10S5T10	0,13163	0,13163	0,13163	00:07.640
Q20S5T1	0,04318	0,03533	0,02257	00:16.313
Q20S5T2	0,07970	0,07970	0,07846	00:20.406
Q20S5T3	0,15047	0,13626	0,12915	00:24.688
Q20S5T4	0,04277	0,04277	0,09017	01:00.890
Q20S5T5	0,13907	0,13907	0,00000	00:45.219
Q20S5T6	0,11860	0,07093	0,12093	00:09.234
Q20S5T7	0,04580	0,04580	0,04580	00:07.188
Q20S5T8	0,11832	0,11578	0,07761	00:18.452
Q20S5T9	0,04061	0,01647	0,01207	02:24.266
Q20S5T10	0,05345	0,05345	0,04120	00:58.188
Q50S5T1	0,18566	0,15558	0,11163	01:43.483
Q50S5T2	0,09406	0,09406	0,08567	01:27.156
Q50S5T3	0,14721	0,10042	0,00000	01:24.156
Q50S5T4	0,12531	0,11830	0,00000	01:29.015
Q50S5T5	0,19359	0,17784	0,13644	01:35.093
Q50S5T6	0,12627	0,12231	0,09966	03:59.625
Q50S5T7	0,12966	0,08294	0,01417	01:27.374
Q50S5T8	0,11680	0,11680	0,00410	01:03.344
Q50S5T9	0,02479	0,02479	0,00000	01:33.329
Q50S5T10	0,02723	0,02723	0,00050	01:38.499
Q1HS5T1	0,26484	0,18948	0,07813	06:49.688
Q1HS5T2	0,22851	0,15451	0,00000	14:31.960
Q1HS5T3	0,16207	0,16207	0,00227	07:16.062
Q1HS5T4	0,18948	0,18948	0,08213	06:52.063
Q1HS5T5	0,39331	0,24866	0,10729	07:03.765
Q1HS5T6	0,17349	0,17349	0,04855	08:11.296
Q1HS5T7	0,43335	0,24796	0,07974	08:02.063
Q1HS5T8	0,34741	0,18322	0,02657	06:09.466
Q1HS5T9	0,29044	0,21104	0,00443	06:53.453
Q1HS5T10	0,26555	0,22710	0,06609	06:30.188

Tablo 13. Q tipi 8 aşamalı Oğuz Kıyaslama Problemleri için elde edilen sonuçlar

Problem	% SAPMA			GA CPU
	Oğuz (2004)	Oğuz (2005)	GA	
Q10S8T1	0,15860	0,15860	0,25134	00:06.141
Q10S8T2	0,07570	0,07570	0,02789	00:01.813
Q10S8T3	0,08005	0,08005	0,07349	00:01.656
Q10S8T4	0,02289	0,02289	0,02169	00:03.125
Q10S8T5	0,23592	0,19635	0,20700	00:00.688
Q10S8T6	0,12278	0,13018	0,17456	00:00.641
Q10S8T7	0,14944	0,14944	0,32432	00:02.078
Q10S8T8	0,13580	0,13580	0,14938	00:00.781
Q10S8T9	0,37049	0,18053	0,19937	00:01.734
Q10S8T10	0,07261	0,07261	0,04331	00:04.266
Q20S8T1	0,11168	0,11168	0,11578	00:09.656
Q20S8T2	0,18653	0,18653	0,24283	00:09.812
Q20S8T3	0,10169	0,10169	0,07439	00:15.454
Q20S8T4	0,10758	0,09436	0,08201	00:21.234
Q20S8T5	0,18469	0,18469	0,21327	00:20.000
Q20S8T6	0,11168	0,11168	0,10041	00:14.734
Q20S8T7	0,45445	0,40285	0,21625	00:39.437
Q20S8T8	0,21613	0,21613	0,22151	00:29.328
Q20S8T9	0,20128	0,20128	0,21949	00:36.982
Q20S8T10	0,14924	0,14924	0,13807	00:09.860
Q50S8T1	0,20491	0,20491	0,18705	04:47.969
Q50S8T2	0,21466	0,14607	0,14921	05:25.906
Q50S8T3	0,16124	0,09231	0,00585	05:20.250
Q50S8T4	0,24140	0,24140	0,19677	04:33.016
Q50S8T5	0,22415	0,22415	0,11916	02:19.346
Q50S8T6	0,13731	0,13731	0,10985	05:53.118
Q50S8T7	0,16048	0,09445	0,00688	05:34.547
Q50S8T8	0,24389	0,24389	0,32048	04:37.625
Q50S8T9	0,41714	0,14828	0,12827	04:37.625
Q50S8T10	0,22949	0,15652	0,07851	04:34.610
Q1HS8T1	0,27763	0,10581	0,06784	21:26.602
Q1HS8T2	0,22015	0,08463	0,02414	20:50.500
Q1HS8T3	0,33563	0,14856	0,09711	22:13.250
Q1HS8T4	0,47247	0,16203	0,07787	22:23.812
Q1HS8T5	0,43682	0,15998	0,09870	23:13.375
Q1HS8T6	0,51218	0,16969	0,13994	23:28.782
Q1HS8T7	0,21159	0,15692	0,08967	16:55.669
Q1HS8T8	0,48492	0,15458	0,15197	22:33.253
Q1HS8T9	0,11234	0,11234	0,00000	16:06.094
Q1HS8T10	0,21362	0,09151	0,05693	21:17.266

P10S2 tipi Oğuz kıyaslama problemlerinden 9 tanesi çözümleri optimal sonuca ulaşmıştır. P10S2T03 problemi optimal sonuca ulaşmamasına rağmen Oğuz'un çalışmaları ile aynı değer bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden dokuz tanesi optimal sonuca ulaşmıştır.

P20S2 tipi Oğuz kıyaslama problemlerinden 6 tanesinde optimal sonuca ulaşılmıştır. P20S2T02 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Problemlerden 2 tanesinde ise optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçları ile aynı değerler bulunmuştur. P20S2T01 problem çözümü Oğuz'un ilk çalışmasından daha iyi olmasına rağmen ikinci çalışmasından daha kötü bir sonuç bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden sadece altı tanesi optimal sonuca ulaşmıştır.

P50S2 tipi Oğuz kıyaslama problemlerinden 5 tanesinde optimal sonuca ulaşılmıştır. 4 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Problemlerden 1 tanesinde ise optimal sonuca ulaşılamamasına rağmen Oğuz'un sonucu ile aynı değer bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden sadece dört tanesi optimal sonuca ulaşmıştır.

P1HS2 tipi Oğuz kıyaslama problemlerinden 5 tanesinde optimal sonuca ulaşılmıştır. 2 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. 3 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü bir değer bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden sadece dört tanesi optimal sonuca ulaşmıştır.

P10S5 tipi Oğuz kıyaslama problemlerinden P10S5T06 problem çözümünde optimal sonuca ulaşılmıştır. P10S5T08 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur.

Problemlerden 4 tanesinde ise optimal sonuçlara ulaşılamamasına rağmen Oğuz'un sonucu ile aynı değerler bulunmuştur. 4 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü bir değer bulunmuştur. Bunlardan sadece P10S5T01 problem çözümü Oğuz'un ilk çalışması ile aynı değer olmasına rağmen ikinci çalışmasından daha kötü bir sonuç bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden sadece bir tanesi optimal sonuca ulaşmıştır.

P20S5 tipi Oğuz kıyaslama problemlerinden 3 problem çözümünde optimal sonuca ulaşılmıştır. 4 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Problemlerden 2 tanesinde ise optimal sonuçlara ulaşılamamasına rağmen Oğuz'un sonucu ile aynı değerler bulunmuştur. P20S5T03 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü bir değer bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden sadece iki tanesi optimal sonuca ulaşmıştır.

P50S5 tipi Oğuz kıyaslama problemlerinden 3 problem çözümünde optimal sonuca ulaşılmıştır. 4 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Problemlerden 2 tanesinde ise optimal sonuçlara ulaşılamamasına rağmen Oğuz'un sonucu ile aynı değerler bulunmuştur. P20S5T03 problem çözümü Oğuz'un ilk çalışmasından daha iyi olmasına rağmen ikinci çalışmasından daha kötü bir sonuç bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden sadece bir tanesi optimal sonuca ulaşmıştır.

P1HS5 tipi Oğuz kıyaslama problemlerinden 3 problem çözümünde optimal sonuca ulaşılmıştır. 3 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Problemlerden 2 tanesinde ise optimal sonuçlara ulaşılamamasına rağmen Oğuz'un sonucu ile aynı değerler bulunmuştur. 2

problem çözümleri Oğuz'un ilk çalışmasından daha iyi olmasına rağmen ikinci çalışmasından daha kötü değerler bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

P10S8 tipi Oğuz kıyaslama problemlerinden sadece P10S5T06 problem çözümünde optimal sonuca ulaşılmamış ancak Oğuz'un sonuçları ile aynı değerler bulunmuştur. 9 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü değerler bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

P20S8 tipi Oğuz kıyaslama problemlerinden 2 problem çözümünde optimal sonuca ulaşılmıştır. 3 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Problemlerden 1 tanesinde ise optimal sonuçlara ulaşılamamasına rağmen Oğuz'un sonucu ile aynı değerler bulunmuştur. 4 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü değerler bulunmuştur. Bunlardan P20S8T01 problem çözümü Oğuz'un ilk çalışmasından daha iyi olmasına rağmen ikinci çalışmasından daha kötü değer bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden sadece iki tanesi optimal sonuca ulaşmıştır.

P50S8 tipi Oğuz kıyaslama problemlerinden sadece P50S8T02 problem çözümünde optimal sonuca ulaşılmıştır. 8 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. P50S8T02 problem çözümü Oğuz'un ilk çalışmasından daha iyi olmasına rağmen ikinci çalışmasından daha kötü bir sonuç bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

P1HS8 tipi Oğuz kıyaslama problemlerinden 1 problem çözümünde optimal sonuca ulaşılmıştır. 9 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un

sonuçlarından daha iyi bir sonuç bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

Q10S2 tipi Oğuz kıyaslama problemlerinden 3 tanesi çözümleri optimal sonuca ulaşmıştır. 4 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un çalışmaları ile aynı değerler bulunmuştur. 3 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü değerler bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden sadece iki tanesi optimal sonuca ulaşmıştır.

Q20S2 tipi Oğuz kıyaslama problemlerinden 3 tanesinde optimal sonuca ulaşılmıştır. Problemlerden 2 tanesinde ise optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçları ile aynı değerler bulunmuştur. 3 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü değerler bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden sadece üç tanesi optimal sonuca ulaşmıştır.

Q50S2 tipi Oğuz kıyaslama problemlerinden 2 tanesinde optimal sonuca ulaşılmıştır. 2 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Problemlerden 2 tanesinde ise optimal sonuca ulaşılamamasına rağmen Oğuz'un sonucu ile aynı değer bulunmuştur. 4 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü değerler bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden sadece iki tanesi optimal sonuca ulaşmıştır.

Q1HS2 tipi Oğuz kıyaslama problemlerinden 1 tanesinde optimal sonuca ulaşılmıştır. 4 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Problemlerden 2 tanesinde ise optimal sonuca ulaşılamamasına rağmen Oğuz'un sonucu ile aynı değer bulunmuştur. 3 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü bir değer

bulunmuştur. Bunlardan sadece Q1HS2T07 problem çözümü Oğuz'un ilk çalışmasından daha iyi olmasına rağmen ikinci çalışmasından daha kötü bir sonuç bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

Q10S5 tipi Oğuz kıyaslama problemlerinden 2 tanesinde optimal sonuca ulaşılmıştır. Problemlerden 4 tanesinde ise optimal sonuçlara ulaşılamamasına rağmen Oğuz'un sonucu ile aynı değerler bulunmuştur. 4 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü bir değer bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

Q20S5 tipi Oğuz kıyaslama problemlerinden 1 problem çözümünde optimal sonuca ulaşılmıştır. 6 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Problemlerden 1 tanesinde ise optimal sonuçlara ulaşılamamasına rağmen Oğuz'un sonucu ile aynı değerler bulunmuştur. 2 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü değerler bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

Q50S5 tipi Oğuz kıyaslama problemlerinden 3 problem çözümünde optimal sonuca ulaşılmıştır. 7 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

Q1HS5 tipi Oğuz kıyaslama problemlerinden Q1HS5T02 problem çözümünde optimal sonuca ulaşılmıştır. 9 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

Q10S8 tipi Oğuz kıyaslama problemlerinden hiçbiri optimal sonuca ulaşamamıştır. Ancak 4 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. 6 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü değerler bulunmuştur. Bunlardan 2 problem çözümü Oğuz'un ilk çalışmasından daha iyi olmasına rağmen ikinci çalışmasından daha kötü değerler bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

Q20S8 tipi Oğuz kıyaslama problemlerinden hiçbiri optimal sonuca ulaşamamıştır. 5 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. 5 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü bir değer bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

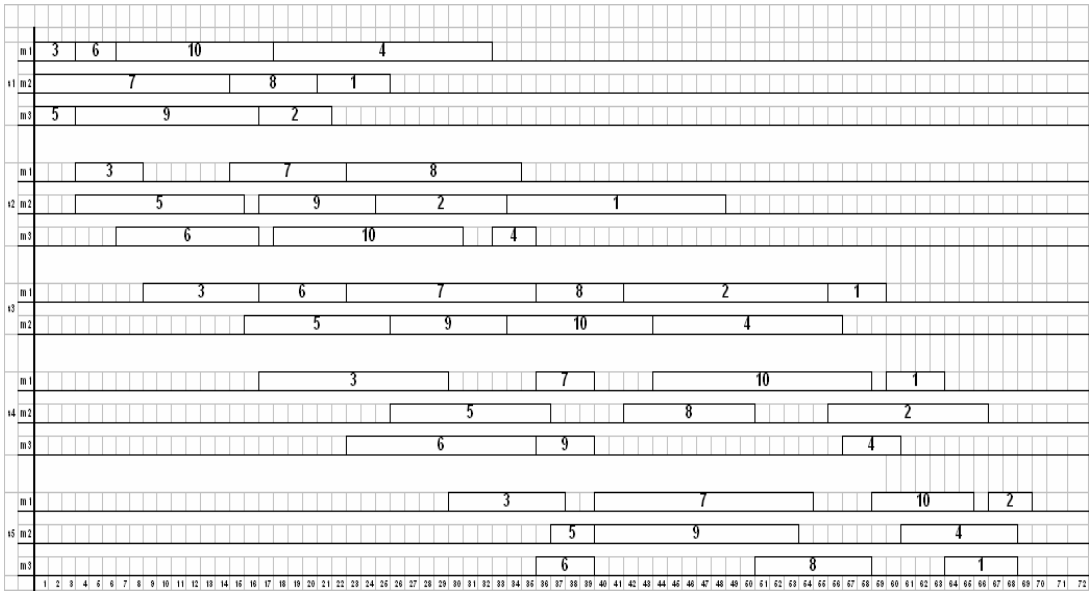
Q50S8 tipi Oğuz kıyaslama problemlerinden hiçbiri optimal sonuca ulaşamamıştır. 8 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. 2 problem çözümünde optimal sonuca ulaşılamadığı gibi Oğuz'un çözümlerinden daha kötü bir değerler bulunmuştur. Bunlardan 1 problem çözümü Oğuz'un ilk çalışmasından daha iyi olmasına rağmen ikinci çalışmasından daha kötü değer bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

Q1HS8 tipi Oğuz kıyaslama problemlerinden Q1HS8T09 problem çözümünde optimal sonuca ulaşılmıştır. 9 problem çözümünde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi bir sonuç bulunmuştur. Ayrıca bu gruptaki Oğuz'un çözümlerinden hiçbiri optimal sonuçları bulamamıştır.

Oğuz'un (2004) kıyaslama problemlerinden toplam 240 tane problem üzerine çalışılmıştır. Bu çalışmada toplam 60 tane çözümde optimal sonuca ulaşılmasına rağmen Oğuz her iki çalışmasında toplam 36 tanesinde optimal sonuca ulaşmıştır. 93 tanesinde optimal sonuca ulaşılamamasına rağmen Oğuz'un sonuçlarından daha iyi sonuçlar bulunmuştur. 31 tane çözümde Oğuz ile aynı sonuçlar bulunmuştur. 46 tanesinde Oğuz'dan daha kötü sonuçlar bulunmuştur. 10 tane çözümde ise Oğuz'un birinci çalışmasından daha iyi ancak ikinci çalışmasından daha kötü sonuçlar bulunmuştur.

Problemlerde yer alan işlem süreleri rassal olarak atandığından, işler bazında farklı aşamalardaki işlem süreleri arasında bir ilişkiden söz etmek mümkün olmaz. Buda problemin çözümü esnasında Gant'a işleri yerleştirirken aşamanın birinde iyi performans vermesine rağmen başka bir aşamada kötü sonuçlar alınmasına neden olabilmektedir. GA' nın temeli olan iyi bireyden iyi bir birey türemesi, problemlerin bu özelliğinden dolayı etkinliğini gösterememiş olabilir.

Şekil 14'te Neron ve Carlier problemlerinden j10c5c1 problemi çözümü sırasında işlerin Gant'a yerleştirilmiş halidir.



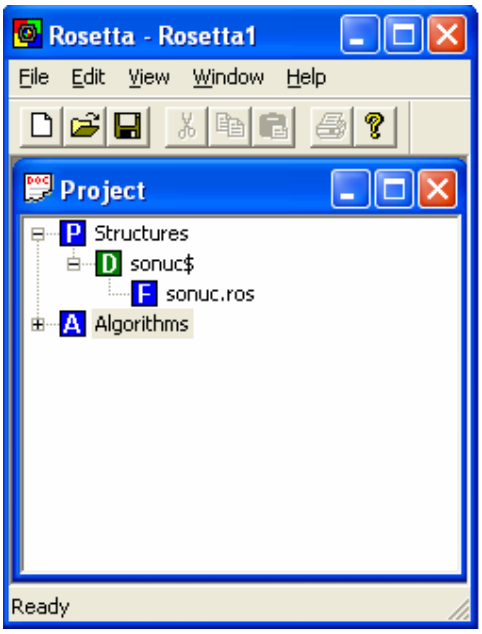
Şekil 14. j10c5c1 Neron ve Carlier problemleri Gant Şeması

7.2. Veri Madenciliği Uygulaması

Madenciliğin amacı bir operasyonun karakteristiği ile onun GA çözümündeki yeri arasındaki ilişkiyi belirlemektir. Genetik algoritmalarla oluşturulan C_{max} kriterini optimize eden iş sıraları, öğrenme ve karar kuralları üretmede kullanılacaktır.

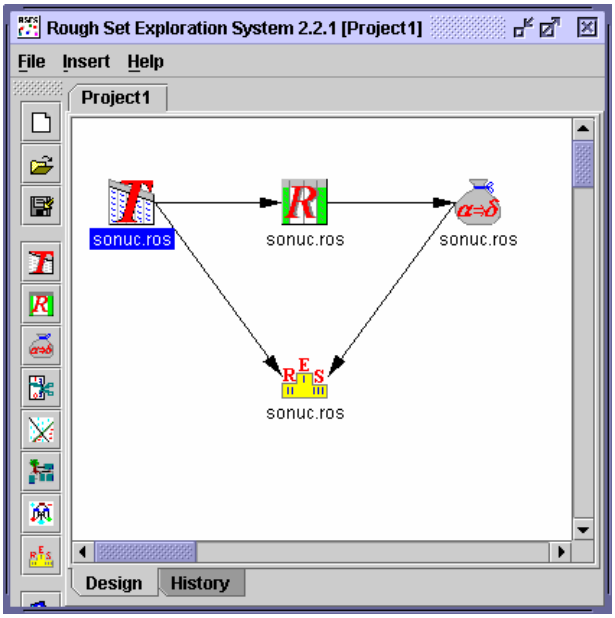
Veri madenciliği ile bilgi çıkarımı yapabilmek için verilerin hazırlanması gerekmektedir. Veri madenciliği için gerekli bilgi için genetik algoritmaları uygulayarak elde ettiğimiz iş sıraları ve C_{max} değerleri kullanılmıştır. İşler üç bölüme ayrılmıştır. Her bölümdeki işlerin ortalama süreleri hesaplanmıştır. Hesaplanan ortalamalar karşılaştırılıp, en küçük olana 1, sonrakine 2 ve en büyüğüne 3 değerini atanmıştır. Bunu yapmaktaki amaç işlere zorluk derecesine göre bir öncelik vermektir. Ayrıca elde edilen kromozom dizilişleri dikkate alınarak C_{max} değerleri en küçük olana 1, %3 lük fazlalığı olanlara 2, %5 fazlalığı olanlara 3 değerini verilmiştir.

Veri madenciliği için verilerin uygun forma transfer edilmesi gerekmektedir. Bu amaçla sınıflandırma işleminden sonra, verilerin veri madenciliği uygulamasına uygun hale getirilmesi için ROSETTA programı kullanılmıştır. Programa ait ekran görüntüsü şekil 15'te gösterilmiştir.



Şekil 15. ROSETTA programı ekran görüntüsü

Veri madenciliği, ilginç ilişkilerin ayrılması bağıntı ve kuralların bilgisayar programları kullanarak aranmasıdır. Bu nedenle; veri madenciliğine uygun hale getirilen veriler, veri madenciliği programı olan RSES2.2.1'de kullanılmıştır. Program çalıştırılırken veri madenciliği tekniklerinden genetik algoritmalar kullanılmış ve model olarak sınıflandırma seçilmiştir. RSES2.2.1 programına ait ekran görüntüleri şekil 16'te verilmiştir.



Şekil 16. RSES2.2.1 programına ait ekran görüntüsü

7.2.1. Kıyaslama Problemleri Analizi

Verinin iç yapısındaki ilişkileri ortaya çıkarmak, kümelenmeleri ve bu kümelerin yapılarını bulmak, varolan verilerden yola çıkarak çeşitli öngörülerde bulunmak, kısaca verinin iç yapısını çözmek için kıyaslama problemleri, problem türüne göre analiz edilmiştir. Analiz sonucunda her tip problem türüne uygun çeşitli kurallar elde edilmiştir. Bu kurallar genetik algoritmalarla elde ettiğimiz çözümlere göre oluşmuştur. Bu kurallardan C_{max} minimize eden ve en çok tekrarlanan kural aradığımız optimum sonucu vermektedir.

7.2.1.1. Neron Ve Carlier Problemleri Analizi

Neron ve Carlier problemleri 4 grubta analiz edilmiştir. Her grup için ayrı ayrı kurallar oluşturulmuştur.

- 10 x 5 tipi problemler kıyaslama problemleri :

  Toplanan verilerden, bu problem tipine uygun 19 tane kural oluşturulmuştur. Bu probleme ait analiz sonuçları ek 3'te verilmiştir. Analiz sonucu öğrenilen kurallardan amacımıza uygun olanı 14 nolu kuraldır ve 193 kere tekrarlanmıştır. Sıralama en en kısa ortalamalı iş grubu, uzun ortalamalı iş grubun, en kısa ortamalı iş grubuna doğrudur.

- 10 x 10 tipi problemler kıyaslama problemleri :

  Toplanan verilerden, bu problem tipine uygun 8 tane kural oluşturulmuştur. Bu probleme ait analiz sonuçları ek 3'te verilmiştir. Analiz sonucu öğrenilen kurallardan amacımıza uygun olanı 5 nolu kuraldır ve 92 kere tekrarlanmıştır . Sıralama en kısa ortalamalı iş grubundan uzun süreliye doğrudur.

- 15 x 5 tipi problemler kıyaslama problemleri :

  Toplanan verilerden, bu problem tipine uygun 19 tane kural oluşturulmuştur. Bu probleme ait analiz sonuçları ek 3'te verilmiştir. Analiz sonucu öğrenilen kurallardan amacımıza uygun olanı 14 nolu kuraldır ve 87 kere tekrarlanmıştır . Sıralama en kısa ortalamalı iş grubundan uzun süreliye doğrudur.

- 15 x10 tipi problemler kıyaslama problemleri :

  Toplanan verilerden, bu problem tipine uygun 6 tane kural oluşturulmuştur. Bu probleme ait analiz sonuçları ek 3'te verilmiştir. Analiz sonucu öğrenilen kurallardan

amacımıza uygun olanı 1 nolu kuraldır ve 61 kere tekrarlanmıştır. Sıralama en uzun ortalamalı iş grubundan en kısa süreliye doğrudur.

7.2.1.2.Oğuz Problemleri Analizi

Oğuz problemleri 3 grupta incelenmiştir. Her grup için ayrı ayrı kurallar oluşturulmuştur.

- 2 aşamalı kıyaslama problemleri :

  Toplanan verilerden, bu problem tipine uygun 18 tane kural oluşturulmuştur. Bu probleme ait analiz sonuçları ek 3'te verilmiştir. Analiz sonucu öğrenilen kurallardan amacımıza uygun olanı 2, 8 ve 14 nolu kurallardır ve 2433 kere tekrarlanmışlardır. Sıralama en uzun ortalamalı iş grubundan en kısa süreliye doğrudur.
- 5 aşamalı kıyaslama problemleri :

  Toplanan verilerden, bu problem tipine uygun 18 tane kural oluşturulmuştur. Bu probleme ait analiz sonuçları ek 3'te verilmiştir. Analiz sonucu öğrenilen kurallardan amacımıza uygun olanı 6, 12 ve 18 nolu kurallardır ve 594 kere tekrarlanmışlardır . Sıralama en uzun ortalamalı iş grubu,sonra en kısa daha sonra kısa süreliye doğrudur.
- 8 aşamalı kıyaslama problemleri :

  Toplanan verilerden, bu problem tipine uygun 18 tane kural oluşturulmuştur. Bu probleme ait analiz sonuçları ek 3'te verilmiştir. Analiz sonucu öğrenilen kurallardan amacımıza uygun olanı 4, 10 ve 16 nolu kurallardır ve 4232 kere tekrarlanmışlardır. Sıralama en uzun ortalamalı iş grubundan en kısa süreliye doğrudur.

8. SONUÇ

Bu çalışma; anlamlı kurallar ortaya çıkarmak amacıyla, çeşitli yöntemler kullanarak büyük miktarlardaki veriyi analiz etme olarak tanımlanabilir. Varolan verilerden yola çıkarak, veriler arasındaki gerçek bağlantıları belirleyip çeşitli öngörülerde bulunmaktır.

Problemlerin çözümü için bir genetik algoritma geliştirilmiştir. Geliştirilen genetik algoritmalar ile literatürde bulunan esnek akış tipi kıyaslama problemleri çözülmüştür. Çalışılan problemler NP-zor tipte problemlerdir. Bu problemleri optimal şekilde çözmek için geliştirilmiş bir yöntem bulunmamakta, daha çok sezgisel yöntemler kullanılmaktadır.

Genetik algoritmaların çözüm performanslarının geliştirilmesi amacıyla parametre optimizasyonu çalışması yapılmıştır. Genetik operatörler ayrıntılarıyla incelenmiş ve birbirleriyle karşılaştırılmış, araştırmalar ve denemeler sonucunda en uygun parametre seti belirlenmiştir. Önerilen algoritmalar; her problem için bulunan en iyi parametrelerle çalıştırılmıştır.

Genetik algoritmaların çalıştırılması sonucu elde edilen Neron ve Carlier kıyaslama problemleri sonuçları, Döyen (2004) tarafından Yapay Bağışıklık Sistemleri (YBS) ile elde edilen sonuçlar ile karşılaştırılmıştır. Her ne kadar elde ettiğimiz sonuçlar YBS'ye göre çok iyi sonuçlar veremesede kısa sürede sonuca ulaşmıştır.

Genetik algoritmaların çalıştırılması sonucu elde edilen Oğuz kıyaslama problemleri sonuçları ise Oğuz'un 2004 yılında ve 2005 yılında yapmış olduğu çalışmalarla karşılaştırılmıştır. Elde edilen sonuçlar Oğuz'un sonuçlarına göre daha iyi değerler vermiştir.

Elde edilen sonuçlar veri madenciliğine uygun hale getirilip, analiz edilmiştir.Analiz sonucunda problem türüne göre çeşitli kurallar çıkmıştır. Bu kurallar C_{max} ve optimum iş

sıraları göz önüne alınarak belirlenmiştir. Kurallar incelendiğinde, birden fazla kural çıkma sebebi olarak çok sayıda optimum veya optimuma yakın (aynı C_{max} değerine sahip fakat iş sırası farklı olan küme) çözümler gösterilebilinir.

Esnek akış tipi çizelgeleme problemleri iş sıralarının eğilimlerinin nasıl olduğu görülebilmektedir. Önceden bilinmeyen ancak potansiyel olarak kullanışlı olan bu kurallar bu kurallar birbirleriyle kıyaslanarak elde edilecek sonuçlar; önerilen algoritmaların başka tekniklerle hibritlenerek daha optimum sonuçlar veren yeni algoritmalar oluşturulmasında etkili olacaktır.

9. KAYNAKÇA

1. AKBAŞ, S., OĞUZ, M., "Genetik Algoritma", http://www.yeniweb.net/genetik_algoritma.asp, 2001
2. AKPINAR, H., 2000, Veri Tabanlarında Bilgi Keşfi ve Veri Madenciliği. İÜ. İşletme Fakülte Dergisi, C29, S:17 Nisan 2000, S: 1-22. www.işletme.istanbul.edu.tr/akpinar
3. ALLAOUI, H., ARTIBA, A., 2006, Scheduling two-stage hybrid flow shop with availability constraints, Computers & Operations Research, 33, 1399-1419
4. ALPAYDIN, E., 2000, Zeki Veri Madenciliği. Bilişim 2000 Eğitim Semimeri Notları. www.cmpe.boun.edu.tr/~ethem
5. ARTHANARİ,T.S., RAMAMURTHY, K.G., 1971. An Extension of Two Machines Sequencing Problem, Opsearch, 8, 10-22.
6. AUSTIN, FRANCES, 1991, The Clift Family Correspondence 1792–1846, Sheffield: CECTAL.
7. AZİZOĞLU, M., ÇAKMAK, E., KONDAKÇI, S., 2001, A flexible flowshop problem with total flow time minimization, European Journal of Operational Research, 132, 528-538
8. BASKAK, M., EROL, V., Sipariş Tipi Atölyelerde İş Sıralama Problemi İçin Bir Genetik Algoritma Uygulaması, Yöneylem Araştırması/Endüstri Mühendisliği Kongresi / Çukurova Üniversitesi/Adana 1716/2004, Bildiriler Kitabı, S187-189.
9. BELMONT, MORENO, E., Nov 2001, The Role of Mutation and Population Size in Genetic Algorithms Applied to Physics Problems, International Journal of Modern Physics C: Physics & Computers, Vol. 12 Issue 9, p1345, 11p

10. BOLAT,B., EROL,K.O., IMRAK,C.E.,2004, Genetıc Algorıthms In Engıneerıng Applıcatıons And The Functıon Of Operators, Mühendislik ve Fen Bilimleri Dergisi, 264-271

11. BRAH, S.A., HUNSUCKER,J.L., 1991, Branch and Bound Algorithm for The Flow Shop With Multiple Processors, European Journal of Operational Research, 51, 88-99.

12. BRAH,S.A, LOO, L.L., 1999, Heuristics For Scheduling in A Flow Shop With Multiple Processors, European Journal of Operational Research, 113, 113-122.

13. CARLIER, J., NERON, E., 2000, An exact method for solving the multiprocessor flowshop, R.A.I.R.O-R.O., 34, 1-25.

14. CHEN, C.L., NEPPALLI, R.V., ALJABER,N., 1996, Genetic Algorithms Applied to the Continuous Flow Shop Problem, Computers and Industrial Engineering, vol 30, no:4, pp:919-929

15. CHEN, C.L., VEMPATİ, V.S., ALJABER, N., 1995, An application of genetic algorithms for flowshop problems, European Journal of Operational Research 80, 389-396.

16. CHENG, R., GEN, M., TSUJIMURAY, Y., 1999**,** A Tutorial Survey of Job Shop Scheduling Problems Using Genetic Algorithms, Part II: Hybrid Genetic Search Strategies, Computers and Industrial Engineering 36, 343-364.

17. CHUNG,Y.L., VAİRAKTARAKİS, G.L.,1994**,** Minimizing makespan in hybrid flowshops, Operations Research Letters

18. CIOCOIU, M.M., 2001, Applications of genetic algotihms in data mining. Yayınlanmış Doktora tezi, UMI

19. COWLING, P., JOHANSSON, M., 2002. Using real time information for effective dynamic scheduling. European Journal of Operational Research, 139, 230-244.

20. ÇERKEZ, H.S., 2003, Müşteri İlişkileri Yönetiminde İş Zekası Ve Veri Madenciliği Yöntemleri.Yayınlanmamış Yıl İçi Projesi, İTÜ, İstanbul.

21. ÇÖLESEN, R., 2002, Veri Yapıları ve Algoritmalar, Papatya Yayıncılık, Istanbul.

22. DAIN, R. A., 1997, Developing Mobile Robot Wall-Following Algorithms Using Genetic Programing, Applied Intelligence, Morgan Kaufmann, USA

23. DİNG,F.Y., KİTTİCHARTPHAYAK,D., 1994, Heuristics for scheduling flexible flow lines, Computers and Industrial Engineering, 26, 27-34.

24. DÖYEN, A., 2004, Akış Tipi Çizelgeleme Problemlerinin Yapay Bağışıklık Sistemleri İle Çözümü ve Parametre Optimizasyonu, Yüksek Lisans Tezi, Selçuk Üniversitesi

25. EMEL, G., TAŞKIN, Ç., 2002, Genetik Algoritmalar Ve Uygulama Alanları, Uludağ Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, Cilt XXI, Sayı 1, Bursa, ss. 129–152.

26. ENGİN, O., 2001, Akış Tipi Çizelgeleme Problemlerinin Genetik Algoritma ile Çözümün Performansının Artırılmasında Parametre Optimizasyonu, Doktora Tezi, İ.T.Ü. Fen Bilimleri Enstitüsü, İstanbul.

27. ENGİN, O., FIĞLALI, A., 2002, Akış Tipi Çizelgeleme Problemlerinin Genetik Algoritma ile Çözüm Performansının Artırılmasında Deney Tasarımı Uygulaması, http://www.mmo.org.tr/endustrimuhendisligi/2002_3/akistipi_cizelgeleme_makale.htm, Eylül 2002 sayı 3

28. ENGİN, O., FIĞLALI, A., 2002. Akış Tipi Çizelgeleme Problemlerinin Genetik Algoritma Yardımı İle Çözümünde Uygun Çaprazlama Operatörlerinin Belirlenmesi, Doğuş Üniversitesi Dergisi, 2002/6.27-35

29. EREN, T., GÜNER, E., 2002, Tek ve Paralel Makinalı Problemlerde Çok Ölçütlü Çizelgeleme Problemleri İçin Bir Literatür Taraması, Gazi Üniv. Müh. Mim. Fak. Der. Cilt 17, No: 4, 37-69

30. GABER, M.M. (2002). A framework for a scalable distributed data mining model. Yayınlanmış Doktora Tezi, UMI

31. GOLDBERG, D., E., 1989, Genetic Algorithms in Search Optimization And Machine Learning, Addison Wesley Publishing Company U.S.A.

32. GOLDBERG, D., E., DEB, K., 1991, A comparison of selection schemes used in genetic algorithms, Foundations of Genetic Algorithms, edited by G. J. E. Rawlins, pp. 69–93

33. GUPTA, J.N.D., 1971. A functional heuristic algorithm for the n-job, m-machine flow shop problem, Operational Research Quartely , 22 , 39-47.

34. GUPTA, J.N.D., 1988. Two-stage hybrid flowshop scheduling problem, Operational Research Society, 39, 359-364.

35. GUPTA,J.N.D., HARIRI, A.M.A., POTTS, C.N.,1997. Scheduling a two stage hybrid flow shop with paralel machines at the first stage. J. Ann.Oper.Res.,69, 171-191.

36. GUPTA,J.N.D., TUNÇ,E.A.,1991. Schedules for a two stage hybrid flowshop with paralel machines at the second stage, International Journal of Production Research, 29, 1489-1502.

37. HARRATH, Y., MORELLO, B., ZERHOUNI, N., 2001, A genetic algorithm and data mining to resolve a job shop Schedule. IEEE, 727-728

38. HARRATH, Y., MORELLO, B., ZERHOUNİ, N., 2002, A genetic algorithm and data mining based meta-heuristic for job shop scheduling problem. IEEE

39. HO, J., C., CHANG, Y., 1991, A New Heuristic For The n-Job, m-Machine Flow-Shop Problem, European J.of Operations Research, 52, pp:194-202

40. HOLLAND, J.H., 1975, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor.

41. İNAN, O., 2003, Öğrenci İşleri Veritabanı Üzerinde Veri Madenciliği Uygulamaları, Yayınlanmamış Yüksek Lisans Tezi, SÜ, Konya

42. JAIN, N., BAGCHI, T.P., 2000, Hybridized Gas: Some New Results in Flowshop Scheduling, Modelling and Simulation (MS2000) International Conferance at Pittsburg in May 2000, http://citeseer.nj.nec.com

43. JIN, Z., YANG, Z., ITO, T., 2006, Metaheuristic algorithms for the multistage hybrid flowshop scheduling problem, International Journal of Production Economics, 100,322–334

44. KARAHOCA, A., 2004, Yapay Sinir Ağları Yardımı ile Telekom Sektöründe Stratejik Veri Madenciliği . www.bilisimhaftasi.org.tr/program_akademik.html

45. KOCHAR,S., MORRİS,R.J.T., 1987. Heuristic methods for flexible flow line scheduling, Journal of Manufacturing Systems, 6, 299-314.

46. KOONCE, D.A., TSAİ S.-C., 2000, Using data mining to find patterns in genetic algorithm solutions to a job shop schedule. Computers &Industrial Engineering 38 (2000) 361-374.

47. KOZA, J.R., 1992, Genetic Programming: on the Programming of Computers by Means of Natural Selection , MIT Press.

48. MOURSLI,O., POCHET,Y.A., 2000. Branch and bound algorithm for the hybrid flowshop, International journal of Production Economics, 64, 113-125.

49. MURATA, T., ISHIBUCHI, H., TANAKA, H., 1996a, Genetic Algorithms for Flow Shop Scheduling Problems, Computers and Industrial Enginering,Vol.30, No.4, 1061-1071.

50. MURATA, T., ISHIBUCHI, H., TANAKA, H., 1996b, Multi-Objective Genetic Algorithms and Its Applications to Flow Shop Scheduling. Computers and Industrial Engineering, vol 30, No 4, pp 957–968.

51. NARASIMHAN,S.L., PANWALKER,S.S., 1984, Scheduling in a two-stage manufacturing process, International Journal of Productions Research, 22, 555-564.

52. NEGENMANN,E.G., 2001. Local search algorithms for the multiprocessor flow shop scheduling problem, European Journal of Operational Research, 128, 147-158.

53. NERON,E., BAPTISTE,P., GUPTA, J.N.D., 2001, Solving hybrid flow shop problem using energetic reasoning and global operations, Omega, 29, 501-511.

54. OĞUZ, C., Kıyaslama Problemleri,
http://home.ku.edu.tr/~coguz/Research/ dataset2.zip

55. OĞUZ, C., ZINDER, Y., DO, V.H., JANIAK, A., LICHTENSTEIN M., 2004, Hybrid flow-shop scheduling problems with multiprocessor task systems, European Journal Of Operational Research, 152, 115-131.

56. OĞUZ, C., FİKRET, E., 2005, A genetic algorithm for hybrid flow-shop scheduling with multiprocessor tasks, Springer Science + Business Media, Inc. Manufactured in The Netherlands,323-351.

57. PORTMANN, M.C. , VIGNIER, A., DARDILHAC, D., DEZALAY, D., 1998. Branch and bound crossed with GA to solve hybrid flowshops, European Journal of Operational Research, 107, 389-400.

58. RAJENDRAN, C., CHAUDHURİ, D., 1992a. Scheduling in n-job, m-stage flowshop with paralel processors to minimize makespan. International Journal of Production Economics, 27,137-143.

59. RAJENDRAN, C., CHAUDHURI, D., 1992b. A multi-stage paralel processor flowshop problem with minimum flowtime. European Journal of Operational Research, 57, 111-122.

60. RIANE,F., ARTIBA,A., ELMAGHRABY, S.E.,1998. A hybrid three-stage flowshop problem: efficient heuristi cs to minimize makespan, European Journal of Operational Research, 109, 321-329.

61. SANTOS,D.L., HUNSUCKER,J.L., DEAL,D.E., 1995. Global lower bounds for flow shops with multiple processors, European Journal of Operational Research, 80, 112-120.

62. SHAPIRO, J., 2001,"Genetic Algorithms in Machine Learning", Volume 2049, Issue , pp 0146-Lecture Notes in Computer Science

63. SİVRİKAYA ŞERİFOĞLU,S.,F., ULUSOY G., 2004, Multiprocessor task scheduling in multistage hybrid flow shops: a genetic algorithm approach, Journal of the Operational Research Society, 55, 504-512.

64. SRISKANDARAJAH,C., SETHI,S.P.,1989. Scheduling algorithms for flexible flowshops: worst and average case performance, European Journal of Operational Research, 43,143-160.

65. SU, L.H**,** 2002, A hybrid two-stage flowshop with limited waiting time constraints Elsevier Science Ltd

66. TAGUCHİ, G., PHADKE, M.S., 1989. Quality Engineering Through Design Optimization, Quality Control, Robust Design and the Taguchi Method, Edited by Khosrow Dehnad, Wadsworth and Brooks/Cole Advanced Books and Software Pacific Grove, California, USA.

67. TANG , L., XUAN, H., LIU, J., (2006) A new Lagrangian relaxation algorithm for hybrid flowshop scheduling to minimize total weighted completion time, Computers&Operations Research, 33, 3344-3359

68. TANTUĞ, A.C. (2002).Veri madenciliği ve demetleme. Yüksek Lisans Tezi, İTU, İstanbul

69. TUĞ, E. (2004) Tıbbi Veri Tabanlarında Bilgi Keşfi. www.bilisimhaftasi.org.tr/program_akademik.html

70. VAHAPLAR, (2004). Veri Madenciliği ve Elektronik Ticaret. www.bayar.edu.tr/bid/dokumanlar/inceoglu.doc

71. WANG, Z. XİNG, W. BAI, F. (2005). No-wait flexible flowshop scheduling with no-idlemachines, Operational Research Letters, 33, 609-614.

72. WARDONO,B., FATHI,Y.A., 2003. Tabu search algorithm for the multi-stage paralel machine problem with limited buffer capacities, European Journal of Operational Research.

73. WITTROCK,R.J., 1988. An adaptable scheduling algorithm for flexible flow lines, Operations Research, 36, 445-453.

74. WITTROCK,R.J.,1985. Scheduling algorithms for flexible flow lines, IBM Journal of Research Development, 29, 401-412

75. YENİAY, Ö., An Overview of Genetic Algorithms, Anadolu Üniversitesi Bilim ve Teknoloji Dergisi, Cilt: 2, Sayı: 1, s. 37-49, Eskişehir, 2001.

EKLER

Ek 1- 1 10x5 tipi Neron ve Carlier problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,4	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	88	1	00:00:00.000
2	Rulet Çemberi	0,6	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	88	1	00:00:00.000
3	Rulet Çemberi	0,2	Pozisyona Dayalı	0,3	Ters Mutasyon	0,1	88	1	00:00:00.000
4	Rulet Çemberi	0,3	Pozisyona Dayalı	0,3	Ters Mutasyon	0,1	88	1	00:00:00.000
5	Rulet Çemberi	0,6	Pozisyona Dayalı	0,3	Ters Mutasyon	0,1	88	1	00:00:00.000
6	Rulet Çemberi	0,7	Pozisyona Dayalı	0,3	Ters Mutasyon	0,1	88	1	00:00:00.000
7	Rulet Çemberi	0,5	Pozisyona Dayalı	0,7	Ters Mutasyon	0,1	88	1	00:00:00.000
8	Rulet Çemberi	0,5	Pozisyona Dayalı	0,9	Ters Mutasyon	0,1	88	1	00:00:00.000
9	Rulet Çemberi	0,3	Pozisyona Dayalı	0,1	Ters Mutasyon	0,2	88	1	00:00:00.000
10	Rulet Çemberi	0,9	Pozisyona Dayalı	0,3	Ters Mutasyon	0,2	88	1	00:00:00.000
11	Rulet Çemberi	0,5	Pozisyona Dayalı	0,5	Ters Mutasyon	0,2	88	1	00:00:00.000
12	Rulet Çemberi	0,5	Pozisyona Dayalı	0,6	Ters Mutasyon	0,2	88	1	00:00:00.000
13	Rulet Çemberi	0,5	Pozisyona Dayalı	0,8	Ters Mutasyon	0,2	88	1	00:00:00.000
14	Rulet Çemberi	0,2	Pozisyona Dayalı	0,1	Ters Mutasyon	0,3	88	1	00:00:00.000
15	Rulet Çemberi	0,3	Pozisyona Dayalı	0,1	Ters Mutasyon	0,3	88	1	00:00:00.000
16	Rulet Çemberi	0,4	Pozisyona Dayalı	0,1	Ters Mutasyon	0,3	88	1	00:00:00.000
17	Rulet Çemberi	0,3	Pozisyona Dayalı	0,2	Ters Mutasyon	0,3	88	1	00:00:00.000
18	Rulet Çemberi	0,6	Pozisyona Dayalı	0,3	Ters Mutasyon	0,3	88	1	00:00:00.000
19	Rulet Çemberi	0,4	Pozisyona Dayalı	0,4	Ters Mutasyon	0,3	88	1	00:00:00.000
20	Rulet Çemberi	0,4	Pozisyona Dayalı	0,7	Ters Mutasyon	0,3	88	1	00:00:00.000
21	Rulet Çemberi	0,8	Pozisyona Dayalı	0,1	Ters Mutasyon	0,4	88	1	00:00:00.000
22	Rulet Çemberi	0,6	Pozisyona Dayalı	0,2	Ters Mutasyon	0,5	88	1	00:00:00.000
23	Rulet Çemberi	0,9	Pozisyona Dayalı	0,4	Ters Mutasyon	0,5	88	1	00:00:00.000
24	Rulet Çemberi	0,4	Pozisyona Dayalı	0,5	Ters Mutasyon	0,5	88	1	00:00:00.000
25	Rulet Çemberi	0,2	Pozisyona Dayalı	0,7	Ters Mutasyon	0,5	88	1	00:00:00.000
26	Rulet Çemberi	0,4	Pozisyona Dayalı	0,7	Ters Mutasyon	0,6	88	1	00:00:00.000
27	Rulet Çemberi	0,5	Pozisyona Dayalı	0,7	Ters Mutasyon	0,6	88	1	00:00:00.000
28	Rulet Çemberi	0,7	Pozisyona Dayalı	0,8	Ters Mutasyon	0,6	88	1	00:00:00.000
29	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Ters Mutasyon	0,7	88	1	00:00:00.000
30	Rulet Çemberi	0,9	Pozisyona Dayalı	0,1	Ters Mutasyon	0,8	88	1	00:00:00.000

Ek 1- 2 10x10 tipi Neron ve Carlier problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORAN	ÇAPRAZLAMA	ÇAPRAZLAMA ORAN	MUTASYON	MUTASYON ORAN	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Ters Mutasyon	0,1	169	1	00:00:00.000
2	Rulet Çemberi	0,1	Pozisyona Dayalı	0,3	Ters Mutasyon	0,1	169	1	00:00:00.000
3	Rulet Çemberi	0,1	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	169	1	00:00:00.000
4	Rulet Çemberi	0,4	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	169	1	00:00:00.000
5	Rulet Çemberi	0,9	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	169	1	00:00:00.000
6	Rulet Çemberi	0,4	Pozisyona Dayalı	0,6	Ters Mutasyon	0,1	169	1	00:00:00.000
7	Rulet Çemberi	0,6	Pozisyona Dayalı	0,1	Ters Mutasyon	0,2	169	1	00:00:00.000
8	Rulet Çemberi	0,1	Pozisyona Dayalı	0,2	Ters Mutasyon	0,2	169	1	00:00:00.000
9	Rulet Çemberi	0,7	Pozisyona Dayalı	0,4	Ters Mutasyon	0,2	169	1	00:00:00.000
10	Rulet Çemberi	0,1	Pozisyona Dayalı	0,5	Ters Mutasyon	0,2	169	1	00:00:00.000
11	Rulet Çemberi	0,1	Pozisyona Dayalı	0,8	Ters Mutasyon	0,2	169	1	00:00:00.000
12	Rulet Çemberi	0,3	Pozisyona Dayalı	0,8	Ters Mutasyon	0,2	169	1	00:00:00.000
13	Rulet Çemberi	0,7	Pozisyona Dayalı	0,1	Ters Mutasyon	0,3	169	1	00:00:00.000
14	Rulet Çemberi	0,8	Pozisyona Dayalı	0,2	Ters Mutasyon	0,3	169	1	00:00:00.000
15	Rulet Çemberi	0,9	Pozisyona Dayalı	0,2	Ters Mutasyon	0,3	169	1	00:00:00.000
16	Rulet Çemberi	0,4	Pozisyona Dayalı	0,5	Ters Mutasyon	0,3	169	1	00:00:00.000
17	Rulet Çemberi	0,2	Pozisyona Dayalı	0,6	Ters Mutasyon	0,3	169	1	00:00:00.000
18	Rulet Çemberi	0,6	Pozisyona Dayalı	0,3	Ters Mutasyon	0,4	169	1	00:00:00.000
19	Rulet Çemberi	0,3	Pozisyona Dayalı	0,1	Ters Mutasyon	0,5	169	1	00:00:00.000
20	Rulet Çemberi	0,8	Pozisyona Dayalı	0,2	Ters Mutasyon	0,5	169	1	00:00:00.000
21	Rulet Çemberi	0,5	Pozisyona Dayalı	0,3	Ters Mutasyon	0,5	169	1	00:00:00.000
22	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Ters Mutasyon	0,6	169	1	00:00:00.000
23	Rulet Çemberi	0,5	Pozisyona Dayalı	0,2	Ters Mutasyon	0,7	169	1	00:00:00.000
24	Rulet Çemberi	0,3	Pozisyona Dayalı	0,2	Ters Mutasyon	0,9	169	1	00:00:00.000
25	Rulet Çemberi	0,1	Pozisyona Dayalı	0,3	Ters Mutasyon	0,9	169	1	00:00:00.000
26	Rulet Çemberi	0,7	Pozisyona Dayalı	0,3	Komşu İki İşi Değiştirme	0,1	169	1	00:00:00.000
27	Rulet Çemberi	0,9	Pozisyona Dayalı	0,3	Komşu İki İşi Değiştirme	0,1	169	1	00:00:00.000
28	Rulet Çemberi	0,8	Pozisyona Dayalı	0,4	Komşu İki İşi Değiştirme	0,1	169	1	00:00:00.000
29	Rulet Çemberi	0,9	Pozisyona Dayalı	0,4	Komşu İki İşi Değiştirme	0,1	169	1	00:00:00.000
30	Rulet Çemberi	0,2	Pozisyona Dayalı	0,6	Komşu İki İşi Değiştirme	0,1	169	1	00:00:00.000

Ek 1- 3 15x5 tipi Neron ve Carlier problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,1	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	157	1	00:00:00.000
2	Rulet Çemberi	0,9	Pozisyona Dayalı	0,1	Ters Mutasyon	0,2	157	1	00:00:00.000
3	Rulet Çemberi	0,9	Pozisyona Dayalı	0,2	Ters Mutasyon	0,2	157	1	00:00:00.000
4	Rulet Çemberi	0,9	Pozisyona Dayalı	0,7	Ters Mutasyon	0,2	157	1	00:00:00.000
5	Rulet Çemberi	0,6	Pozisyona Dayalı	0,1	Ters Mutasyon	0,3	157	1	00:00:00.000
6	Rulet Çemberi	0,5	Pozisyona Dayalı	0,2	Ters Mutasyon	0,3	157	1	00:00:00.000
7	Rulet Çemberi	0,2	Pozisyona Dayalı	0,3	Ters Mutasyon	0,3	157	1	00:00:00.000
8	Rulet Çemberi	0,6	Pozisyona Dayalı	0,5	Ters Mutasyon	0,3	157	1	00:00:00.000
9	Rulet Çemberi	0,1	Pozisyona Dayalı	0,6	Ters Mutasyon	0,3	157	1	00:00:00.000
10	Rulet Çemberi	0,2	Pozisyona Dayalı	0,4	Ters Mutasyon	0,4	157	1	00:00:00.000
11	Rulet Çemberi	0,3	Pozisyona Dayalı	0,3	Ters Mutasyon	0,5	157	1	00:00:00.000
12	Rulet Çemberi	0,5	Pozisyona Dayalı	0,3	Ters Mutasyon	0,5	157	1	00:00:00.000
13	Rulet Çemberi	0,8	Pozisyona Dayalı	0,4	Ters Mutasyon	0,5	157	1	00:00:00.000
14	Rulet Çemberi	0,7	Pozisyona Dayalı	0,1	Ters Mutasyon	0,6	157	1	00:00:00.000
15	Rulet Çemberi	0,1	Pozisyona Dayalı	0,4	Ters Mutasyon	0,6	157	1	00:00:00.000
16	Rulet Çemberi	0,1	Pozisyona Dayalı	0,5	Ters Mutasyon	0,6	157	1	00:00:00.000
17	Rulet Çemberi	0,6	Pozisyona Dayalı	0,5	Ters Mutasyon	0,6	157	1	00:00:00.000
18	Rulet Çemberi	0,8	Pozisyona Dayalı	0,4	Komşu İki İşi Değiştirme	0,1	157	1	00:00:00.000
19	Rulet Çemberi	0,9	Pozisyona Dayalı	0,4	Komşu İki İşi Değiştirme	0,1	157	1	00:00:00.000
20	Rulet Çemberi	0,3	Pozisyona Dayalı	0,8	Komşu İki İşi Değiştirme	0,1	157	1	00:00:00.000
21	Rulet Çemberi	0,6	Pozisyona Dayalı	0,2	Komşu İki İşi Değiştirme	0,2	157	1	00:00:00.000
22	Rulet Çemberi	0,1	Pozisyona Dayalı	0,5	Komşu İki İşi Değiştirme	0,2	157	1	00:00:00.000
23	Rulet Çemberi	0,8	Pozisyona Dayalı	0,6	Komşu İki İşi Değiştirme	0,2	157	1	00:00:00.000
24	Rulet Çemberi	0,2	Pozisyona Dayalı	0,7	Komşu İki İşi Değiştirme	0,2	157	1	00:00:00.000
25	Rulet Çemberi	0,8	Pozisyona Dayalı	0,7	Komşu İki İşi Değiştirme	0,2	157	1	00:00:00.000
26	Rulet Çemberi	0,3	Pozisyona Dayalı	0,9	Komşu İki İşi Değiştirme	0,2	157	1	00:00:00.000
27	Rulet Çemberi	0,5	Pozisyona Dayalı	0,6	Komşu İki İşi Değiştirme	0,3	157	1	00:00:00.000
28	Rulet Çemberi	0,4	Pozisyona Dayalı	0,7	Komşu İki İşi Değiştirme	0,3	157	1	00:00:00.000
29	Rulet Çemberi	0,1	Pozisyona Dayalı	0,1	Komşu İki İşi Değiştirme	0,4	157	1	00:00:00.000
30	Rulet Çemberi	0,2	Pozisyona Dayalı	0,9	Komşu İki İşi Değiştirme	0,4	157	1	00:00:00.000

Ek 1- 14 15x10 tipi Neron ve Carlier problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,2	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	222	1	00:00:00.000
2	Rulet Çemberi	0,5	Pozisyona Dayalı	0,2	Ters Mutasyon	0,1	222	1	00:00:00.000
3	Rulet Çemberi	0,3	Pozisyona Dayalı	0,3	Araya Sokma	0,1	222	1	00:00:00.000
4	Rulet Çemberi	0,9	Pozisyona Dayalı	0,1	Araya Sokma	0,2	222	1	00:00:00.000
5	Rulet Çemberi	0,9	Sıraya Dayalı	0,1	Ters Mutasyon	0,3	222	1	00:00:00.000
6	Rulet Çemberi	0,3	Sıraya Dayalı	0,2	Komşu İki İşi Değiştirme	0,2	222	1	00:00:00.000
7	Rulet Çemberi	0,3	Sıraya Dayalı	0,2	Araya Sokma	0,2	222	1	00:00:00.000
8	Rulet Çemberi	0,5	Kısmi Planlı	0,1	Ters Mutasyon	0,1	222	1	00:00:00.000
9	Rulet Çemberi	0,4	Dairesel	0,2	Keyfi İki İşi Değiştirme	0,2	222	1	00:00:00.000
10	Rulet Çemberi	0,2	Doğrusal Sıralı	0,1	Ters Mutasyon	0,1	222	1	00:00:00.000
11	Rulet Çemberi	0,1	Doğrusal Sıralı	0,1	Komşu İki İşi Değiştirme	0,1	222	1	00:00:00.000
12	Rulet Çemberi	0,3	Sıralı	0,1	Komşu İki İşi Değiştirme	0,2	222	1	00:00:00.000
13	Rulet Çemberi	0,1	Sıralı	0,3	Komşu İki İşi Değiştirme	0,5	222	1	00:00:00.000
14	Rulet Çemberi	0,2	Sıralı	0,1	Keyfi Üç İşi Değiştirme	0,2	222	1	00:00:00.000
15	Rulet Çemberi	0,4	Sıralı	0,2	Keyfi Üç İşi Değiştirme	0,4	222	1	00:00:00.000
16	Tournament	2	Kısmi Planlı	0,3	Araya Sokma	0,3	222	1	00:00:00.000
17	Tournament	2	Doğrusal Sıralı	0,3	Keyfi İki İşi Değiştirme	0,3	222	1	00:00:00.000
18	Tournament	2	Sıralı	0,2	Keyfi Üç İşi Değiştirme	0,1	222	1	00:00:00.000
19	Rulet Çemberi	0,2	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	222	1	00:00:00.015
20	Rulet Çemberi	0,7	Pozisyona Dayalı	0,6	Ters Mutasyon	0,1	222	1	00:00:00.015
21	Rulet Çemberi	0,6	Pozisyona Dayalı	0,3	Ters Mutasyon	0,4	222	1	00:00:00.015
22	Rulet Çemberi	0,2	Pozisyona Dayalı	0,3	Ters Mutasyon	0,7	222	1	00:00:00.015
23	Rulet Çemberi	0,7	Pozisyona Dayalı	0,3	Komşu İki İşi Değiştirme	0,3	222	1	00:00:00.015
24	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Komşu İki İşi Değiştirme	0,6	222	1	00:00:00.015
25	Rulet Çemberi	0,3	Pozisyona Dayalı	0,2	Komşu İki İşi Değiştirme	0,6	222	1	00:00:00.015
26	Rulet Çemberi	0,5	Pozisyona Dayalı	0,2	Keyfi İki İşi Değiştirme	0,3	222	1	00:00:00.015
27	Rulet Çemberi	0,9	Pozisyona Dayalı	0,1	Keyfi İki İşi Değiştirme	0,4	222	1	00:00:00.015
28	Rulet Çemberi	0,2	Pozisyona Dayalı	0,3	Keyfi İki İşi Değiştirme	0,4	222	1	00:00:00.015
29	Rulet Çemberi	0,8	Pozisyona Dayalı	0,2	Keyfi İki İşi Değiştirme	0,7	222	1	00:00:00.015
30	Rulet Çemberi	0,3	Pozisyona Dayalı	0,3	Keyfi İki İşi Değiştirme	0,7	222	1	00:00:00.015

Ek 1- 5 5x2 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,2	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	267	1	00:00:00.000
2	Rulet Çemberi	0,3	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	267	1	00:00:00.000
3	Rulet Çemberi	0,4	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	267	1	00:00:00.000
4	Rulet Çemberi	0,5	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	267	1	00:00:00.000
5	Rulet Çemberi	0,6	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	267	1	00:00:00.000
6	Rulet Çemberi	0,8	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	267	1	00:00:00.000
7	Rulet Çemberi	0,9	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	267	1	00:00:00.000
8	Rulet Çemberi	0,1	Pozisyona Dayalı	0,2	Ters Mutasyon	0,1	267	1	00:00:00.000
9	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Ters Mutasyon	0,1	267	1	00:00:00.000
10	Rulet Çemberi	0,3	Pozisyona Dayalı	0,2	Ters Mutasyon	0,1	267	1	00:00:00.000
11	Rulet Çemberi	0,4	Pozisyona Dayalı	0,2	Ters Mutasyon	0,1	267	1	00:00:00.000
12	Rulet Çemberi	0,5	Pozisyona Dayalı	0,2	Ters Mutasyon	0,1	267	1	00:00:00.000
13	Rulet Çemberi	0,6	Pozisyona Dayalı	0,2	Ters Mutasyon	0,1	267	1	00:00:00.000
14	Rulet Çemberi	0,8	Pozisyona Dayalı	0,2	Ters Mutasyon	0,1	267	1	00:00:00.000
15	Rulet Çemberi	0,1	Pozisyona Dayalı	0,3	Ters Mutasyon	0,1	267	1	00:00:00.000
16	Rulet Çemberi	0,3	Pozisyona Dayalı	0,3	Ters Mutasyon	0,1	267	1	00:00:00.000
17	Rulet Çemberi	0,5	Pozisyona Dayalı	0,3	Ters Mutasyon	0,1	267	1	00:00:00.000
18	Rulet Çemberi	0,6	Pozisyona Dayalı	0,3	Ters Mutasyon	0,1	267	1	00:00:00.000
19	Rulet Çemberi	0,8	Pozisyona Dayalı	0,3	Ters Mutasyon	0,1	267	1	00:00:00.000
20	Rulet Çemberi	0,1	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	267	1	00:00:00.000
21	Rulet Çemberi	0,2	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	267	1	00:00:00.000
22	Rulet Çemberi	0,3	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	267	1	00:00:00.000
23	Rulet Çemberi	0,6	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	267	1	00:00:00.000
24	Rulet Çemberi	0,1	Pozisyona Dayalı	0,5	Ters Mutasyon	0,1	267	1	00:00:00.000
25	Rulet Çemberi	0,2	Pozisyona Dayalı	0,5	Ters Mutasyon	0,1	267	1	00:00:00.000
26	Rulet Çemberi	0,4	Pozisyona Dayalı	0,5	Ters Mutasyon	0,1	267	1	00:00:00.000
27	Rulet Çemberi	0,6	Pozisyona Dayalı	0,5	Ters Mutasyon	0,1	267	1	00:00:00.000
28	Rulet Çemberi	0,7	Pozisyona Dayalı	0,5	Ters Mutasyon	0,1	267	1	00:00:00.000
29	Rulet Çemberi	0,9	Pozisyona Dayalı	0,5	Ters Mutasyon	0,1	267	1	00:00:00.000
30	Rulet Çemberi	0,1	Pozisyona Dayalı	0,6	Ters Mutasyon	0,1	267	1	00:00:00.000

Ek 1- 6 10x2 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,4	Pozisyona Dayalı	0,1	Ters Mutasyon	0,1	526	1	00:00:00.000
2	Rulet Çemberi	0,7	Pozisyona Dayalı	0,3	Ters Mutasyon	0,1	526	1	00:00:00.000
3	Rulet Çemberi	0,4	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	526	1	00:00:00.000
4	Rulet Çemberi	0,7	Pozisyona Dayalı	0,1	Ters Mutasyon	0,3	526	1	00:00:00.000
5	Rulet Çemberi	0,1	Pozisyona Dayalı	0,2	Ters Mutasyon	0,3	526	1	00:00:00.000
6	Rulet Çemberi	0,3	Pozisyona Dayalı	0,2	Ters Mutasyon	0,3	526	1	00:00:00.000
7	Rulet Çemberi	0,6	Pozisyona Dayalı	0,1	Ters Mutasyon	0,4	526	1	00:00:00.000
8	Rulet Çemberi	0,2	Pozisyona Dayalı	0,3	Komşu İki İşi Değiştirme	0,2	526	1	00:00:00.000
9	Rulet Çemberi	0,6	Pozisyona Dayalı	0,1	Keyfi İki İşi Değiştirme	0,1	526	1	00:00:00.000
10	Rulet Çemberi	0,3	Pozisyona Dayalı	0,1	Keyfi İki İşi Değiştirme	0,4	526	1	00:00:00.000
11	Rulet Çemberi	0,7	Pozisyona Dayalı	0,1	Keyfi İki İşi Değiştirme	0,4	526	1	00:00:00.000
12	Rulet Çemberi	0,1	Pozisyona Dayalı	0,3	Keyfi İki İşi Değiştirme	0,6	526	1	00:00:00.000
13	Rulet Çemberi	0,4	Pozisyona Dayalı	0,3	Keyfi Üç İşi Değiştirme	0,2	526	1	00:00:00.000
14	Rulet Çemberi	0,5	Pozisyona Dayalı	0,1	Keyfi Üç İşi Değiştirme	0,3	526	1	00:00:00.000
15	Rulet Çemberi	0,5	Pozisyona Dayalı	0,3	Keyfi Üç İşi Değiştirme	0,3	526	1	00:00:00.000
16	Rulet Çemberi	0,1	Pozisyona Dayalı	0,2	Keyfi Üç İşi Değiştirme	0,4	526	1	00:00:00.000
17	Rulet Çemberi	0,3	Pozisyona Dayalı	0,3	Keyfi Üç İşi Değiştirme	0,5	526	1	00:00:00.000
18	Rulet Çemberi	0,8	Pozisyona Dayalı	0,1	Araya Sokma	0,1	526	1	00:00:00.000
19	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Araya Sokma	0,1	526	1	00:00:00.000
20	Rulet Çemberi	0,5	Pozisyona Dayalı	0,2	Araya Sokma	0,1	526	1	00:00:00.000
21	Rulet Çemberi	0,9	Pozisyona Dayalı	0,3	Araya Sokma	0,1	526	1	00:00:00.000
22	Rulet Çemberi	0,1	Pozisyona Dayalı	0,2	Araya Sokma	0,2	526	1	00:00:00.000
23	Rulet Çemberi	0,4	Pozisyona Dayalı	0,1	Araya Sokma	0,5	526	1	00:00:00.000
24	Rulet Çemberi	0,1	Sıraya Dayalı	0,1	Ters Mutasyon	0,7	526	1	00:00:00.000
25	Rulet Çemberi	0,8	Sıraya Dayalı	0,2	Komşu İki İşi Değiştirme	0,1	526	1	00:00:00.000
26	Rulet Çemberi	0,3	Sıraya Dayalı	0,3	Komşu İki İşi Değiştirme	0,1	526	1	00:00:00.000
27	Rulet Çemberi	0,1	Sıraya Dayalı	0,5	Komşu İki İşi Değiştirme	0,4	526	1	00:00:00.000
28	Rulet Çemberi	0,1	Sıraya Dayalı	0,5	Keyfi İki İşi Değiştirme	0,1	526	1	00:00:00.000
29	Rulet Çemberi	0,1	Sıraya Dayalı	0,4	Keyfi İki İşi Değiştirme	0,3	526	1	00:00:00.000
30	Rulet Çemberi	0,3	Sıraya Dayalı	0,1	Keyfi İki İşi Değiştirme	0,4	526	1	00:00:00.000

Ek 1- 7 20x2 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,4	Pozisyona Dayalı	0,4	Ters Mutasyon	0,1	869	1	00:00:00.000
2	Rulet Çemberi	0,1	Pozisyona Dayalı	0,2	Ters Mutasyon	0,3	869	1	00:00:00.000
3	Rulet Çemberi	0,8	Pozisyona Dayalı	0,6	Ters Mutasyon	0,1	869	1	00:00:00.015
4	Rulet Çemberi	0,8	Pozisyona Dayalı	0,5	Ters Mutasyon	0,2	869	1	00:00:00.015
5	Rulet Çemberi	0,1	Pozisyona Dayalı	0,6	Ters Mutasyon	0,9	869	1	00:00:00.015
6	Rulet Çemberi	0,2	Pozisyona Dayalı	0,5	Ters Mutasyon	0,1	869	1	00:00:00.016
7	Rulet Çemberi	0,3	Pozisyona Dayalı	0,3	Ters Mutasyon	0,2	869	1	00:00:00.016
8	Rulet Çemberi	0,1	Pozisyona Dayalı	0,4	Ters Mutasyon	0,2	869	1	00:00:00.016
9	Rulet Çemberi	0,6	Pozisyona Dayalı	0,3	Ters Mutasyon	0,3	869	1	00:00:00.016
10	Rulet Çemberi	0,8	Pozisyona Dayalı	0,2	Ters Mutasyon	0,4	869	1	00:00:00.016
11	Rulet Çemberi	0,5	Pozisyona Dayalı	0,3	Ters Mutasyon	0,4	869	1	00:00:00.016
12	Rulet Çemberi	0,3	Pozisyona Dayalı	0,4	Ters Mutasyon	0,4	869	1	00:00:00.016
13	Rulet Çemberi	0,8	Pozisyona Dayalı	0,1	Ters Mutasyon	0,5	869	1	00:00:00.016
14	Rulet Çemberi	0,6	Pozisyona Dayalı	0,1	Ters Mutasyon	0,6	869	1	00:00:00.016
15	Rulet Çemberi	0,3	Pozisyona Dayalı	0,6	Ters Mutasyon	0,6	869	1	00:00:00.016
16	Rulet Çemberi	0,6	Pozisyona Dayalı	0,2	Ters Mutasyon	0,8	869	1	00:00:00.016
17	Rulet Çemberi	0,2	Pozisyona Dayalı	0,1	Ters Mutasyon	0,9	869	1	00:00:00.016
18	Rulet Çemberi	0,3	Pozisyona Dayalı	0,6	Ters Mutasyon	0,2	869	1	00:00:00.031
19	Rulet Çemberi	0,5	Pozisyona Dayalı	0,8	Ters Mutasyon	0,3	869	1	00:00:00.031
20	Rulet Çemberi	0,1	Pozisyona Dayalı	0,6	Ters Mutasyon	0,4	869	1	00:00:00.031
21	Rulet Çemberi	0,4	Pozisyona Dayalı	0,7	Ters Mutasyon	0,4	869	1	00:00:00.031
22	Rulet Çemberi	0,7	Pozisyona Dayalı	0,7	Ters Mutasyon	0,4	869	1	00:00:00.031
23	Rulet Çemberi	0,7	Pozisyona Dayalı	0,4	Ters Mutasyon	0,5	869	1	00:00:00.031
24	Rulet Çemberi	0,9	Pozisyona Dayalı	0,6	Ters Mutasyon	0,5	869	1	00:00:00.031
25	Rulet Çemberi	0,6	Pozisyona Dayalı	0,5	Ters Mutasyon	0,6	869	1	00:00:00.031
26	Rulet Çemberi	0,9	Pozisyona Dayalı	0,5	Ters Mutasyon	0,7	869	1	00:00:00.031
27	Rulet Çemberi	0,5	Pozisyona Dayalı	0,4	Ters Mutasyon	0,8	869	1	00:00:00.031
28	Rulet Çemberi	0,3	Pozisyona Dayalı	0,6	Ters Mutasyon	0,8	869	1	00:00:00.031
29	Rulet Çemberi	0,1	Pozisyona Dayalı	0,9	Ters Mutasyon	0,3	869	2	00:00:00.031
30	Rulet Çemberi	0,4	Pozisyona Dayalı	0,3	Ters Mutasyon	0,5	869	2	00:00:00.031

Ek 1- 8 50x2 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,8	Sıraya Dayalı	0,8	Araya Sokma	0,5	1729	11	00:00:03.109
2	Rulet Çemberi	0,9	Doğrusal Sıralı	0,7	Araya Sokma	0,8	1729	14	00:00:03.703
3	Rulet Çemberi	0,7	Kısmi Planlı	0,4	Keyfi İki İşi Değiştirme	0,9	1730	12	00:00:02.250
4	Rulet Çemberi	0,7	Kısmi Planlı	0,1	Keyfi İki İşi Değiştirme	0,8	1732	13	00:00:01.360
5	Rulet Çemberi	0,5	Sıraya Dayalı	0,8	Keyfi İki İşi Değiştirme	0,5	1736	14	00:00:02.328
6	Rulet Çemberi	0,9	Doğrusal Sıralı	0,8	Araya Sokma	0,7	1738	12	00:00:03.375
7	Rulet Çemberi	0,8	Doğrusal Sıralı	0,8	Araya Sokma	0,9	1739	11	00:00:03.454
8	Rulet Çemberi	0,9	Sıraya Dayalı	0,1	Keyfi Üç İşi Değiştirme	0,9	1741	14	00:00:01.563
9	Rulet Çemberi	0,9	Pozisyona Dayalı	0,4	Keyfi İki İşi Değiştirme	0,9	1741	15	00:00:02.828
10	Rulet Çemberi	0,9	Sıralı	0,8	Keyfi İki İşi Değiştirme	0,8	1742	9	00:00:02.750
11	Rulet Çemberi	0,8	Sıraya Dayalı	0,6	Keyfi İki İşi Değiştirme	0,8	1743	13	00:00:03.563
12	Rulet Çemberi	0,9	Doğrusal Sıralı	0,8	Keyfi Üç İşi Değiştirme	0,8	1743	13	00:00:04.094
13	Rulet Çemberi	0,7	Sıraya Dayalı	0,8	Ters Mutasyon	0,8	1744	13	00:00:03.718
14	Rulet Çemberi	0,9	Sıraya Dayalı	0,4	Araya Sokma	0,7	1745	12	00:00:01.907
15	Rulet Çemberi	0,6	Doğrusal Sıralı	0,9	Keyfi Üç İşi Değiştirme	0,8	1745	8	00:00:02.531
16	Rulet Çemberi	0,9	Doğrusal Sıralı	0,7	Komşu İki İşi Değiştirme	0,7	1745	10	00:00:02.578
17	Rulet Çemberi	0,9	Sıraya Dayalı	0,7	Keyfi İki İşi Değiştirme	0,8	1745	14	00:00:03.891
18	Rulet Çemberi	0,6	Kısmi Planlı	0,6	Keyfi Üç İşi Değiştirme	0,6	1746	11	00:00:01.828
19	Rulet Çemberi	0,6	Doğrusal Sıralı	0,7	Araya Sokma	0,2	1747	15	00:00:02.063
20	Rulet Çemberi	0,8	Doğrusal Sıralı	0,7	Ters Mutasyon	0,3	1749	15	00:00:02.718
21	Rulet Çemberi	0,8	Doğrusal Sıralı	0,7	Keyfi Üç İşi Değiştirme	0,8	1749	11	00:00:03.047
22	Rulet Çemberi	0,1	Dairesel	0,5	Araya Sokma	0,3	1750	10	00:00:00.672
23	Rulet Çemberi	0,9	Sıraya Dayalı	0,5	Keyfi Üç İşi Değiştirme	0,5	1750	13	00:00:02.109
24	Rulet Çemberi	0,5	Dairesel	0,6	Keyfi Üç İşi Değiştirme	0,9	1750	13	00:00:02.563
25	Rulet Çemberi	0,9	Doğrusal Sıralı	0,6	Keyfi İki İşi Değiştirme	0,8	1751	10	00:00:02.437
26	Rulet Çemberi	0,7	Sıraya Dayalı	0,9	Keyfi İki İşi Değiştirme	0,4	1751	12	00:00:02.594
27	Rulet Çemberi	0,7	Doğrusal Sıralı	0,8	Keyfi İki İşi Değiştirme	0,9	1751	11	00:00:03.172
28	Rulet Çemberi	0,2	Sıralı	0,7	Keyfi İki İşi Değiştirme	0,5	1752	9	00:00:01.375
29	Rulet Çemberi	0,9	Kısmi Planlı	0,4	Ters Mutasyon	0,9	1752	11	00:00:02.062
30	Rulet Çemberi	0,8	Sıralı	0,8	Keyfi Üç İşi Değiştirme	0,4	1752	10	00:00:02.203

Ek 1- 9 100x2 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,2	Sıraya Dayalı	0,3	Keyfi Üç İşi Değiştirme	0,2	5094	1	00:00:00.203
2	Rulet Çemberi	0,8	Pozisyona Dayalı	0,1	Keyfi Üç İşi Değiştirme	0,1	5094	1	00:00:00.218
3	Rulet Çemberi	0,8	Sıraya Dayalı	0,2	Ters Mutasyon	0,1	5094	1	00:00:00.218
4	Rulet Çemberi	0,1	Dairesel	0,1	Araya Sokma	0,2	5094	1	00:00:00.218
5	Rulet Çemberi	0,6	Doğrusal Sıralı	0,1	Ters Mutasyon	0,1	5094	1	00:00:00.219
6	Rulet Çemberi	0,1	Doğrusal Sıralı	0,2	Araya Sokma	0,1	5094	1	00:00:00.219
7	Tournament	2	Sıraya Dayalı	0,1	Keyfi İki İşi Değiştirme	0,1	5094	1	00:00:00.219
8	Tournament	2	Kısmi Planlı	0,3	Keyfi Üç İşi Değiştirme	0,1	5094	1	00:00:00.219
9	Rulet Çemberi	0,4	Pozisyona Dayalı	0,2	Keyfi İki İşi Değiştirme	0,3	5094	1	00:00:00.234
10	Rulet Çemberi	0,2	Pozisyona Dayalı	0,3	Keyfi Üç İşi Değiştirme	0,2	5094	1	00:00:00.234
11	Tournament	2	Dairesel	0,1	Keyfi Üç İşi Değiştirme	0,1	5094	1	00:00:00.234
12	Rulet Çemberi	0,3	Pozisyona Dayalı	0,2	Komşu İki İşi Değiştirme	0,1	5094	1	00:00:00.235
13	Rulet Çemberi	0,2	Pozisyona Dayalı	0,3	Keyfi İki İşi Değiştirme	0,1	5094	1	00:00:00.235
14	Rulet Çemberi	0,7	Dairesel	0,2	Araya Sokma	0,1	5094	1	00:00:00.235
15	Tournament	2	Sıralı	0,1	Ters Mutasyon	0,2	5094	1	00:00:00.235
16	Rulet Çemberi	0,5	Pozisyona Dayalı	0,2	Komşu İki İşi Değiştirme	0,3	5094	1	00:00:00.250
17	Rulet Çemberi	0,2	Pozisyona Dayalı	0,3	Araya Sokma	0,1	5094	1	00:00:00.250
18	Rulet Çemberi	0,1	Sıraya Dayalı	0,1	Komşu İki İşi Değiştirme	0,3	5094	1	00:00:00.250
19	Rulet Çemberi	0,1	Sıraya Dayalı	0,2	Keyfi İki İşi Değiştirme	0,4	5094	1	00:00:00.250
20	Rulet Çemberi	0,7	Sıraya Dayalı	0,1	Araya Sokma	0,2	5094	1	00:00:00.250
21	Rulet Çemberi	0,4	Sıraya Dayalı	0,2	Araya Sokma	0,3	5094	1	00:00:00.250
22	Rulet Çemberi	0,3	Kısmi Planlı	0,3	Keyfi İki İşi Değiştirme	0,3	5094	1	00:00:00.250
23	Rulet Çemberi	0,1	Kısmi Planlı	0,3	Keyfi İki İşi Değiştirme	0,4	5094	1	00:00:00.250
24	Rulet Çemberi	0,1	Dairesel	0,3	Komşu İki İşi Değiştirme	0,4	5094	1	00:00:00.250
25	Rulet Çemberi	0,2	Dairesel	0,2	Keyfi İki İşi Değiştirme	0,2	5094	1	00:00:00.250
26	Rulet Çemberi	0,1	Dairesel	0,1	Keyfi Üç İşi Değiştirme	0,3	5094	1	00:00:00.250
27	Rulet Çemberi	0,7	Doğrusal Sıralı	0,1	Keyfi Üç İşi Değiştirme	0,1	5094	1	00:00:00.250
28	Rulet Çemberi	0,9	Doğrusal Sıralı	0,1	Araya Sokma	0,2	5094	1	00:00:00.250
29	Tournament	2	Sıraya Dayalı	0,3	Komşu İki İşi Değiştirme	0,4	5094	1	00:00:00.250
30	Rulet Çemberi	0,3	Pozisyona Dayalı	0,1	Araya Sokma	0,4	5094	1	00:00:00.265

Ek 1- 10 5x5 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Ters Mutasyon	0,2	511	1	00:00:00.015
2	Rulet Çemberi	0,1	Pozisyona Dayalı	0,1	Ters Mutasyon	0,4	511	1	00:00:00.015
3	Rulet Çemberi	0,2	Pozisyona Dayalı	0,1	Ters Mutasyon	0,8	511	1	00:00:00.015
4	Rulet Çemberi	0,6	Pozisyona Dayalı	0,4	Komşu İki İşi Değiştirme	0,1	511	1	00:00:00.015
5	Rulet Çemberi	0,7	Pozisyona Dayalı	0,1	Keyfi İki İşi Değiştirme	0,3	511	1	00:00:00.015
6	Rulet Çemberi	0,6	Pozisyona Dayalı	0,1	Keyfi Üç İşi Değiştirme	0,1	511	1	00:00:00.015
7	Rulet Çemberi	0,9	Pozisyona Dayalı	0,1	Keyfi Üç İşi Değiştirme	0,3	511	1	00:00:00.015
8	Rulet Çemberi	0,9	Pozisyona Dayalı	0,3	Keyfi Üç İşi Değiştirme	0,3	511	1	00:00:00.015
9	Rulet Çemberi	0,2	Sıraya Dayalı	0,3	Ters Mutasyon	0,1	511	1	00:00:00.015
10	Rulet Çemberi	0,2	Sıraya Dayalı	0,2	Ters Mutasyon	0,2	511	1	00:00:00.015
11	Rulet Çemberi	0,3	Sıraya Dayalı	0,1	Ters Mutasyon	0,7	511	1	00:00:00.015
12	Rulet Çemberi	0,6	Sıraya Dayalı	0,1	Komşu İki İşi Değiştirme	0,2	511	1	00:00:00.015
13	Rulet Çemberi	0,4	Sıraya Dayalı	0,2	Komşu İki İşi Değiştirme	0,7	511	1	00:00:00.015
14	Rulet Çemberi	0,9	Sıraya Dayalı	0,1	Keyfi İki İşi Değiştirme	0,1	511	1	00:00:00.015
15	Rulet Çemberi	0,3	Sıraya Dayalı	0,2	Keyfi İki İşi Değiştirme	0,1	511	1	00:00:00.015
16	Rulet Çemberi	0,1	Sıraya Dayalı	0,2	Keyfi İki İşi Değiştirme	0,3	511	1	00:00:00.015
17	Rulet Çemberi	0,3	Sıraya Dayalı	0,1	Keyfi Üç İşi Değiştirme	0,2	511	1	00:00:00.015
18	Rulet Çemberi	0,2	Sıraya Dayalı	0,6	Araya Sokma	0,1	511	1	00:00:00.015
19	Rulet Çemberi	0,4	Kısmi Planlı	0,2	Ters Mutasyon	0,1	511	1	00:00:00.015
20	Rulet Çemberi	0,5	Kısmi Planlı	0,3	Ters Mutasyon	0,2	511	1	00:00:00.015
21	Rulet Çemberi	0,5	Kısmi Planlı	0,1	Ters Mutasyon	0,5	511	1	00:00:00.015
22	Rulet Çemberi	0,3	Kısmi Planlı	0,1	Ters Mutasyon	0,8	511	1	00:00:00.015
23	Rulet Çemberi	0,2	Kısmi Planlı	0,3	Komşu İki İşi Değiştirme	0,3	511	1	00:00:00.015
24	Rulet Çemberi	0,5	Kısmi Planlı	0,3	Komşu İki İşi Değiştirme	0,3	511	1	00:00:00.015
25	Rulet Çemberi	0,1	Kısmi Planlı	0,3	Komşu İki İşi Değiştirme	0,8	511	1	00:00:00.015
26	Rulet Çemberi	0,4	Kısmi Planlı	0,3	Keyfi İki İşi Değiştirme	0,1	511	1	00:00:00.015
27	Rulet Çemberi	0,3	Kısmi Planlı	0,6	Keyfi Üç İşi Değiştirme	0,1	511	1	00:00:00.015
28	Rulet Çemberi	0,4	Kısmi Planlı	0,1	Keyfi Üç İşi Değiştirme	0,3	511	1	00:00:00.015
29	Rulet Çemberi	0,4	Kısmi Planlı	0,1	Keyfi Üç İşi Değiştirme	0,6	511	1	00:00:00.015
30	Rulet Çemberi	0,4	Kısmi Planlı	0,1	Araya Sokma	0,5	511	1	00:00:00.015

Ek 1- 11 10x5 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,9	Sıraya Dayalı	0,6	Komşu İki İşi Değiştirme	0,6	591	1	00:00:00.063
2	Rulet Çemberi	0,2	Sıraya Dayalı	0,2	Keyfi Üç İşi Değiştirme	0,6	591	6	00:00:00.093
3	Rulet Çemberi	0,1	Sıraya Dayalı	0,8	Araya Sokma	0,1	591	5	00:00:00.109
4	Tournament	2	Doğrusal Sıralı	0,5	Ters Mutasyon	0,5	591	5	00:00:00.109
5	Rulet Çemberi	0,2	Sıraya Dayalı	0,4	Araya Sokma	0,1	591	9	00:00:00.109
6	Rulet Çemberi	0,2	Doğrusal Sıralı	0,2	Keyfi İki İşi Değiştirme	0,7	591	6	00:00:00.110
7	Tournament	2	Sıralı	0,6	Ters Mutasyon	0,6	591	4	00:00:00.125
8	Tournament	2	Sıraya Dayalı	0,8	Keyfi Üç İşi Değiştirme	0,2	591	5	00:00:00.125
9	Tournament	2	Sıraya Dayalı	0,5	Keyfi Üç İşi Değiştirme	0,6	591	6	00:00:00.125
10	Rulet Çemberi	0,3	Sıraya Dayalı	0,3	Komşu İki İşi Değiştirme	0,7	591	8	00:00:00.125
11	Rulet Çemberi	0,8	Sıraya Dayalı	0,5	Araya Sokma	0,2	591	6	00:00:00.140
12	Rulet Çemberi	0,9	Kısmi Planlı	0,3	Ters Mutasyon	0,7	591	4	00:00:00.141
13	Rulet Çemberi	0,3	Sıraya Dayalı	0,6	Komşu İki İşi Değiştirme	0,5	591	5	00:00:00.141
14	Rulet Çemberi	0,9	Kısmi Planlı	0,2	Keyfi İki İşi Değiştirme	0,8	591	6	00:00:00.141
15	Tournament	2	Kısmi Planlı	0,6	Keyfi İki İşi Değiştirme	0,8	591	5	00:00:00.156
16	Rulet Çemberi	0,5	Dairesel	0,1	Araya Sokma	0,8	591	7	00:00:00.156
17	Rulet Çemberi	0,1	Dairesel	0,2	Keyfi İki İşi Değiştirme	0,3	591	16	00:00:00.156
18	Tournament	2	Dairesel	0,4	Keyfi İki İşi Değiştirme	0,7	591	7	00:00:00.157
19	Rulet Çemberi	0,7	Sıralı	0,2	Keyfi İki İşi Değiştirme	0,2	591	15	00:00:00.157
20	Rulet Çemberi	0,2	Doğrusal Sıralı	0,8	Araya Sokma	0,3	591	6	00:00:00.172
21	Rulet Çemberi	0,3	Sıraya Dayalı	0,2	Keyfi İki İşi Değiştirme	0,8	591	9	00:00:00.172
22	Rulet Çemberi	0,3	Doğrusal Sıralı	0,3	Keyfi İki İşi Değiştirme	0,6	591	11	00:00:00.172
23	Rulet Çemberi	0,2	Sıraya Dayalı	0,3	Komşu İki İşi Değiştirme	0,4	591	15	00:00:00.172
24	Rulet Çemberi	0,9	Sıralı	0,2	Araya Sokma	0,3	591	15	00:00:00.172
25	Rulet Çemberi	0,5	Sıralı	0,6	Araya Sokma	0,4	591	6	00:00:00.187
26	Tournament	2	Sıralı	0,9	Keyfi Üç İşi Değiştirme	0,2	591	8	00:00:00.187
27	Tournament	2	Doğrusal Sıralı	0,5	Araya Sokma	0,3	591	13	00:00:00.187
28	Tournament	2	Doğrusal Sıralı	0,2	Komşu İki İşi Değiştirme	0,3	591	20	00:00:00.188
29	Rulet Çemberi	0,5	Doğrusal Sıralı	0,8	Keyfi Üç İşi Değiştirme	0,7	591	4	00:00:00.203
30	Rulet Çemberi	0,3	Sıraya Dayalı	0,2	Keyfi Üç İşi Değiştirme	0,8	591	11	00:00:00.203

Ek 1- 12 20x5 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,9	Dairesel	0,9	Araya Sokma	0,6	972	12	00:00:02.656
2	Rulet Çemberi	0,5	Sıraya Dayalı	0,8	Keyfi İki İşi Değiştirme	0,8	975	15	00:00:02.578
3	Rulet Çemberi	0,3	Sıraya Dayalı	0,6	Araya Sokma	0,5	985	24	00:00:02.047
4	Rulet Çemberi	0,8	Kısmi Planlı	0,9	Araya Sokma	0,9	986	14	00:00:03.719
5	Rulet Çemberi	0,3	Sıraya Dayalı	0,2	Keyfi İki İşi Değiştirme	0,9	990	20	00:00:01.219
6	Rulet Çemberi	0,7	Sıraya Dayalı	0,8	Araya Sokma	0,3	990	20	00:00:02.485
7	Rulet Çemberi	0,7	Sıraya Dayalı	0,8	Araya Sokma	0,8	990	21	00:00:04.000
8	Rulet Çemberi	0,9	Dairesel	0,9	Keyfi İki İşi Değiştirme	0,9	991	14	00:00:03.438
9	Rulet Çemberi	0,4	Sıraya Dayalı	0,8	Araya Sokma	0,6	992	22	00:00:02.969
10	Rulet Çemberi	0,8	Pozisyona Dayalı	0,6	Keyfi İki İşi Değiştirme	0,7	992	22	00:00:03.688
11	Rulet Çemberi	0,6	Sıraya Dayalı	0,5	Araya Sokma	0,9	993	23	00:00:03.453
12	Rulet Çemberi	0,8	Sıraya Dayalı	0,9	Keyfi Üç İşi Değiştirme	0,8	993	16	00:00:03.953
13	Rulet Çemberi	0,6	Kısmi Planlı	0,5	Keyfi Üç İşi Değiştirme	0,8	994	10	00:00:01.453
14	Rulet Çemberi	0,7	Dairesel	0,7	Keyfi İki İşi Değiştirme	0,7	994	15	00:00:02.047
15	Rulet Çemberi	0,4	Sıraya Dayalı	0,7	Keyfi İki İşi Değiştirme	0,5	994	24	00:00:02.547
16	Rulet Çemberi	0,6	Sıraya Dayalı	0,8	Keyfi İki İşi Değiştirme	0,6	994	17	00:00:02.782
17	Rulet Çemberi	0,4	Doğrusal Sıralı	0,8	Keyfi İki İşi Değiştirme	0,8	995	13	00:00:02.203
18	Rulet Çemberi	0,7	Sıraya Dayalı	0,5	Keyfi İki İşi Değiştirme	0,8	996	25	00:00:03.515
19	Rulet Çemberi	0,8	Doğrusal Sıralı	0,8	Komşu İki İşi Değiştirme	0,1	997	8	00:00:00.921
20	Rulet Çemberi	0,3	Sıraya Dayalı	0,8	Ters Mutasyon	0,3	997	22	00:00:01.828
21	Rulet Çemberi	0,8	Dairesel	0,5	Araya Sokma	0,9	997	13	00:00:01.937
22	Rulet Çemberi	0,8	Sıralı	0,9	Keyfi Üç İşi Değiştirme	0,9	997	10	00:00:02.719
23	Rulet Çemberi	0,4	Sıraya Dayalı	0,8	Keyfi İki İşi Değiştirme	0,7	997	25	00:00:03.734
24	Rulet Çemberi	0,8	Sıraya Dayalı	0,8	Araya Sokma	0,4	997	24	00:00:03.968
25	Rulet Çemberi	0,9	Doğrusal Sıralı	0,6	Komşu İki İşi Değiştirme	0,5	998	11	00:00:01.468
26	Rulet Çemberi	0,6	Sıraya Dayalı	0,7	Keyfi İki İşi Değiştirme	0,8	998	19	00:00:03.234
27	Rulet Çemberi	0,5	Doğrusal Sıralı	0,6	Komşu İki İşi Değiştirme	0,8	999	13	00:00:01.797
28	Rulet Çemberi	0,9	Sıraya Dayalı	0,3	Keyfi Üç İşi Değiştirme	0,6	1000	20	00:00:01.578
29	Rulet Çemberi	0,9	Sıraya Dayalı	0,9	Araya Sokma	0,4	1000	11	00:00:02.156
30	Rulet Çemberi	0,1	Sıraya Dayalı	0,7	Keyfi İki İşi Değiştirme	0,9	1000	22	00:00:02.343

Ek 1- 13 50x5 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,7	Sıralı	0,4	Komşu İki İşi Değiştirme	0,1	2638	9	00:00:01.657
2	Rulet Çemberi	0,6	Pozisyona Dayalı	0,3	Ters Mutasyon	0,3	2638	12	00:00:02.078
3	Rulet Çemberi	0,9	Pozisyona Dayalı	0,1	Ters Mutasyon	0,9	2638	9	00:00:02.438
4	Rulet Çemberi	0,9	Sıraya Dayalı	0,6	Ters Mutasyon	0,4	2638	6	00:00:02.625
5	Tournament	2	Doğrusal Sıralı	0,9	Komşu İki İşi Değiştirme	0,7	2638	9	00:00:03.829
6	Rulet Çemberi	0,5	Sıraya Dayalı	0,9	Keyfi Üç İşi Değiştirme	0,4	2638	10	00:00:04.671
7	Rulet Çemberi	0,5	Pozisyona Dayalı	0,6	Keyfi İki İşi Değiştirme	0,4	2638	24	00:00:07.875
8	Rulet Çemberi	0,5	Pozisyona Dayalı	0,8	Keyfi Üç İşi Değiştirme	0,7	2638	25	00:00:12.156
9	Rulet Çemberi	0,9	Pozisyona Dayalı	0,3	Keyfi İki İşi Değiştirme	0,5	2642	20	00:00:04.703
10	Tournament	2	Sıraya Dayalı	0,9	Keyfi Üç İşi Değiştirme	0,2	2643	4	00:00:01.094
11	Rulet Çemberi	0,7	Sıraya Dayalı	0,9	Keyfi Üç İşi Değiştirme	0,5	2643	5	00:00:03.124
12	Rulet Çemberi	0,4	Pozisyona Dayalı	0,7	Ters Mutasyon	0,6	2643	8	00:00:03.344
13	Rulet Çemberi	0,9	Doğrusal Sıralı	0,4	Araya Sokma	0,3	2644	10	00:00:02.422
14	Tournament	2	Doğrusal Sıralı	0,8	Keyfi Üç İşi Değiştirme	0,6	2645	10	00:00:03.797
15	Rulet Çemberi	0,7	Sıraya Dayalı	0,9	Ters Mutasyon	0,5	2645	25	00:00:15.109
16	Tournament	2	Sıraya Dayalı	0,9	Keyfi Üç İşi Değiştirme	0,9	2646	3	00:00:01.516
17	Rulet Çemberi	0,8	Pozisyona Dayalı	0,1	Keyfi İki İşi Değiştirme	0,9	2646	16	00:00:04.110
18	Rulet Çemberi	0,4	Dairesel	0,9	Ters Mutasyon	0,8	2646	10	00:00:06.531
19	Rulet Çemberi	0,6	Sıraya Dayalı	0,5	Ters Mutasyon	0,5	2646	24	00:00:08.969
20	Rulet Çemberi	0,2	Pozisyona Dayalı	0,7	Araya Sokma	0,8	2646	20	00:00:09.062
21	Rulet Çemberi	0,9	Pozisyona Dayalı	0,8	Araya Sokma	0,9	2646	24	00:00:19.328
22	Rulet Çemberi	0,5	Doğrusal Sıralı	0,5	Keyfi Üç İşi Değiştirme	0,1	2647	9	00:00:01.594
23	Tournament	2	Sıraya Dayalı	0,8	Keyfi Üç İşi Değiştirme	0,3	2647	6	00:00:01.641
24	Rulet Çemberi	0,3	Dairesel	0,9	Araya Sokma	0,5	2647	6	00:00:02.578
25	Rulet Çemberi	0,5	Sıraya Dayalı	0,3	Ters Mutasyon	0,7	2647	11	00:00:02.938
26	Rulet Çemberi	0,9	Sıraya Dayalı	0,8	Komşu İki İşi Değiştirme	0,4	2647	8	00:00:04.437
27	Rulet Çemberi	0,3	Pozisyona Dayalı	0,3	Keyfi İki İşi Değiştirme	0,8	2648	16	00:00:03.157
28	Tournament	2	Doğrusal Sıralı	0,6	Araya Sokma	0,8	2649	10	00:00:03.500
29	Rulet Çemberi	0,6	Sıraya Dayalı	0,8	Komşu İki İşi Değiştirme	0,4	2650	2	00:00:01.078
30	Tournament	2	Doğrusal Sıralı	0,8	Komşu İki İşi Değiştirme	0,9	2650	5	00:00:02.375

Ek 1- 14 100x5 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,5	Sıralı	0,3	Araya Sokma	0,5	4985	7	00:00:03.515
2	Rulet Çemberi	0,6	Sıralı	0,6	Keyfi İki İşi Değiştirme	0,2	4996	9	00:00:05.047
3	Rulet Çemberi	0,8	Sıralı	0,7	Keyfi Üç İşi Değiştirme	0,8	4996	8	00:00:11.360
4	Rulet Çemberi	0,6	Sıralı	0,9	Keyfi Üç İşi Değiştirme	0,9	4999	10	00:00:16.266
5	Rulet Çemberi	0,9	Sıralı	0,4	Keyfi İki İşi Değiştirme	0,4	5001	10	00:00:06.187
6	Rulet Çemberi	0,5	Doğrusal Sıralı	0,5	Araya Sokma	0,9	5001	8	00:00:06.875
7	Tournament	2	Dairesel	0,7	Keyfi Üç İşi Değiştirme	0,9	5001	10	00:00:07.469
8	Rulet Çemberi	0,8	Doğrusal Sıralı	0,5	Araya Sokma	0,9	5001	7	00:00:08.109
9	Rulet Çemberi	0,6	Doğrusal Sıralı	0,4	Keyfi Üç İşi Değiştirme	0,9	5001	10	00:00:08.921
10	Rulet Çemberi	0,5	Pozisyona Dayalı	0,5	Ters Mutasyon	0,8	5001	13	00:00:10.343
11	Rulet Çemberi	0,8	Doğrusal Sıralı	0,2	Keyfi Üç İşi Değiştirme	0,5	5002	5	00:00:02.047
12	Rulet Çemberi	0,3	Sıralı	0,6	Komşu İki İşi Değiştirme	0,7	5003	9	00:00:06.343
13	Rulet Çemberi	0,3	Sıralı	0,1	Ters Mutasyon	0,6	5004	4	00:00:01.422
14	Rulet Çemberi	0,7	Doğrusal Sıralı	0,5	Ters Mutasyon	0,7	5004	6	00:00:05.922
15	Rulet Çemberi	0,7	Doğrusal Sıralı	0,5	Ters Mutasyon	0,3	5005	9	00:00:05.453
16	Rulet Çemberi	0,2	Pozisyona Dayalı	0,6	Ters Mutasyon	0,9	5005	19	00:00:15.000
17	Rulet Çemberi	0,6	Doğrusal Sıralı	0,4	Komşu İki İşi Değiştirme	0,1	5006	8	00:00:03.000
18	Tournament	2	Doğrusal Sıralı	0,5	Komşu İki İşi Değiştirme	0,7	5007	8	00:00:04.110
19	Rulet Çemberi	0,2	Sıralı	0,6	Keyfi İki İşi Değiştirme	0,5	5007	7	00:00:04.266
20	Rulet Çemberi	0,6	Sıralı	0,6	Araya Sokma	0,9	5008	9	00:00:09.922
21	Rulet Çemberi	0,5	Doğrusal Sıralı	0,3	Araya Sokma	0,9	5009	3	00:00:02.203
22	Rulet Çemberi	0,5	Sıralı	0,4	Komşu İki İşi Değiştirme	0,3	5009	8	00:00:02.984
23	Rulet Çemberi	0,2	Doğrusal Sıralı	0,7	Araya Sokma	0,6	5009	6	00:00:04.954
24	Rulet Çemberi	0,3	Pozisyona Dayalı	0,9	Ters Mutasyon	0,9	5009	5	00:00:06.266
25	Rulet Çemberi	0,7	Sıralı	0,9	Araya Sokma	0,8	5009	7	00:00:10.937
26	Rulet Çemberi	0,5	Sıralı	0,3	Keyfi Üç İşi Değiştirme	0,7	5010	10	00:00:05.640
27	Rulet Çemberi	0,1	Sıralı	0,9	Komşu İki İşi Değiştirme	0,9	5010	7	00:00:06.859
28	Rulet Çemberi	0,4	Doğrusal Sıralı	0,8	Keyfi İki İşi Değiştirme	0,4	5010	9	00:00:07.250
29	Tournament	2	Pozisyona Dayalı	0,1	Keyfi İki İşi Değiştirme	0,9	5011	5	00:00:02.187
30	Rulet Çemberi	0,6	Doğrusal Sıralı	0,4	Ters Mutasyon	0,7	5011	8	00:00:06.079

Ek 1- 15 5x8 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Komşu İki İşi Değiştirme	0,1	616	1	00:00:00.015
2	Rulet Çemberi	0,7	Pozisyona Dayalı	0,2	Komşu İki İşi Değiştirme	0,1	616	1	00:00:00.015
3	Rulet Çemberi	0,2	Pozisyona Dayalı	0,2	Keyfi İki İşi Değiştirme	0,1	616	1	00:00:00.015
4	Rulet Çemberi	0,1	Pozisyona Dayalı	0,1	Araya Sokma	0,1	616	1	00:00:00.015
5	Rulet Çemberi	0,2	Pozisyona Dayalı	0,1	Araya Sokma	0,1	616	1	00:00:00.015
6	Rulet Çemberi	0,7	Sıraya Dayalı	0,2	Ters Mutasyon	0,2	616	1	00:00:00.015
7	Rulet Çemberi	0,3	Sıraya Dayalı	0,3	Ters Mutasyon	0,2	616	1	00:00:00.015
8	Rulet Çemberi	0,1	Sıraya Dayalı	0,4	Ters Mutasyon	0,2	616	1	00:00:00.015
9	Rulet Çemberi	0,3	Sıraya Dayalı	0,1	Komşu İki İşi Değiştirme	0,2	616	1	00:00:00.015
10	Rulet Çemberi	0,3	Sıraya Dayalı	0,3	Komşu İki İşi Değiştirme	0,2	616	1	00:00:00.015
11	Rulet Çemberi	0,8	Sıraya Dayalı	0,2	Keyfi Üç İşi Değiştirme	0,2	616	1	00:00:00.015
12	Rulet Çemberi	0,3	Sıraya Dayalı	0,2	Keyfi Üç İşi Değiştirme	0,3	616	1	00:00:00.015
13	Rulet Çemberi	0,7	Sıraya Dayalı	0,2	Araya Sokma	0,1	616	1	00:00:00.015
14	Rulet Çemberi	0,1	Sıraya Dayalı	0,2	Araya Sokma	0,2	616	1	00:00:00.015
15	Rulet Çemberi	0,2	Kısmi Planlı	0,1	Ters Mutasyon	0,1	616	1	00:00:00.015
16	Rulet Çemberi	0,1	Kısmi Planlı	0,3	Komşu İki İşi Değiştirme	0,3	616	1	00:00:00.015
17	Rulet Çemberi	0,2	Kısmi Planlı	0,3	Keyfi İki İşi Değiştirme	0,1	616	1	00:00:00.015
18	Rulet Çemberi	0,2	Kısmi Planlı	0,1	Keyfi Üç İşi Değiştirme	0,1	616	1	00:00:00.015
19	Rulet Çemberi	0,4	Kısmi Planlı	0,2	Keyfi Üç İşi Değiştirme	0,3	616	1	00:00:00.015
20	Rulet Çemberi	0,1	Kısmi Planlı	0,1	Araya Sokma	0,4	616	1	00:00:00.015
21	Rulet Çemberi	0,3	Dairesel	0,2	Keyfi İki İşi Değiştirme	0,2	616	1	00:00:00.015
22	Rulet Çemberi	0,2	Dairesel	0,3	Keyfi İki İşi Değiştirme	0,2	616	1	00:00:00.015
23	Rulet Çemberi	0,4	Dairesel	0,2	Keyfi Üç İşi Değiştirme	0,1	616	1	00:00:00.015
24	Rulet Çemberi	0,2	Doğrusal Sıralı	0,2	Ters Mutasyon	0,2	616	1	00:00:00.015
25	Rulet Çemberi	0,2	Doğrusal Sıralı	0,2	Keyfi İki İşi Değiştirme	0,1	616	1	00:00:00.015
26	Rulet Çemberi	0,8	Sıralı	0,1	Ters Mutasyon	0,1	616	1	00:00:00.015
27	Rulet Çemberi	0,8	Sıralı	0,1	Keyfi İki İşi Değiştirme	0,1	616	1	00:00:00.015
28	Rulet Çemberi	0,3	Sıralı	0,1	Keyfi Üç İşi Değiştirme	0,3	616	1	00:00:00.015
29	Rulet Çemberi	0,5	Sıralı	0,2	Araya Sokma	0,3	616	1	00:00:00.015
30	Rulet Çemberi	0,1	Sıralı	0,1	Araya Sokma	0,4	616	1	00:00:00.015

Ek 1- 16 10x8 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,2	Dairesel	0,2	Araya Sokma	0,9	767	2	00:00:00.172
2	Rulet Çemberi	0,3	Sıraya Dayalı	0,3	Komşu İki İşi Değiştirme	0,9	767	4	00:00:00.235
3	Rulet Çemberi	0,4	Sıraya Dayalı	0,6	Komşu İki İşi Değiştirme	0,8	767	2	00:00:00.265
4	Tournament	2	Doğrusal Sıralı	0,5	Keyfi Üç İşi Değiştirme	0,2	767	7	00:00:00.281
5	Rulet Çemberi	0,4	Sıralı	0,4	Keyfi Üç İşi Değiştirme	0,9	767	3	00:00:00.297
6	Rulet Çemberi	0,2	Kısmi Planlı	0,2	Araya Sokma	0,9	767	5	00:00:00.312
7	Tournament	2	Pozisyona Dayalı	0,8	Keyfi İki İşi Değiştirme	0,1	767	6	00:00:00.328
8	Tournament	2	Kısmi Planlı	0,1	Komşu İki İşi Değiştirme	0,9	767	6	00:00:00.328
9	Rulet Çemberi	0,9	Kısmi Planlı	0,4	Keyfi İki İşi Değiştirme	0,6	767	3	00:00:00.343
10	Rulet Çemberi	0,7	Doğrusal Sıralı	0,2	Komşu İki İşi Değiştirme	0,1	767	16	00:00:00.344
11	Rulet Çemberi	0,1	Doğrusal Sıralı	0,2	Araya Sokma	0,7	767	9	00:00:00.359
12	Rulet Çemberi	0,1	Sıraya Dayalı	0,3	Araya Sokma	0,6	767	8	00:00:00.360
13	Rulet Çemberi	0,5	Sıraya Dayalı	0,2	Keyfi Üç İşi Değiştirme	0,6	767	8	00:00:00.391
14	Rulet Çemberi	0,9	Sıralı	0,2	Keyfi İki İşi Değiştirme	0,3	767	14	00:00:00.406
15	Rulet Çemberi	0,9	Sıralı	0,1	Keyfi İki İşi Değiştirme	0,4	767	11	00:00:00.421
16	Rulet Çemberi	0,4	Sıraya Dayalı	0,1	Keyfi İki İşi Değiştirme	0,9	767	6	00:00:00.422
17	Rulet Çemberi	0,6	Doğrusal Sıralı	0,4	Keyfi İki İşi Değiştirme	0,2	767	8	00:00:00.422
18	Rulet Çemberi	0,9	Sıraya Dayalı	0,1	Komşu İki İşi Değiştirme	0,3	767	14	00:00:00.438
19	Rulet Çemberi	0,8	Kısmi Planlı	0,9	Komşu İki İşi Değiştirme	0,6	767	2	00:00:00.453
20	Rulet Çemberi	0,8	Sıralı	0,7	Keyfi Üç İşi Değiştirme	0,5	767	3	00:00:00.469
21	Rulet Çemberi	0,7	Sıralı	0,4	Ters Mutasyon	0,4	767	6	00:00:00.469
22	Tournament	2	Sıraya Dayalı	0,4	Komşu İki İşi Değiştirme	0,9	767	6	00:00:00.469
23	Tournament	2	Doğrusal Sıralı	0,1	Araya Sokma	0,8	767	10	00:00:00.469
24	Rulet Çemberi	0,9	Doğrusal Sıralı	0,4	Keyfi Üç İşi Değiştirme	0,4	767	6	00:00:00.484
25	Rulet Çemberi	0,1	Doğrusal Sıralı	0,1	Keyfi İki İşi Değiştirme	0,4	767	17	00:00:00.484
26	Rulet Çemberi	0,4	Sıralı	0,1	Komşu İki İşi Değiştirme	0,5	767	13	00:00:00.485
27	Rulet Çemberi	0,4	Doğrusal Sıralı	0,4	Ters Mutasyon	0,1	767	14	00:00:00.485
28	Rulet Çemberi	0,5	Sıralı	0,7	Araya Sokma	0,5	767	5	00:00:00.500
29	Rulet Çemberi	0,2	Doğrusal Sıralı	0,4	Araya Sokma	0,4	767	9	00:00:00.500
30	Rulet Çemberi	0,9	Doğrusal Sıralı	0,4	Komşu İki İşi Değiştirme	0,7	767	5	00:00:00.515

Ek 1- 17 20x8 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,5	Sıraya Dayalı	0,7	Komşu İki İşi Değiştirme	0,2	1208	11	00:00:01.328
2	Rulet Çemberi	0,9	Sıralı	0,9	Keyfi Üç İşi Değiştirme	0,4	1208	14	00:00:04.359
3	Rulet Çemberi	0,3	Sıralı	0,8	Komşu İki İşi Değiştirme	0,6	1209	15	00:00:02.892
4	Rulet Çemberi	0,9	Sıralı	0,8	Keyfi İki İşi Değiştirme	0,4	1210	13	00:00:03.593
5	Rulet Çemberi	0,7	Sıraya Dayalı	0,6	Araya Sokma	0,9	1214	12	00:00:03.140
6	Rulet Çemberi	0,9	Sıraya Dayalı	0,9	Komşu İki İşi Değiştirme	0,6	1215	9	00:00:03.297
7	Rulet Çemberi	0,2	Sıralı	0,7	Komşu İki İşi Değiştirme	0,7	1216	14	00:00:02.876
8	Rulet Çemberi	0,5	Sıralı	0,6	Araya Sokma	0,6	1216	15	00:00:03.000
9	Rulet Çemberi	0,8	Sıralı	0,5	Araya Sokma	0,9	1218	14	00:00:03.765
10	Rulet Çemberi	0,6	Sıralı	0,6	Keyfi Üç İşi Değiştirme	0,4	1220	4	00:00:00.750
11	Rulet Çemberi	0,9	Doğrusal Sıralı	0,6	Ters Mutasyon	0,2	1220	9	00:00:01.422
12	Rulet Çemberi	0,4	Sıraya Dayalı	0,6	Ters Mutasyon	0,3	1220	10	00:00:01.500
13	Rulet Çemberi	0,6	Sıralı	0,3	Araya Sokma	0,5	1220	12	00:00:01.500
14	Tournament	2	Sıraya Dayalı	0,4	Keyfi Üç İşi Değiştirme	0,8	1220	15	00:00:02.000
15	Tournament	2	Sıraya Dayalı	0,4	Keyfi Üç İşi Değiştirme	0,8	1220	15	00:00:02.000
16	Tournament	2	Doğrusal Sıralı	0,6	Komşu İki İşi Değiştirme	0,6	1220	15	00:00:02.172
17	Tournament	2	Doğrusal Sıralı	0,6	Komşu İki İşi Değiştirme	0,6	1220	15	00:00:02.172
18	Rulet Çemberi	0,1	Kısmi Planlı	0,8	Keyfi Üç İşi Değiştirme	0,5	1220	14	00:00:02.391
19	Tournament	2	Doğrusal Sıralı	0,8	Araya Sokma	0,9	1220	11	00:00:02.516
20	Tournament	2	Doğrusal Sıralı	0,8	Araya Sokma	0,9	1220	11	00:00:02.516
21	Rulet Çemberi	0,2	Sıralı	0,5	Keyfi İki İşi Değiştirme	0,8	1220	14	00:00:02.547
22	Rulet Çemberi	0,7	Kısmi Planlı	0,5	Araya Sokma	0,7	1220	13	00:00:02.906
23	Rulet Çemberi	0,8	Sıraya Dayalı	0,7	Keyfi İki İşi Değiştirme	0,6	1220	10	00:00:02.922
24	Rulet Çemberi	0,7	Kısmi Planlı	0,5	Keyfi İki İşi Değiştirme	0,6	1220	15	00:00:03.015
25	Rulet Çemberi	0,5	Sıraya Dayalı	0,5	Keyfi Üç İşi Değiştirme	0,9	1220	15	00:00:03.031
26	Rulet Çemberi	0,1	Sıraya Dayalı	0,8	Keyfi İki İşi Değiştirme	0,7	1220	15	00:00:03.219
27	Rulet Çemberi	0,6	Sıraya Dayalı	0,8	Araya Sokma	0,6	1220	10	00:00:03.250
28	Rulet Çemberi	0,9	Sıralı	0,8	Komşu İki İşi Değiştirme	0,4	1220	13	00:00:03.375
29	Rulet Çemberi	0,7	Sıralı	0,7	Keyfi Üç İşi Değiştirme	0,5	1220	15	00:00:03.437
30	Rulet Çemberi	0,5	Kısmi Planlı	0,9	Araya Sokma	0,6	1220	13	00:00:03.516

Ek 1- 18 50x8 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,7	Pozisyona Dayalı	0,3	Keyfi Üç İşi Değiştirme	0,9	2711	16	00:00:06.547
2	Rulet Çemberi	0,6	Dairesel	0,7	Keyfi İki İşi Değiştirme	0,8	2715	8	00:00:05.547
3	Rulet Çemberi	0,8	Pozisyona Dayalı	0,5	Keyfi Üç İşi Değiştirme	0,6	2721	20	00:00:10.000
4	Rulet Çemberi	0,9	Kısmi Planlı	0,9	Keyfi İki İşi Değiştirme	0,9	2726	10	00:00:10.875
5	Rulet Çemberi	0,5	Sıraya Dayalı	0,8	Keyfi Üç İşi Değiştirme	0,2	2728	10	00:00:03.891
6	Rulet Çemberi	0,8	Sıralı	0,6	Keyfi Üç İşi Değiştirme	0,9	2728	10	00:00:07.639
7	Rulet Çemberi	0,8	Doğrusal Sıralı	0,5	Keyfi İki İşi Değiştirme	0,4	2729	9	00:00:03.875
8	Rulet Çemberi	0,7	Doğrusal Sıralı	0,9	Araya Sokma	0,5	2731	4	00:00:02.875
9	Rulet Çemberi	0,9	Pozisyona Dayalı	0,5	Keyfi İki İşi Değiştirme	0,6	2732	17	00:00:08.516
10	Rulet Çemberi	0,8	Sıraya Dayalı	0,6	Araya Sokma	0,6	2733	10	00:00:06.094
11	Rulet Çemberi	0,8	Sıraya Dayalı	0,7	Araya Sokma	0,5	2734	8	00:00:05.218
12	Rulet Çemberi	0,4	Doğrusal Sıralı	0,7	Araya Sokma	0,9	2734	9	00:00:05.265
13	Rulet Çemberi	0,8	Doğrusal Sıralı	0,7	Keyfi Üç İşi Değiştirme	0,9	2734	9	00:00:07.406
14	Rulet Çemberi	0,8	Sıralı	0,7	Ters Mutasyon	0,7	2734	10	00:00:07.734
15	Rulet Çemberi	0,2	Sıralı	0,8	Keyfi Üç İşi Değiştirme	0,9	2736	6	00:00:03.562
16	Tournament	2	Sıralı	0,9	Keyfi İki İşi Değiştirme	0,5	2736	10	00:00:04.078
17	Rulet Çemberi	0,9	Sıraya Dayalı	0,6	Araya Sokma	0,6	2739	8	00:00:04.937
18	Rulet Çemberi	0,8	Dairesel	0,6	Keyfi İki İşi Değiştirme	0,8	2739	10	00:00:07.578
19	Rulet Çemberi	0,2	Kısmi Planlı	0,8	Keyfi İki İşi Değiştirme	0,8	2741	8	00:00:04.515
20	Rulet Çemberi	0,8	Kısmi Planlı	0,4	Keyfi Üç İşi Değiştirme	0,8	2741	10	00:00:05.108
21	Rulet Çemberi	0,7	Doğrusal Sıralı	0,4	Araya Sokma	0,6	2743	5	00:00:02.032
22	Rulet Çemberi	0,3	Sıralı	0,9	Keyfi İki İşi Değiştirme	0,5	2744	7	00:00:03.719
23	Rulet Çemberi	0,3	Doğrusal Sıralı	0,9	Keyfi Üç İşi Değiştirme	0,8	2744	10	00:00:06.031
24	Rulet Çemberi	0,8	Doğrusal Sıralı	0,6	Keyfi İki İşi Değiştirme	0,8	2745	10	00:00:07.656
25	Rulet Çemberi	0,9	Sıraya Dayalı	0,7	Araya Sokma	0,6	2746	8	00:00:05.688
26	Rulet Çemberi	0,5	Sıralı	0,9	Keyfi İki İşi Değiştirme	0,9	2746	8	00:00:06.672
27	Rulet Çemberi	0,9	Doğrusal Sıralı	0,6	Araya Sokma	0,7	2747	9	00:00:05.750
28	Rulet Çemberi	0,3	Sıralı	0,9	Keyfi İki İşi Değiştirme	0,9	2747	9	00:00:06.359
29	Rulet Çemberi	0,9	Doğrusal Sıralı	0,8	Araya Sokma	0,8	2747	9	00:00:08.062
30	Rulet Çemberi	0,3	Sıralı	0,9	Keyfi İki İşi Değiştirme	0,3	2748	10	00:00:04.125

Ek 1- 19 100x8 tipi Oğuz problemleri için optimizasyon sonucu

	ÜREME	ÜREME ORANI	ÇAPRAZLAMA	ÇAPRAZLAMA ORANI	MUTASYON	MUTASYON ORANI	CMAX	İTERASYON	ZAMAN
1	Rulet Çemberi	0,1	Kısmi Planlı	0,9	Araya Sokma	0,9	5219	8	00:00:10.250
2	Rulet Çemberi	0,9	Sıraya Dayalı	0,8	Keyfi İki İşi Değiştirme	0,4	5228	10	00:00:15.922
3	Rulet Çemberi	0,9	Sıraya Dayalı	0,7	Ters Mutasyon	0,1	5234	10	00:00:10.266
4	Rulet Çemberi	0,6	Sıralı	0,6	Araya Sokma	0,4	5243	10	00:00:09.828
5	Rulet Çemberi	0,4	Sıralı	0,8	Araya Sokma	0,6	5245	4	00:00:05.828
6	Rulet Çemberi	0,1	Doğrusal Sıralı	0,4	Ters Mutasyon	0,5	5248	4	00:00:02.547
7	Rulet Çemberi	0,8	Doğrusal Sıralı	0,6	Keyfi Üç İşi Değiştirme	0,7	5250	10	00:00:15.891
8	Rulet Çemberi	0,2	Doğrusal Sıralı	0,6	Keyfi Üç İşi Değiştirme	0,8	5253	5	00:00:05.484
9	Rulet Çemberi	0,7	Sıraya Dayalı	0,5	Keyfi Üç İşi Değiştirme	0,5	5254	7	00:00:08.078
10	Rulet Çemberi	0,7	Sıraya Dayalı	0,8	Keyfi Üç İşi Değiştirme	0,6	5257	4	00:00:07.031
11	Rulet Çemberi	0,6	Sıraya Dayalı	0,8	Keyfi İki İşi Değiştirme	0,4	5257	10	00:00:14.125
12	Tournament	2	Doğrusal Sıralı	0,7	Keyfi İki İşi Değiştirme	0,6	5259	7	00:00:06.015
13	Rulet Çemberi	0,6	Sıralı	0,9	Keyfi Üç İşi Değiştirme	0,8	5260	9	00:00:18.969
14	Rulet Çemberi	0,9	Sıraya Dayalı	0,6	Araya Sokma	0,5	5261	6	00:00:08.093
15	Rulet Çemberi	0,8	Sıraya Dayalı	0,9	Keyfi İki İşi Değiştirme	0,1	5261	7	00:00:08.843
16	Rulet Çemberi	0,6	Doğrusal Sıralı	0,4	Ters Mutasyon	0,2	5262	10	00:00:05.781
17	Rulet Çemberi	0,7	Sıraya Dayalı	0,9	Keyfi İki İşi Değiştirme	0,8	5262	10	00:00:20.781
18	Rulet Çemberi	0,8	Kısmi Planlı	0,3	Keyfi Üç İşi Değiştirme	0,7	5263	8	00:00:07.453
19	Rulet Çemberi	0,6	Doğrusal Sıralı	0,8	Keyfi Üç İşi Değiştirme	0,7	5263	9	00:00:16.469
20	Tournament	2	Sıralı	0,8	Keyfi Üç İşi Değiştirme	0,1	5264	10	00:00:06.406
21	Rulet Çemberi	0,2	Doğrusal Sıralı	0,8	Ters Mutasyon	0,4	5264	8	00:00:07.484
22	Rulet Çemberi	0,6	Sıraya Dayalı	0,9	Komşu İki İşi Değiştirme	0,3	5265	7	00:00:10.094
23	Rulet Çemberi	0,9	Sıralı	0,7	Ters Mutasyon	0,2	5265	9	00:00:10.109
24	Rulet Çemberi	0,5	Doğrusal Sıralı	0,6	Araya Sokma	0,5	5266	7	00:00:07.578
25	Rulet Çemberi	0,7	Sıraya Dayalı	0,9	Keyfi Üç İşi Değiştirme	0,8	5266	9	00:00:19.359
26	Rulet Çemberi	0,6	Dairesel	0,4	Keyfi İki İşi Değiştirme	0,3	5267	6	00:00:04.296
27	Rulet Çemberi	0,4	Doğrusal Sıralı	0,9	Araya Sokma	0,1	5267	8	00:00:07.562
28	Rulet Çemberi	0,3	Sıraya Dayalı	0,7	Komşu İki İşi Değiştirme	0,2	5268	9	00:00:06.750
29	Rulet Çemberi	0,4	Doğrusal Sıralı	0,9	Araya Sokma	0,6	5268	9	00:00:14.421
30	Rulet Çemberi	0,4	Sıralı	0,9	Araya Sokma	0,6	5268	10	00:00:15.703

Ek 2 PROGRAMIN KAYNAK KODLARI

```
unit un_anaform;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, Grids, ExtCtrls,Math,FileCtrl;

type
  Tfr_anaform = class(TForm)
    dosya: TOpenDialog;
    bt_yukle: TBitBtn;
    Shape1: TShape;
    Shape2: TShape;
    kaydet: TSaveDialog;
    Panel1: TPanel;
    bt_coz: TBitBtn;
    bt_iptal: TBitBtn;
    tx_iteras: TEdit;
    Label4: TLabel;
    Label3: TLabel;
    tx_bas_pop: TEdit;
    chb_kalanyerden: TCheckBox;
    chb_tum: TCheckBox;
    lb_bilgi: TLabel;
    Panel2: TPanel;
    tx_mutasi: TEdit;
    Label2: TLabel;
    tx_caprazi: TEdit;
    Label1: TLabel;
    tx_uremei: TEdit;
    Label5: TLabel;
    Label6: TLabel;
    tx_tour: TEdit;
    rg_capraz: TRadioGroup;
    rg_mutas: TRadioGroup;
    rg_ureme: TRadioGroup;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    tx_uremes: TEdit;
    tx_uremea: TEdit;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label17: TLabel;
    Label18: TLabel;
```

```
    tx_caprazs: TEdit;
    tx_capraza: TEdit;
    tx_mutass: TEdit;
    tx_mutasa: TEdit;
    lb_ilerleme: TLabel;
    bt_klasor: TButton;
    Label7: TLabel;
    tx_calismasay: TEdit;

    procedure bt_yukleClick(Sender: TObject);
    procedure bt_cozClick(Sender: TObject);
    function isyerlestir(asama,isno, jobfinish: integer):integer;
    function gantbul(gen:integer): integer;
    procedure baspopulasyon;
    procedure kromozomduzelt(kromo:integer);
    procedure ruletcemberi;
    procedure quicksort(iLo,iHi:integer);
    procedure tournament;
    procedure pozisyonadayali;
    procedure sirayadayali;
    procedure kismiplanli;
    procedure dairesel;
    procedure dogrusalsirali;
    procedure sirali;
    procedure tersmutasyon;
    procedure komsuikiis;
    procedure keyfiikiis;
    procedure keyfiucis;
    procedure arayasokma;
    procedure coz;
    procedure eniyinesilsec;
    procedure sonuckaydet;
    procedure anacoz;
    procedure chb_tumClick(Sender: TObject);
    procedure bt_iptalClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure chb_kalanyerdenClick(Sender: TObject);
    procedure giriskontrol(Sender: TObject; var Key: Char);
    procedure bt_klasorClick(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  fr_anaform: Tfr_anaform;
  jobsize,stagesize,touroran,cmax,mmax,baspop,kromozomsay,iterasyon:integer;
  machines,eniyinesil: array of integer;
  kromozom: array of array of integer;
```

```
  jobs: array of array of array of integer;
  gant: array of array of array of boolean;
  cputime: array of TDateTime;
  uremeoran,caprazoran,mutasoran:extended;
  cik,klasorcoz:boolean;
implementation


{$R *.dfm}

procedure Tfr_anaform.bt_yukleClick(Sender: TObject);
var
exfile: TextFile;
line,temp:string;
i,j,k: integer;
begin
 if (klasorcoz=true) or (dosya.Execute = true) then
 begin
  mmax :=0;
  Assignfile(exfile,dosya.FileName);
  Reset(exfile);
  readln(exfile,line);
  line := trim(line);
  jobsize := strtoint(copy(line,5,length(line)-4));
  readln(exfile,line);
  line := trim(line);
  stagesize := strtoint(copy(line,7,length(line)-6));
// Makine sayıları
  SetLength(machines, stagesize);
  readln(exfile,line);
  line := trim(line);
  line := copy(line,10,length(line)-8);
  line := line + ' ';
  temp := emptystr;
  j:=0;
  for i := 1 to length(line) do
   if line[i]<> ' ' then
    temp:= temp+line[i]
   else
   begin
    machines[j]:=strtoint(temp);
//En büyük makina sayısı
    if mmax<machines[j] then
     mmax := machines[j];
    j:=j+1;
    temp := emptystr;
   end;
  SetLength(jobs, stagesize, jobsize, 2);
// İşlem size
  readln(exfile,line);
```

```
  for k := 0 to stagesize-1 do
  begin
   readln(exfile,line);
   line := trim(line);
   line := line + ' ';
   temp := emptystr;
   j:=0;
   for i := 1 to length(line) do
    if line[i]<> ' ' then
     temp:= temp+line[i]
    else
    begin
     jobs[k,j,0]:=strtoint(temp);
     j:=j+1;
     temp := emptystr;
    end;
  end;
// İşlem süreleri
  readln(exfile,line);
  for k := 0 to stagesize-1 do
  begin
   readln(exfile,line);
   line := trim(line);
   line := line + ' ';
   temp := emptystr;
   j:=0;
   for i := 1 to length(line) do
    if line[i]<> ' ' then
     temp:= temp+line[i]
    else
    begin
     jobs[k,j,1]:=strtoint(temp);
     j:=j+1;
     temp := emptystr;
    end;
  end;
  closefile(exfile);
  bt_coz.Enabled:=true;
 end;
end;

function Tfr_anaform.gantbul(gen:integer): integer;
var
i,j: integer;
jobfinish: array of integer;
begin
 setlength(gant,0);
 setlength(jobfinish,jobsize);
 cmax :=0;
 for i:= 0 to stagesize-1 do
```

```
  for j:= 0 to jobsize-1 do
   jobfinish[j] := isyerlestir(i,kromozom[gen,j],jobfinish[j]);
 kromozom[gen,jobsize]:= cmax;
 kromozom[gen,jobsize+1]:= iterasyon;
 Result:=cmax;
end;

function Tfr_anaform.isyerlestir(asama,isno,jobfinish: integer):integer;
var
i,j,k,size: integer;
bos: boolean;
bosyer: array of array of integer;
begin
 size := jobs[asama,isno,0];
 setlength(bosyer,size,2);
 k:=jobfinish;
 if cmax < jobs[asama,isno,1]+k then
 begin
  cmax := jobs[asama,isno,1]+k;
  SetLength(gant, stagesize, mmax, cmax);
 end;
 while size <> 0 do
 begin
  for i := 0 to machines[asama]-1 do
  begin
   for j:=0 to jobs[asama,isno,1]-1 do
    if gant[asama,i,j+k] = false then
     bos := true
    else
    begin
     bos := false;
     break;
    end;
    if bos = true then
    begin
     bosyer[size-1,0]:= i;
     bosyer[size-1,1]:= k;
     size := size-1;
    end;
    if size = 0 then break;
  end;
  if size <> 0 then //iş bu sütundan baş. sığdırıldımı
  begin
   size := jobs[asama,isno,0]; //size orjinal haline
   k := k+1; //bir sütun ileri
   if jobs[asama,isno,1]+k > cmax then //gant yetersizse genişletilir
   begin
    cmax := cmax+1;
    SetLength(gant, stagesize, mmax, cmax);
   end;
```

```
  end;
 end;
 for i := 0 to jobs[asama,isno,0] - 1 do
  for j:=bosyer[i,1] to jobs[asama,isno,1]+bosyer[i,1]-1 do
   gant[asama,bosyer[i,0],j]:=true;
 Result := jobs[asama,isno,1]+bosyer[0,1];
end;

procedure Tfr_anaform.bt_cozClick(Sender: TObject);
var
i,j,k:integer;
a,b,c,kalansay:integer;
temp:string;
begin
 temp := '';
 a:=trunc(strtofloat(tx_uremes.text)*1000);
 b:=trunc(strtofloat(tx_uremei.text)*1000);
 c:=trunc(strtofloat(tx_uremea.text)*1000);
 if ((a-b) mod c <> 0) or (a<b) then
   temp:='Üreme';
 a:=trunc(strtofloat(tx_caprazs.text)*1000);
 b:=trunc(strtofloat(tx_caprazi.text)*1000);
 c:=trunc(strtofloat(tx_capraza.text)*1000);
 if ((a-b) mod c <> 0) or (a<b) then
   temp:='Çaprazlama';
 a:=trunc(strtofloat(tx_mutass.text)*1000);
 b:=trunc(strtofloat(tx_mutasi.text)*1000);
 c:=trunc(strtofloat(tx_mutasa.text)*1000);
 if ((a-b) mod c <> 0) or (a<b) then
   temp:='Mutasyon';
 if temp <> '' then
 begin
  messagedlg(temp+' oranları düzgün değil!',mtError,[mbOK],0);
  exit;
 end;
 if klasorcoz=false then
 begin
   kaydet.FileName := copy(dosya.FileName,1,length(dosya.FileName)-3)+'csv';
   if kaydet.Execute = false then exit;
   if (FileExists(kaydet.FileName)=true) and (Messagedlg('Sonuç dosyası temizlensin
mi?',mtConfirmation,[mbYes,mbNo],0)= mrYes) then
     DeleteFile(kaydet.FileName);
 end;
 for kalansay := 1 to strtoint(tx_calismasay.text) do
 begin
  bt_iptal.enabled:=true;
  cik:=false;
  setlength(cputime,strtoint(tx_iteras.text)+1);
  lb_ilerleme.Caption := '0';
  setlength(eniyinesil,0);
```

```
  setlength(eniyinesil,jobsize+2);
  if chb_kalanyerden.Checked = false then
  begin
   rg_ureme.ItemIndex := 0;
   rg_capraz.ItemIndex := 0;
   rg_mutas.ItemIndex := 0;
  end;
  if chb_tum.Checked = true then
  begin
   for i:= rg_ureme.ItemIndex to rg_ureme.Items.Count-1 do
   begin
    for j:= rg_capraz.ItemIndex to rg_capraz.Items.Count-1 do
    begin
     for k:= rg_mutas.ItemIndex to rg_mutas.Items.Count-1 do
     begin
      rg_ureme.ItemIndex :=i;
      rg_capraz.ItemIndex :=j;
      rg_mutas.ItemIndex :=k;
      anacoz;
     end;
     rg_mutas.ItemIndex := 0;
    end;
    rg_capraz.ItemIndex := 0;
   end;
   rg_ureme.ItemIndex := 0;
  end
  else
  begin
   anacoz;
  end;
  bt_iptal.enabled:=false;
 end;
 if klasorcoz=false then
   Messagedlg('Promlem çözüm işlemi
tamamlandı.'+#13+kaydet.FileName,mtInformation,[mbOK],0);
end;

procedure Tfr_anaform.coz;
begin
////üreme yöntemleri
 case rg_ureme.ItemIndex of
  0:ruletcemberi;
  1:tournament;
 end;
////çaprazlama yöntemleri
 case rg_capraz.ItemIndex of
  0:pozisyonadayali;
  1:sirayadayali;
  2:kismiplanli;
  3:dairesel;
```

```
  4:dogrusalsirali;
  5:sirali;
 end;
////mutasyon yöntemleri
 case rg_mutas.ItemIndex of
  0:tersmutasyon;
  1:komsuikiis;
  2:keyfiikiis;
  3:keyfiucis;
  4:arayasokma;
 end;
end;

procedure Tfr_anaform.eniyinesilsec;
var
i,j:integer;
begin
 if iterasyon = 1 then
  for j:= 0 to jobsize+1 do
   eniyinesil[j] := kromozom[0,j]
 else
  for i:= 0 to kromozomsay-1 do
   if eniyinesil[jobsize]>kromozom[i,jobsize] then
    for j:= 0 to jobsize+1 do
     eniyinesil[j] := kromozom[i,j]
end;

procedure Tfr_anaform.sonuckaydet;
var
exfile: TextFile;
line,temp:string;
i,j,top: integer;
begin
  Assignfile(exfile,kaydet.FileName);
  if FileExists(kaydet.FileName) = true then
   Append(exfile)
  else
  begin
   rewrite(exfile);
   line:='ÜREME;ÜREME ORAN;ÇAPRAZLAMA;ÇAPRAZLAMA
ORAN;MUTASYON;MUTASYON ORAN;';
   for i:= 0 to jobsize-1 do
    line:=line+inttostr(i+1)+';O.SR;';
   line:=line+'CMAX;İTERASYON;ZAMAN';
   writeln(exfile,line);
  end;
  line:= rg_ureme.Items[rg_ureme.ItemIndex];
   if rg_ureme.ItemIndex = 0 then
    line:= line+';'+floattostr(uremeoran)
   else
```

```
   line:= line+';'+inttostr(touroran);
   line:= line+';'+rg_capraz.Items[rg_capraz.ItemIndex]
   +';'+floattostr(caprazoran)
   +';'+rg_mutas.Items[rg_mutas.ItemIndex]
   +';'+floattostr(mutasoran)+';';
  for i:= 0 to jobsize-1 do
  begin
   line:=line+inttostr(eniyinesil[i]+1)+';';
   top:=0;
   for j:=0 to stagesize-1 do
     top:=top+jobs[j,eniyinesil[i],1];
   line:=line+inttostr(top div stagesize)+';'
  end;
  line:=line+inttostr(eniyinesil[jobsize])
   +';'+inttostr(eniyinesil[jobsize+1]);
  LongTimeFormat := 'hh:mm:ss.zzz';
  DateTimeToString(temp, 'tt', cputime[eniyinesil[jobsize+1]]-cputime[0]);
  line:=line+';'+temp;
  writeln(exfile,line);
  closefile(exfile);
end;

procedure Tfr_anaform.baspopulasyon;
var
i,j,k: integer;
dizi: array of boolean;
begin
  setlength(kromozom,0,0);
  setlength(kromozom,strtoint(tx_bas_pop.text),jobsize+2);
  baspop := strtoint(tx_bas_pop.text);
  randomize;
  for i:=(strtoint(tx_bas_pop.text)-1) downto 0 do
  begin
  setlength(dizi,0);
  setlength(dizi,jobsize);
    for j:=jobsize-1 downto 0 do
    begin
      k := random(jobsize-1)+1;
      while dizi[k] = true do
      begin
        k:=k+1;
        if k> jobsize-1 then k:=0
      end;
      dizi[k]:= true;
      kromozom[i,j]:= k;
    end;
//Cmax değerleri bulunuyor
  gantbul(i);
  end;
end;
```

```
procedure Tfr_anaform.ruletcemberi;
var
i,j,say: integer;
ras:extended;
yuzdeler: array of extended;
begin
  setlength(yuzdeler,baspop);
  j:= kromozom[baspop-1,jobsize];
  for i:=baspop-2 downto 0 do
    j:=j+kromozom[i,jobsize];
  for i:=baspop-1 downto 0 do
    yuzdeler[i]:=(j+baspop-kromozom[i,jobsize])/((baspop-1)*(j+1)+1);
  for i:=1 to baspop-1 do
    yuzdeler[i]:=yuzdeler[i]+yuzdeler[i-1];
  say:=trunc(baspop*uremeoran);
  if say=0 then say:=1;
  randomize;
  for j:=say downto 1 do
  begin
   ras:=random;
   for i:=0 to baspop-1 do
    if yuzdeler[i] >= ras then
    begin
     kromozomsay:=kromozomsay+1;
     setlength(kromozom,kromozomsay,jobsize+2);
     kromozom[kromozomsay-1]:=copy(kromozom[i],0,jobsize+2);
     break;
    end;
  end;
end;

procedure Tfr_anaform.tournament;
var
i:integer;
begin
 quicksort(0,kromozomsay-1);
 for i := touroran-1 downto 0 do
  kromozom[kromozomsay-1-i]:=kromozom[i]
end;

procedure Tfr_anaform.pozisyonadayali;
var
i,j,k,l,z,y,gen:integer;
m:boolean;
bosyer:array of integer;
begin
 randomize;
 gen:= trunc(jobsize/2);
 m:=false;
```

```
 z:= trunc(kromozomsay*caprazoran);
 if z=0 then z:=1;
 for y:=1 to z do
 begin
  setlength(bosyer,0);
  setlength(bosyer,gen);
  i:=gen;
  while i<>0 do
  begin
   l:=random(jobsize-1)+1;
   for k:= i-1 to gen-1 do
    if bosyer[k]=l then
     m:=false
    else
     m:=true;
   if m=true then
   begin
    bosyer[i-1]:=l;
    i:=i-1;
   end;
  end;
  m:=false;
  i:=random(kromozomsay);
  j:=random(kromozomsay-1)+1;
  while j=i do
   j:=random(kromozomsay-1)+1;
  kromozomsay:=kromozomsay+2;
  setlength(kromozom,kromozomsay,jobsize+2);
  for k:=0 to jobsize-1 do
  begin
   for l:= gen-1 downto 0 do
    if bosyer[l]=k then
    begin
     kromozom[kromozomsay-2,bosyer[l]] := kromozom[j,bosyer[l]];
     kromozom[kromozomsay-1,bosyer[l]] := kromozom[i,bosyer[l]];
     m:=true;
    end;
   if m=false then
   begin
    kromozom[kromozomsay-2,k] := kromozom[i,k];
    kromozom[kromozomsay-1,k] := kromozom[j,k];
   end;
  end;
  kromozomduzelt(kromozomsay-1);
  kromozomduzelt(kromozomsay-2);
  gantbul(kromozomsay-1);
  gantbul(kromozomsay-2);
 end;
end;
```

```
procedure Tfr_anaform.kromozomduzelt(kromo:integer);
var
i,j,k:integer;
olan:array of boolean;
begin
 setlength(olan,0);
 setlength(olan,jobsize);
 for i:= jobsize-1 downto 0 do
  if kromozom[kromo,i]<>jobsize then
   olan[kromozom[kromo,i]]:=true;
 for i:= 0 to jobsize-1 do
  for j:= jobsize-1 downto i+1 do
   if (kromozom[kromo,i]=kromozom[kromo,j]) or (kromozom[kromo,i]=jobsize) then
    for k:= 0 to jobsize-1 do
     if olan[k]=false then
     begin
      kromozom[kromo,i]:=k;
      olan[k]:= true;
      break;
     end;
end;

procedure Tfr_anaform.sirayadayali;
var
i,j,k,z,y,gen,k1,k2:integer;
sabit:array of integer;
m:boolean;
begin
 randomize;
 gen:= trunc(jobsize/2);
 setlength(sabit,gen);
 z:= trunc(kromozomsay*caprazoran);
 if z=0 then z:=1;
 for y:=1 to z do
 begin
  k1:=random(kromozomsay-y*2+2);
  k2:=random(kromozomsay-y*2+2);
  while k2=k1 do
   k2:=random(kromozomsay-y*2+2);
  kromozomsay:=kromozomsay+2;
  setlength(kromozom,kromozomsay,jobsize+2);
  for i:=jobsize-1 downto 0 do
  begin
   kromozom[kromozomsay-2,i]:=jobsize;
   kromozom[kromozomsay-1,i]:=jobsize;
  end;
  i:=0;
  for j:=gen downto 1 do
  begin
   while kromozom[kromozomsay-2,i]<>jobsize do
```

```
   i:=random(jobsize);
  kromozom[kromozomsay-2,i]:=kromozom[k1,i];
  kromozom[kromozomsay-1,i]:=kromozom[k2,i];
  sabit[j-1]:=i;
 end;
 for j := gen-1 downto 0 do
 begin
  m:=false;
  for k:= jobsize-1 downto 0 do
   if kromozom[kromozomsay-2,k] = kromozom[k2,sabit[j]] then
   begin
    m:=true;
    break;
   end;
  if m=false then
   for k:= 0 to jobsize-1 do
    if kromozom[kromozomsay-2,k]=jobsize then
    begin
     kromozom[kromozomsay-2,k]:=kromozom[k2,sabit[j]];
     break;
    end;
  m:=false;
  for k:= jobsize-1 downto 0 do
   if kromozom[kromozomsay-1,k] = kromozom[k1,sabit[j]] then
   begin
    m:=true;
    break;
   end;
  if m=false then
   for k:= 0 to jobsize-1 do
    if kromozom[kromozomsay-1,k]=jobsize then
    begin
     kromozom[kromozomsay-1,k]:=kromozom[k1,sabit[j]];
     break;
    end;
 end;
 for k:= jobsize-1 downto 0 do
 begin
  if kromozom[kromozomsay-2,k]=jobsize then
   kromozom[kromozomsay-2,k]:=kromozom[k2,k];
  if kromozom[kromozomsay-1,k]=jobsize then
   kromozom[kromozomsay-1,k]:=kromozom[k1,k];
 end;
 kromozomduzelt(kromozomsay-1);
 kromozomduzelt(kromozomsay-2);
 gantbul(kromozomsay-1);
 gantbul(kromozomsay-2);
end;
end;
```

```
procedure Tfr_anaform.kismiplanli;
var
i,j,k,z,y,k1,k2:integer;
begin
 randomize;
 z:= trunc(kromozomsay*caprazoran);
 if z=0 then z:=1;
 for y:=1 to z do
 begin
  k1:=random(kromozomsay-y*2+2);
  k2:=random(kromozomsay-y*2+2);
  while k2=k1 do
   k2:=random(kromozomsay-y*2+2);
  i:=random(jobsize);
  j:=random(jobsize);
  while j=i do
   j:=random(jobsize-1)+1;
  if i < j then
  begin
   k:=i;
   i:=j;
   j:=k;
  end;
  kromozomsay:=kromozomsay+2;
  setlength(kromozom,kromozomsay,jobsize+2);
  for k:=0 to jobsize-1 do
  begin
   if (k<i) and (k>=j) then
   begin
    kromozom[kromozomsay-2,k] := kromozom[k2,k];
    kromozom[kromozomsay-1,k] := kromozom[k1,k];
   end
   else
   begin
    kromozom[kromozomsay-2,k] := kromozom[k1,k];
    kromozom[kromozomsay-1,k] := kromozom[k2,k];
   end;
  end;
  kromozomduzelt(kromozomsay-1);
  kromozomduzelt(kromozomsay-2);
  gantbul(kromozomsay-1);
  gantbul(kromozomsay-2);
 end;
end;

procedure Tfr_anaform.dairesel;
var
i,j,k,z,y,l:integer;
m:boolean;
begin
```

```
 randomize;
 z:= trunc(kromozomsay*caprazoran);
 if z=0 then z:=1;
 for y:=1 to z do
 begin
  i:=random(kromozomsay-y*2+2);
  j:=random(kromozomsay-y*2+2);
  while j=i do
   j:=random(kromozomsay-y*2+2);
  kromozomsay:=kromozomsay+2;
  setlength(kromozom,kromozomsay,jobsize+2);
  for k:= jobsize-1 downto 0 do
  begin
   kromozom[kromozomsay-1,k]:=jobsize;
   kromozom[kromozomsay-2,k]:=jobsize;
  end;
//1. kromozom için
  m:=false;
  l:=0;
  kromozom[kromozomsay-2,l]:= kromozom[i,l];
  while m=false do
  begin
   for k:= jobsize-1 downto 0 do
    if kromozom[i,k]=kromozom[j,l] then
    begin
     kromozom[kromozomsay-2,k] := kromozom[j,l];
     l := k;
     break;
    end;
   for k:= jobsize-1 downto 0 do
    if kromozom[kromozomsay-2,k]=kromozom[i,k] then m:=true;
  end;
//2. kromozom için
  m:=false;
  l:=0;
  kromozom[kromozomsay-1,l]:= kromozom[j,l];
  while m=false do
  begin
   for k:= jobsize-1 downto 0 do
    if kromozom[j,k]=kromozom[i,l] then
    begin
     kromozom[kromozomsay-1,k] := kromozom[i,l];
     l := k;
     break;
    end;
   for k:= jobsize-1 downto 0 do
    if kromozom[kromozomsay-1,k]=kromozom[j,k] then m:=true;
  end;
  for k:= jobsize-1 downto 0 do
  begin
```

```
   if kromozom[kromozomsay-2,k]=jobsize then
    kromozom[kromozomsay-2,k]:=kromozom[j,k];
   if kromozom[kromozomsay-1,k]=jobsize then
    kromozom[kromozomsay-1,k]:=kromozom[i,k];
  end;
  kromozomduzelt(kromozomsay-1);
  kromozomduzelt(kromozomsay-2);
  gantbul(kromozomsay-1);
  gantbul(kromozomsay-2);
 end;
end;

procedure Tfr_anaform.quicksort(iLo,iHi:integer);
var
Lo, Hi, Mid, T, M: Integer;
begin
 Lo := iLo;
 Hi := iHi;
 Mid := kromozom[(Lo + Hi) div 2,jobsize];
    repeat
      while kromozom[Lo,jobsize] < Mid do Inc(Lo);
      while kromozom[Hi,jobsize] > Mid do Dec(Hi);
      if Lo <= Hi then
      begin
        for T:= jobsize+1 downto 0 do
        begin
         M:= kromozom[Lo,T];
         kromozom[Lo,T]:=kromozom[Hi,T];
         kromozom[Hi,T]:=M;
        end;
        Inc(Lo);
        Dec(Hi);
      end;
    until Lo > Hi;
    if Hi > iLo then quicksort(iLo, Hi);
    if Lo < iHi then quicksort(Lo, iHi);
end;

procedure Tfr_anaform.dogrusalsirali;
var
i,z,y,k1,k2,b1,b2,uz:integer;
begin
 randomize;
 z:= trunc(kromozomsay*caprazoran);
 if z=0 then z:=1;
 for y:=1 to z do
 begin
  k1:=random(kromozomsay-y*2+2);
  k2:=random(kromozomsay-y*2+2);
  while k2=k1 do
```

```
  k2:=random(kromozomsay-y*2+2);
  uz:=random(jobsize-1)+1;
  b1:=random(jobsize-uz);
  b2:=random(jobsize-uz);
  kromozomsay:=kromozomsay+2;
  setlength(kromozom,kromozomsay,jobsize+2);
  kromozom[kromozomsay-2]:=copy(kromozom[k1],0,jobsize+2);
  kromozom[kromozomsay-1]:=copy(kromozom[k2],0,jobsize+2);
  for i:= uz-1 downto 0 do
  begin
   kromozom[kromozomsay-2,i+b1]:=kromozom[k2,i+b2];
   kromozom[kromozomsay-1,i+b2]:=kromozom[k1,i+b1];
  end;
  kromozomduzelt(kromozomsay-1);
  kromozomduzelt(kromozomsay-2);
  gantbul(kromozomsay-1);
  gantbul(kromozomsay-2);
 end;
end;

procedure Tfr_anaform.sirali;
var
i,j,z,y,k1,k2,b1,b2,uz:integer;
begin
 randomize;
 z:= trunc(kromozomsay*caprazoran);
 if z=0 then z:=1;
 for y:=1 to z do
 begin
  k1:=random(kromozomsay-y*2+2);
  k2:=random(kromozomsay-y*2+2);
  while k2=k1 do
   k2:=random(kromozomsay-y*2+2);
  uz:=random(jobsize-1)+1;
  b1:=random(jobsize-uz);
  b2:=random(jobsize-uz);
  kromozomsay:=kromozomsay+2;
  setlength(kromozom,kromozomsay,jobsize+2);
  kromozom[kromozomsay-2] := copy(kromozom[k1],0,jobsize+2);
  kromozom[kromozomsay-1] := copy(kromozom[k2],0,jobsize+2);;
  for i:= 0 to uz-1 do
  begin
   kromozom[kromozomsay-2,i+b1]:=kromozom[k2,i+b2];
   kromozom[kromozomsay-1,i+b2]:=kromozom[k1,i+b1];
  end;
  for i:= b1-1 downto 0 do
   for j:=b1+uz-1 downto b1 do
    if kromozom[kromozomsay-2,i] = kromozom[k1,j] then
    begin
     kromozom[kromozomsay-2,i]:=jobsize;
```

```
     break;
    end;
   for i:= b2-1 downto 0 do
    for j:=b2+uz-1 downto b2 do
     if kromozom[kromozomsay-1,i] = kromozom[k2,j] then
     begin
      kromozom[kromozomsay-1,i]:=jobsize;
      break;
     end;
   for i:= jobsize-1 downto b1+uz do
    for j:=b1+uz-1 downto b1 do
     if kromozom[kromozomsay-2,i] = kromozom[k1,j] then
     begin
      kromozom[kromozomsay-2,i]:=jobsize;
      break;
     end;
   for i:= jobsize-1 downto b2+uz do
    for j:=b2+uz-1 downto b2 do
     if kromozom[kromozomsay-1,i] = kromozom[k2,j] then
     begin
      kromozom[kromozomsay-1,i]:=jobsize;
      break;
     end;
   kromozomduzelt(kromozomsay-1);
   kromozomduzelt(kromozomsay-2);
   gantbul(kromozomsay-1);
   gantbul(kromozomsay-2);
 end;
end;

procedure Tfr_anaform.tersmutasyon;
var
i,j,z,l,y,k,b,uz:integer;
begin
 randomize;
 z:= trunc(kromozomsay*mutasoran);
 if z=0 then z:=1;
 for y:=1 to z do
 begin
  k:=random(kromozomsay);
  uz:=random(jobsize-2)+2;
  b:=random(jobsize-uz);
  kromozomsay:=kromozomsay+1;
  setlength(kromozom,kromozomsay,jobsize+2);
  kromozom[kromozomsay-1]:=copy(kromozom[k],0,jobsize+2);
  i := b;
  j := b+uz-1;
  for l:= 1 to uz do
  begin
   kromozom[kromozomsay-1,j]:=kromozom[k,i];
```

```
   inc(i);
   dec(j);
  end;
  gantbul(kromozomsay-1);
 end;
end;

procedure Tfr_anaform.komsuikiis;
var
z,y,k,b:integer;
begin
 randomize;
 z:= trunc(kromozomsay*mutasoran);
 if z=0 then z:=1;
 for y:=1 to z do
 begin
  k:=random(kromozomsay);
  b:=random(jobsize-1);
  kromozomsay:=kromozomsay+1;
  setlength(kromozom,kromozomsay,jobsize+2);
  kromozom[kromozomsay-1]:=copy(kromozom[k],0,jobsize+2);
  kromozom[kromozomsay-1,b+1]:=kromozom[k,b];
  kromozom[kromozomsay-1,b]:=kromozom[k,b+1];
  gantbul(kromozomsay-1);
 end;
end;

procedure Tfr_anaform.keyfiikiis;
var
z,y,k,b1,b2:integer;
begin
 randomize;
 z:= trunc(kromozomsay*mutasoran);
 if z=0 then z:=1;
 for y:=1 to z do
 begin
  k:=random(kromozomsay);
  b1:=random(jobsize);
  b2:=random(jobsize);
  while b1=b2 do
   b2 := random(jobsize);
  kromozomsay:=kromozomsay+1;
  setlength(kromozom,kromozomsay,jobsize+2);
  kromozom[kromozomsay-1]:=copy(kromozom[k],0,jobsize+2);
  kromozom[kromozomsay-1,b1]:=kromozom[k,b2];
  kromozom[kromozomsay-1,b2]:=kromozom[k,b1];
  gantbul(kromozomsay-1);
 end;
end;
```

```
procedure Tfr_anaform.keyfiucis;
var
i,j,z,y,k:integer;
b:array[0..2] of integer;
begin
 randomize;
 z:= trunc(kromozomsay*mutasoran);
 if z=0 then z:=1;
 for y:=1 to z do
 begin
  k:=random(kromozomsay);
  b[0]:=random(jobsize);
  b[1]:=random(jobsize);
  while b[0]=b[1] do
   b[1] := random(jobsize);
  b[2]:=random(jobsize);
  while (b[0]=b[2]) or (b[1]=b[2])  do
   b[2] := random(jobsize);
  kromozomsay:=kromozomsay+1;
  setlength(kromozom,kromozomsay,jobsize+2);
  i:=random(2);
  j:=b[i];
  b[i]:=b[i+1];
  b[i+1]:=j;
  i:=random(2)+1;
  j:=b[i];
  b[i]:=b[i-1];
  b[i-1]:=j;
  kromozom[kromozomsay-1]:=copy(kromozom[k],0,jobsize+2);
  kromozom[kromozomsay-1,b[0]]:=kromozom[k,b[1]];
  kromozom[kromozomsay-1,b[1]]:=kromozom[k,b[2]];
  kromozom[kromozomsay-1,b[2]]:=kromozom[k,b[0]];
  gantbul(kromozomsay-1);
 end;
end;

procedure Tfr_anaform.arayasokma;
var
i,j,z,y,k,b1,b2:integer;
begin
 randomize;
 z:= trunc(kromozomsay*mutasoran);
 if z=0 then z:=1;
 for y:=1 to z do
 begin
  k:=random(kromozomsay);
  b1:=random(jobsize);
  b2:=random(jobsize);
  if jobsize<=3 then
   j:=0
```

```
  else
   j:=1;
  while (b2=b1+j) or (b2=b1) or (b2=b1-j) do
   b2 := random(jobsize);
  kromozomsay:=kromozomsay+1;
  setlength(kromozom,kromozomsay,jobsize+2);
  kromozom[kromozomsay-1]:=copy(kromozom[k],0,jobsize+2);
  j:=kromozom[kromozomsay-1,b2];
  if b1<b2 then
   for i:= b2-1 downto b1 do
    kromozom[kromozomsay-1,i+1]:=kromozom[k,i]
  else
   for i:= b2 to b1-1 do
    kromozom[kromozomsay-1,i]:=kromozom[k,i+1];
  kromozom[kromozomsay-1,b1]:=j;
  gantbul(kromozomsay-1);
 end;
end;

procedure Tfr_anaform.chb_tumClick(Sender: TObject);
begin
 rg_ureme.Enabled:= not chb_tum.Checked;
 rg_capraz.Enabled:= not chb_tum.Checked;
 rg_mutas.Enabled:= not chb_tum.Checked;
end;

procedure Tfr_anaform.bt_iptalClick(Sender: TObject);
begin
 cik:= true;
 MessageDlg('Kullancı tarafından iptal edildi.',mtError,[mbOK],0);
end;

procedure Tfr_anaform.anacoz;
var
uremei,uremes,uremea,caprazi,caprazs,capraza,mutasi,mutass,mutasa:extended;
begin
 uremeoran := 0;
 touroran := 0;
 caprazoran := 0;
 mutasoran := 0;
 uremei := strtofloat(tx_uremei.Text);
 uremea := strtofloat(tx_uremea.Text);
 if rg_ureme.ItemIndex = 0 then
  uremes := strtofloat(tx_uremes.Text)
 else
  uremes:=uremei;
 caprazi := strtofloat(tx_caprazi.Text);
 caprazs := strtofloat(tx_caprazs.Text);
 capraza := strtofloat(tx_capraza.Text);
 mutasi := strtofloat(tx_mutasi.Text);
```

```
 mutass := strtofloat(tx_mutass.Text);
 mutasa := strtofloat(tx_mutasa.Text);
{ i:=1;
 if ((uremes-uremei)/uremea) <> 0 then
  i:=trunc((uremes-uremei)/uremea)+1;
 if ((caprazs-caprazi)/capraza) <> 0 then
  i:=trunc(i*((caprazs-caprazi)/capraza+1));
 if ((mutass-mutasi)/mutasa) <> 0 then
  i:=trunc(i*((mutass-mutasi)/mutasa+1));
 i:=i*(strtoint(tx_iteras.text)-1);}
 touroran := strtoint(tx_tour.Text);
 mutasoran := mutasi;
 while mutass >= mutasoran do
 begin
  caprazoran := caprazi;
  while caprazs >= caprazoran do
  begin
   uremeoran := uremei;
   while uremes >= uremeoran do
   begin
     cputime[0]:=time;
     iterasyon:=1;
     baspopulasyon;
     kromozomsay := baspop;
     for iterasyon := 1 to strtoint(tx_iteras.text) do
     begin
      if cik then exit;
      coz;
      quicksort(0,kromozomsay-1);
      SetLength(kromozom,baspop,jobsize+2);
      kromozomsay:= baspop;
      eniyinesilsec;
      cputime[iterasyon]:=time;
      lb_ilerleme.Caption:=inttostr(1+strtoint(lb_ilerleme.Caption));
      lb_ilerleme.Refresh;
     end;
     sonuckaydet;
     application.ProcessMessages;
    uremeoran := uremeoran+uremea
   end;
   caprazoran:=caprazoran+capraza;
  end;
  mutasoran:=mutasoran+mutasa;
 end;
end;

procedure Tfr_anaform.chb_kalanyerdenClick(Sender: TObject);
begin
 chb_tum.Checked := chb_kalanyerden.Checked;
 rg_ureme.Enabled:= not chb_tum.Checked;
```

```
 rg_capraz.Enabled:= not chb_tum.Checked;
 rg_mutas.Enabled:= not chb_tum.Checked;
end;

procedure Tfr_anaform.giriskontrol(Sender: TObject; var Key: Char);
begin
 if Key = '.' then Key:= #44;
 if Key = #44 then exit;
 if Key = #8 then exit;
 if (Key < #48) or (Key > #57) then
 begin
  Key := chr(256);
  Exit;
 end;
end;

procedure Tfr_anaform.bt_klasorClick(Sender: TObject);
var
DirSelected: string;
dosyasay:integer;
sonuc : TSearchRec;
begin
  if SelectDirectory('Select a folder:', 'D:\Delphi', DirSelected) then
  begin
    cik:=false;
    klasorcoz:=true;
    dosyasay := FindFirst(DirSelected+'\*.dat', faAnyFile - faDirectory, sonuc);
    while dosyasay = 0 do
    begin
      if cik then exit;
      dosya.FileName :=DirSelected+'\'+sonuc.Name;
      kaydet.FileName := DirSelected+'\'+ChangeFileExt(sonuc.name,'.csv');
      bt_yukleClick(bt_yukle);
      bt_cozClick(bt_coz);
      dosyasay := FindNext(sonuc);
    end;
    FindClose(sonuc);
    klasorcoz:=false;
    Messagedlg('Klasör çözüm işlemi
tamamlandı.'+#13+DirSelected,mtInformation,[mbOK],0);
  end;
end;

end.
```

Ek 3- 1 10x5 Tipi Neron Ve Carlier Problemleri İçin Analiz Sonucu

Rule set: sonuc.ros

(1-19)	Match	Decision rules
1	360	(F2=3)&(F3=2)=>(F4={1[91],2[148],3[121]})
2	33	(F2=1)&(F3=3)=>(F4={1[33]})
3	120	(F2=2)&(F3=2)=>(F4={1[102],3[16],2[2]})
4	155	(F2=3)&(F3=1)=>(F4={1[26],2[70],3[59]})
5	219	(F2=2)&(F3=1)=>(F4={1[62],2[53],3[104]})
6	80	(F2=2)&(F3=3)=>(F4={1[66],3[7],2[7]})
7	134	(F2=1)&(F3=2)=>(F4={3[68],2[54],1[12]})
8	360	(F1=1)&(F2=3)=>(F4={1[91],2[148],3[121]})
9	33	(F1=2)&(F2=1)=>(F4={1[33]})
10	200	(F1=1)&(F2=2)=>(F4={1[168],3[23],2[9]})
11	155	(F1=2)&(F2=3)=>(F4={1[26],2[70],3[59]})
12	219	(F1=3)&(F2=2)=>(F4={1[62],2[53],3[104]})
13	134	(F1=3)&(F2=1)=>(F4={3[68],2[54],1[12]})
14	480	(F1=1)&(F3=2)=>(F4={1[193],2[150],3[137]})
15	33	(F1=2)&(F3=3)=>(F4={1[33]})
16	155	(F1=2)&(F3=1)=>(F4={1[26],2[70],3[59]})
17	219	(F1=3)&(F3=1)=>(F4={1[62],2[53],3[104]})
18	80	(F1=1)&(F3=3)=>(F4={1[66],3[7],2[7]})
19	134	(F1=3)&(F3=2)=>(F4={3[68],2[54],1[12]})

Results of experiments by train&test method: sonuc.ros

		Predicted					
		1	2	3	No. of obj.	Accuracy	Coverage
Actual	1	201	117	74	392	0.513	1
	2	9	218	107	334	0.653	1
	3	23	180	172	375	0.459	1
	True positive rate	0.86	0.42	0.49			

Total number of tested objects: 1,101
Total accuracy: 0.537
Total coverage: 1

Ek 3- 2 10x10 Tipi Neron Ve Carlier Problemleri İçin Analiz Sonucu

Rule set: sonuc.ros

(1-8)	Match	Decision rules
1	229	(F3=1)=>(F4={2[106],3[123]})
2	215	(F3=3)=>(F4={1[44],2[123],3[48]})
3	407	(F3=2)=>(F4={1[84],3[132],2[191]})
4	71	(F1=2)&(F2=3)=>(F4={2[48],3[23]})
5	417	(F1=1)&(F2=2)=>(F4={1[92],2[215],3[110]})
6	96	(F1=1)&(F2=3)=>(F4={1[24],3[34],2[38]})
7	109	(F1=3)&(F2=1)=>(F4={1[12],2[61],3[36]})
8	158	(F1=3)&(F2=2)=>(F4={3[100],2[58]})

Results of experiments by train&test method: sonuc.ros

		Predicted					
		2	1	3	No. of obj.	Accuracy	Coverage
Actual	2	0	314	106	420	0	1
	1	0	128	0	128	1	1
	3	0	180	123	303	0.406	1
	True positive rate	0	0.21	0.54			

```
Total number of tested objects: 851
Total accuracy: 0.295
Total coverage: 1
```

Ek 3- 3 15x 5 Tipi Neron Ve Carlier Problemleri İçin Analiz Sonucu

Rule set: sonuc.ros

(1-19)	Match	Decision rules
1	74	(F2=2)&(F3=2)=>(F4={2[41],1[7],3[26]})
2	370	(F2=3)&(F3=2)=>(F4={2[195],3[148],1[27]})
3	38	(F2=2)&(F3=1)=>(F4={2[11],1[1],3[26]})
4	72	(F2=1)&(F3=2)=>(F4={1[3],2[23],3[46]})
5	425	(F2=2)&(F3=3)=>(F4={2[158],3[187],1[80]})
6	13	(F2=3)&(F3=1)=>(F4={2[12],3[1]})
7	190	(F2=1)&(F3=3)=>(F4={1[20],2[98],3[72]})
8	444	(F1=1)&(F3=2)=>(F4={2[236],3[174],1[34]})
9	38	(F1=3)&(F3=1)=>(F4={2[11],1[1],3[26]})
10	72	(F1=3)&(F3=2)=>(F4={1[3],2[23],3[46]})
11	425	(F1=1)&(F3=3)=>(F4={2[158],3[187],1[80]})
12	13	(F1=2)&(F3=1)=>(F4={2[12],3[1]})
13	190	(F1=2)&(F3=3)=>(F4={1[20],2[98],3[72]})
14	499	(F1=1)&(F2=2)=>(F4={2[199],1[87],3[213]})
15	370	(F1=1)&(F2=3)=>(F4={2[195],3[148],1[27]})
16	38	(F1=3)&(F2=2)=>(F4={2[11],1[1],3[26]})
17	72	(F1=3)&(F2=1)=>(F4={1[3],2[23],3[46]})
18	13	(F1=2)&(F2=3)=>(F4={2[12],3[1]})
19	190	(F1=2)&(F2=1)=>(F4={1[20],2[98],3[72]})

Results of experiments by train&test method: sonuc.ros

		Predicted					
		2	3	1	No. of obj.	Accuracy	Coverage
Actual	2	305	34	199	538	0.567	1
	3	221	72	213	506	0.142	1
	1	47	4	87	138	0.63	1
	True positive rate	0.53	0.65	0.17			

```
Total number of tested objects: 1,182
Total accuracy: 0.393
Total coverage: 1
```

Ek 3- 4 15x 10 Tipi Neron Ve Carlier Problemleri İçin Analiz Sonucu

Rule set: sonuc.ros

(1-6)	Match	Decision rules
1	261	(F2=2)&(F3=1)=>(F4={2[158],3[42],1[61]})
2	72	(F2=1)&(F3=2)=>(F4={2[35],3[19],1[18]})
3	365	(F2=3)&(F3=2)=>(F4={3[151],2[202],1[12]})
4	1	(F2=2)&(F3=2)=>(F4={2[1]})
5	23	(F2=3)&(F3=1)=>(F4={2[17],3[4],1[2]})
6	40	(F2=2)&(F3=3)=>(F4={3[35],2[5]})

Results of experiments by train&test method: sonuc.ros

		Predicted					
		2	3	1	No. of obj.	Accuracy	Coverage
Actual	2	18	207	193	418	0.043	1
	3	4	186	61	251	0.741	1
	1	2	12	79	93	0.849	1
	True positive rate	0.75	0.46	0.24			

```
Total number of tested objects: 762
Total accuracy: 0.371
Total coverage: 1
```

Ek 3- 5 2 Aşamalı Oğuz Problemleri İçin Analiz Sonucu

Rule set: sonuc.ros

(1-18)	Match	Decision rules
1	387	(F2=3)&(F3=2)=>(F4={1[338],3[25],2[24]})
2	2510	(F2=2)&(F3=1)=>(F4={3[7],2[70],1[2433]})
3	9	(F2=2)&(F3=3)=>(F4={1[4],3[5]})
4	568	(F2=3)&(F3=1)=>(F4={1[245],3[33],2[290]})
5	38	(F2=1)&(F3=3)=>(F4={1[4],3[26],2[8]})
6	36	(F2=1)&(F3=2)=>(F4={1[31],3[3],2[2]})
7	387	(F1=1)&(F2=3)=>(F4={1[338],3[25],2[24]})
8	2510	(F1=3)&(F2=2)=>(F4={3[7],2[70],1[2433]})
9	9	(F1=1)&(F2=2)=>(F4={1[4],3[5]})
10	568	(F1=2)&(F2=3)=>(F4={1[245],3[33],2[290]})
11	38	(F1=2)&(F2=1)=>(F4={1[4],3[26],2[8]})
12	36	(F1=3)&(F2=1)=>(F4={1[31],3[3],2[2]})
13	387	(F1=1)&(F3=2)=>(F4={1[338],3[25],2[24]})
14	2510	(F1=3)&(F3=1)=>(F4={3[7],2[70],1[2433]})
15	9	(F1=1)&(F3=3)=>(F4={1[4],3[5]})
16	568	(F1=2)&(F3=1)=>(F4={1[245],3[33],2[290]})
17	38	(F1=2)&(F3=3)=>(F4={1[4],3[26],2[8]})
18	36	(F1=3)&(F3=2)=>(F4={1[31],3[3],2[2]})

Results of experiments by train&test method: sonuc.ros

		Predicted					
		1	3	2	No. of obj.	Accuracy	Coverage
Actual	1	2,433	377	245	3,055	0.796	1
	3	7	59	33	99	0.596	1
	2	70	34	290	394	0.736	1
	True positive rate	0.97	0.13	0.51			

```
Total number of tested objects: 3,548
Total accuracy: 0.784
Total coverage: 1
```

Ek 3- 6 5 Aşamalı Oğuz Problemleri İçin Analiz Sonucu

Rule set: sonuc.ros

(1-18)	Match	Decision rules
1	2131	(F1=2)&(F2=3)=>(F4={2[1234],3[564],1[333]})
2	2186	(F1=3)&(F2=2)=>(F4={2[864],3[776],1[546]})
3	963	(F1=1)&(F2=3)=>(F4={3[613],2[345],1[5]})
4	223	(F1=2)&(F2=1)=>(F4={2[52],3[106],1[65]})
5	742	(F1=1)&(F2=2)=>(F4={2[168],3[574]})
6	1580	(F1=3)&(F2=1)=>(F4={3[986],1[594]})
7	2131	(F2=3)&(F3=1)=>(F4={2[1234],3[564],1[333]})
8	2186	(F2=2)&(F3=1)=>(F4={2[864],3[776],1[546]})
9	963	(F2=3)&(F3=2)=>(F4={3[613],2[345],1[5]})
10	223	(F2=1)&(F3=3)=>(F4={2[52],3[106],1[65]})
11	742	(F2=2)&(F3=3)=>(F4={2[168],3[574]})
12	1580	(F2=1)&(F3=2)=>(F4={3[986],1[594]})
13	2131	(F1=2)&(F3=1)=>(F4={2[1234],3[564],1[333]})
14	2186	(F1=3)&(F3=1)=>(F4={2[864],3[776],1[546]})
15	963	(F1=1)&(F3=2)=>(F4={3[613],2[345],1[5]})
16	223	(F1=2)&(F3=3)=>(F4={2[52],3[106],1[65]})
17	742	(F1=1)&(F3=3)=>(F4={2[168],3[574]})
18	1580	(F1=3)&(F3=2)=>(F4={3[986],1[594]})

Results of experiments by train&test method: sonuc.ros

		Predicted					
		2	3	1	No. of obj.	Accuracy	Coverage
Actual	2	1,234	513	916	2,663	0.463	1
	3	564	1,187	1,868	3,619	0.328	1
	1	333	5	1,205	1,543	0.781	1
	True positive rate	0.58	0.7	0.3			

Total number of tested objects: 7,825
Total accuracy: 0.463
Total coverage: 1

Ek 3- 7 8 Aşamalı Oğuz Problemleri İçin Analiz Sonucu

Rule set: sonuc.ros

(1-18)	Match	Decision rules
1	284	(F1=2)&(F2=3)=>(F4={2[186],3[67],1[31]})
2	108	(F1=3)&(F2=1)=>(F4={2[21],3[87]})
3	1621	(F1=1)&(F2=3)=>(F4={2[858],3[654],1[109]})
4	4462	(F1=3)&(F2=2)=>(F4={3[161],2[69],1[4232]})
5	194	(F1=1)&(F2=2)=>(F4={3[29],2[165]})
6	10	(F1=2)&(F2=1)=>(F4={3[5],1[5]})
7	284	(F2=3)&(F3=1)=>(F4={2[186],3[67],1[31]})
8	108	(F2=1)&(F3=2)=>(F4={2[21],3[87]})
9	1621	(F2=3)&(F3=2)=>(F4={2[858],3[654],1[109]})
10	4462	(F2=2)&(F3=1)=>(F4={3[161],2[69],1[4232]})
11	194	(F2=2)&(F3=3)=>(F4={3[29],2[165]})
12	10	(F2=1)&(F3=3)=>(F4={3[5],1[5]})
13	284	(F1=2)&(F3=1)=>(F4={2[186],3[67],1[31]})
14	108	(F1=3)&(F3=2)=>(F4={2[21],3[87]})
15	1621	(F1=1)&(F3=2)=>(F4={2[858],3[654],1[109]})
16	4462	(F1=3)&(F3=1)=>(F4={3[161],2[69],1[4232]})
17	194	(F1=1)&(F3=3)=>(F4={3[29],2[165]})
18	10	(F1=2)&(F3=3)=>(F4={3[5],1[5]})

Results of experiments by train&test method: sonuc.ros

		Predicted					
		2	3	1	No. of obj.	Accuracy	Coverage
Actual	2	1,209	21	69	1,299	0.931	1
	3	750	92	161	1,003	0.092	1
	1	140	5	4,232	4,377	0.967	1
	True positive rate	0.58	0.78	0.95			

Total number of tested objects: 6,679
Total accuracy: 0.828
Total coverage: 1