**T.C.**
**BAHÇEŞEHİR ÜNİVERSİTESİ**

# GENETİK ALGORİTMA VE KARINCA KOLONİ ALGORİTMASI İLE MELEZ MODEL OLUŞTURULARAK ERP SİSTEMLERİNDE KAPASİTE PLANLAMASININ YAPILMASI

**Yüksek Lisans Tezi**

**CELAL BİLGİN**

**T.C.**
**BAHÇEŞEHİR ÜNİVERSİTESİ**

**FEN BİLİMLERİ ENSTİTÜSÜ**
**BİLGİ TEKNOLOJİLERİ**

# GENETİK ALGORİTMA VE KARINCA KOLONİ ALGORİTMASI İLE MELEZ MODEL OLUŞTURULARAK ERP SİSTEMLERİNDE KAPASİTE PLANLAMASININ YAPILMASI

**Yüksek Lisans Tezi**

**CELAL BİLGİN**

**Tez Danışmanı: Doç. Dr. Adem KARAHOCA**

**T.C.**
**BAHÇEŞEHİR ÜNİVERSİTESİ**

**FEN BİLİMLERİ ENSTİTÜSÜ**

**BİLGİ TEKNOLOJİLERİ**

Tezin Adı: Genetik algoritma ve karınca koloni algoritması ile melez model oluşturularak erp sistemlerinde kapasite planlamasının yapılması

Öğrencinin Adı Soyadı: Celal BİLGİN
Tez Savunma Tarihi:

Bu tezin Yüksek Lisans tezi olarak gerekli şartları yerine getirmiş olduğu Fen Bilimleri Enstitüsü tarafından onaylanmıştır.

Doç. Dr. M. Tunç BOZBURA
Enstitü Müdürü
İmza

Bu tezin Yüksek Lisans tezi olarak gerekli şartları yerine getirmiş olduğunu onaylarım.

Y. Doç. Dr. Alper TUNGA
Program Koordinatörü
İmza

Bu Tez tarafımızca okunmuş, nitelik ve içerik açısından bir Yüksek Lisans tezi olarak yeterli görülmüş ve kabul edilmiştir.

Jüri Üyeleri — İmzalar

Tez Danışmanı — ----------
Doç. Dr. Adem KARAHOCA

Üye — ----------
Y. Doç. Dr. Alper TUNGA

Üye — ----------
Yrd. Doç. Dr. Yalçın ÇEKİÇ

# ÖZET

GENETİK ALGORİTMA VE KARINCA KOLONİ ALGORİTMASI İLE MELEZ MODEL OLUŞTURULARAK ERP SİSTEMLERİNDE KAPASİTE PLANLAMASININ YAPILMASI

Celal Bilgin

BİLGİ TEKNOLOJİLERİ – İŞ ZEKASI

Tez Danışmanı: Doç. Dr. Adem KARAHOCA

01/2013

Günümüzde firmalar tüm süreçlerini kontrol altına alıp, iş zekası uygulamaları ile otonom kontroller oluşturmak istemektedir. Bu noktada sistemin en temelinde vazgeçilmez olan erp programlarıdır. Erp programları verilerin oluştuğu ve saklandığı en tabandaki sistemdir. İş zekası uygulamaları oluşan verileri yorumlayıp belirlenen kararları vermesi için sistemi tamamlayan bir parçadır.

Üretim yapan firmalar, üretim hattını doğru planlayabilmesi için, maliyet başta olmak üzere verimlilik, siparişlerin zamanında yetişmesi gibi birçok kısıtı sağlanmaları gerekmektedir. Kapasite planlama; içerisinde operasyonlar arası gecikmeler, makinaların boşta kalması, işlerin üretim hatlarına doğru dağıtılması, siparişlerin zamanında hazır olması, kurulum süreleri, aktivite maliyetleri, stokta bulundurma maliyeti ve benzeri bir çok maliyet kaynağının doğru dağıtılması ve/veya kullanılması gibi karmaşık bir yapı bulunduran bir optimizasyon problemidir.

Genetik algoritmaları ve karınca koloni algoritmaları ile yapılmış kapasite planlama problemlerinin çözümleri literatürde mevcuttur. Genetik algoritmalar bir çözüm popülasyonu oluşturmak ve bu popülasyonu her jenerasyonda daha iyiye götürme konusunda bir altyapı sağlarken, karınca koloni algoritması sıralama problemlerinin çözümünde bir alt yapı sağlamaktadır.

Bu tezin amacı firmalarda kapasite planlama problemini çözerken genetik algoritma ve karınca koloni algoritmasından melez bir model oluşturup, amaç fonksiyonu tanımlarına göre bir çözüm kümesi oluşturmaktır. Çözüm testleri için Java 7.0 kullanılarak bir program geliştirilmiştir.

Anahtar Kelimeler: ERP, genetik algoritma, karınca koloni algoritması, Kapasite planlama, optimizasyon.

## ABSTRACT

SCHEDULING WITH CREATING A HYBRID MODEL OF GENETIC ALGOROTIHM AND ANT COLONY AT ERP SYSTEMS

Celal Bilgin

INFORMATION TECHNOLOGIES – BUSINESS INTELLIGENCE

Thesis Supervisor: Assoc. Prof. Dr. Adem KARAHOCA

01/2013

Todays companies demand to take control all processes and build autonomous systems using business intelligence applications. The ERP programs are the essential frameworks to meet these demands. ERP systems integrate internal and external management information across an organization, while business intelligence process and evaluate these collected data.

In order to schedule a production line, the companies need to provide constraints such as costing, efficiency and preparing orders on time. Scheduling is an optimization problem including complex systems like tardiness, idle times, distributing operations to production line, activity costs, set-up times and earliness stocks.

There are solutions about scheduling reported in literature using genetic algorithms and ant colony optimization. While genetic algorithms generate a solution population and improve it on each generation, ant colony algorithms are based on the solving of ordering problems.

The aim of this study is to design a hybrid model of genetic and ant colony algorithms in solving schedule problems of companies by generating a set of solutions for aim functions. The proposed model was coded by using Java 7.0 programming language.



Keywords: ERP, genetic algorithm, ant colony, schedule ,optimization.

İÇİNDEKİLER

## TABLOLAR

## ŞEKİLLER

## KISALTMALAR

| | | |
|---|---|---|
| MRP | : | Malzeme ihtiyaç planlaması |
| MRP II | : | Üretim kaynakları planlaması |
| ERP | : | Kurumsal kaynak planlaması |
| C-commerce | : | Ortak ticaret |
| SCM | : | Tedarik zinciri yönetimi |
| APS | : | İleri planlama ve çizelgeleme |
| MES | : | Üretim yürütme sistemi |
| MMAS | : | Min-max karınca sistemi |
| MR | : | Mutasyon oranı |
| MDR | : | Mutasyon azaltma oranı |
| rho | : | feromon artırım oranı |
| Grho | : | Global feromon artırım oranı |

# 1. GİRİŞ

Günümüz firmaları teknolojiye adapte olarak tüm işlerini bilgisayar programları üzerinden yürütmekte ve karar destek aşamalarında bu yazılımlardan destek alarak insan hatasını en aza indirmeye çalışmaktadır. Bazı kararları verebilmek için sadece birkaç bilgi değil sistemin tümünden gelen bilgileri değerlendirmek gerekir. Bunu yapabilmek içinde bu bilgileri ilişkileri ile birlikte detaylı bir şekilde saklamakla iş başlıyor.

Enterprise Resource Planning - Kurumsal Kaynak Planlaması (ERP) , firmaların A'dan Z'ye tüm süreçlerini içerisine almakta ve bunlar arasındaki ilişkileri yönetmektedir. İçerisinde birçok alt sistemi ve karar destek mekanizmasını barındırmaktadır. Departmanlar ve bölümler arasındaki veri akışlarını ve iletişimi kontrollü hale getirmektedir. İçerisinde bizi en çok ilgilendiren modüller ise MRP II ve kapasite planlamadır.

Kapasite planlama; amaç fonksiyonunda çok fazla parametre olan karmaşık bir problemdir. Yöneylem araştırmaları; amaç fonksiyonu olan ve bu amaca ulaşmak için belirli kısıtları göz önünde bulundurduğumuz bir problem çözme yöntemidir. Günümüz bilgisayarlarının hız ve etkinliğinden faydalanarak bu problemleri daha hızlı çözebilmek için çeşitli yöntemler geliştirilmiştir.

Genetic Algorithm - Genetik Algoritma, canlıların doğal seleksiyonu ve mutasyonuna dayalı olarak optimizasyon problemlerini çözmek için çözüm kümesinde aramalar yapan algoritmadır. Kapasite planlama, optimizasyon problemi gibi düşündüğümüzde GA 'nın çözüm yöntemi olan ihtiyacımızı ortaya koymaktadır.

Ant Colony Algorithm - Karınca kolonisi Algoritması (ACO) , meta-sezgisel (meta-heuristic) bir algoritmadır. Karıncaların yemek arayışında en kısa yolu nasıl bulduklarını formüle ederek gerçek hayat problemlerinde kullanmamız sağlanmıştır. Özellikle sıralama problemlerinde çok iyi oranda başarı gösterdiği için bu yöntem seçildi.

Bu tezde amaç olarak kapasite planlaması yapılırken GA ve ACO ile hibrid bir model oluşturarak problemi çözmektir. Kapasite planlamasında işi yapacak olan iş merkezleri veya kaynakları genetik algoritma üzerinde bulunurken , karınca koloni algoritması da

planlanması gereken operasyonları sıralama yöntemi ile iş merkezlerine yerleştirmektedir.

Tez içeriğinde öncelikle ERP'ye kısa bir giriş yapılacak ve devamından kapasite planlama problemlerinin tanımlanması yapılacaktır. Çözümde kullandığımız yapay zeka yöntemleri arasından nasıl seçim yaptık ve seçtiğimiz yöntemler hakkında bilgiler verilecektir. Hibrid modelin hangi kurallar dahilinde oluşturulduğu ve çalışma prensibi anlatıldıktan sonra yazılan programa bilgi girişleri ve programın çalışması hakkında bilgiler verilecektir. Son olarak örnek problemler ve çıktıları paylaşılacaktır.

# 2. LİTERATÜR TARAMASI

## 2.1. ERP

Kurumsal Kaynak Planlaması – Enterprise Resource Planning (ERP) veri ve işlemleri entegre etmeye çalışan sistemlerdir. Veriler bir veya birden fazla veri tabanında saklanır. ERP sistemleri veri tabanındaki bilgileri departmanlar arasında ve iş fonksiyonları içerisinde oluşturur, paylaşır ve veri döngüsünü gerçekleştirir. ERP, organizasyonlar içerisinde bilgi teknolojileri çözümleri olarak en fazla adaptasyonu sağlanan sistemlerdir ( Al-Masharia,Al-Mudimigha, Zairib, 2003).

ERP'nin tarihçesine kısaca baktığımız zaman; başlangıçta sadece stokların yönetildiği PIC programları bulunmaktadır. Daha sonra stokları doğru yönetebilmek için Malzeme İhtiyaç Planlaması (Material Resource Planning-MRP) 1960 yıllarında ortaya çıkmıştır. İşletmelerin malzeme ihtiyaçlarını planlamak ve üretim planlarını çıkarmak amacıyla oluşturulmuştur. MRP ile hammadde ve diğer malzeme ihtiyaçları planlaması yapıldıktan sonra firma içinde üretim planlarını ve işleri, kurallar dahilinde daha iyi yönetebilmek için Üretim Kaynakları Planlaması (Manufacturing Resource Planning – MRP II ) ortaya çıkmıştır. 1980'lerin başında yazılım firmaları ortaya belirli bir framework üzerinde çalışan modüler sistemler geliştirmeye başladı. Bu modüller kendi içinde haberleşen büyük sistemin küçük sistem parçaları olarak işleyen programlardı. 1990 yıllarında firmalar üretimin haricinde diğer departmanlarında sisteme entegre olması ihtiyacını doğurduğunda ERP sistemleri ortaya çıktı. İnsan kaynakları, kapasite planlaması, müşteri ilişkileri yönetimi gibi modüller ERP içerisinde ve diğer modüllere entegre olarak sunuldu (Jacobs ve Weston, 2007 ).

ERP sistemlerinin geleceğine baktığımız zaman ise artık yapay sinir ağları , karar destek mekanizmaları, veri madenciliği ve sezgisel yöntemler gibi ileri düzey fonksiyonları içinde barındıran ve kullanıcılara alternatif bilgileri üreten sistemler haline gelmektedir.

Birçok işletme mevcut ERP uyarlaması sonucunda ortaya çıkan yeniden yapılandırılmış iş fonksiyonlarına adapte olmaya çalışırken diğer işletmeler mevcut ERP sistemlerini yükseltmeye ve uygulama alanını genişletmeye çalışmaktadır (Beatty ve Williams, 2006). Gartner Research yayınladığı bir raporda mevcut ERP sistemlerinin öldüğünü yerini yenilikçi ve standart oluşturacak yeni nesil ERP II sistemlerine

bırakacağını yayınlamıştır. Tedarik zinciri yönetimini ve e-ticareti içinde barındıracak olan ERP II, ortak-ticareti (collaborative-commerce - c-commerce) beraberinde getirmiştir. Böylece işletmeler yatay ve dikey olmak üzere birbirlerine doğrudan bağlantılı hale gelmiştir. Müşteri İlişkileri Yönetimi (Customer Relationship Management CRM) de bu yeni modüllere entegre olduğunda işletmeler müşterileri ile anlık bilgi paylaşımı platformuna erişmiş oluyorlar (F.D.Ted & Weston, Jr. , 2003) .

### 2.1.1. ERP'NİN YENİ İHTİYAÇLARI

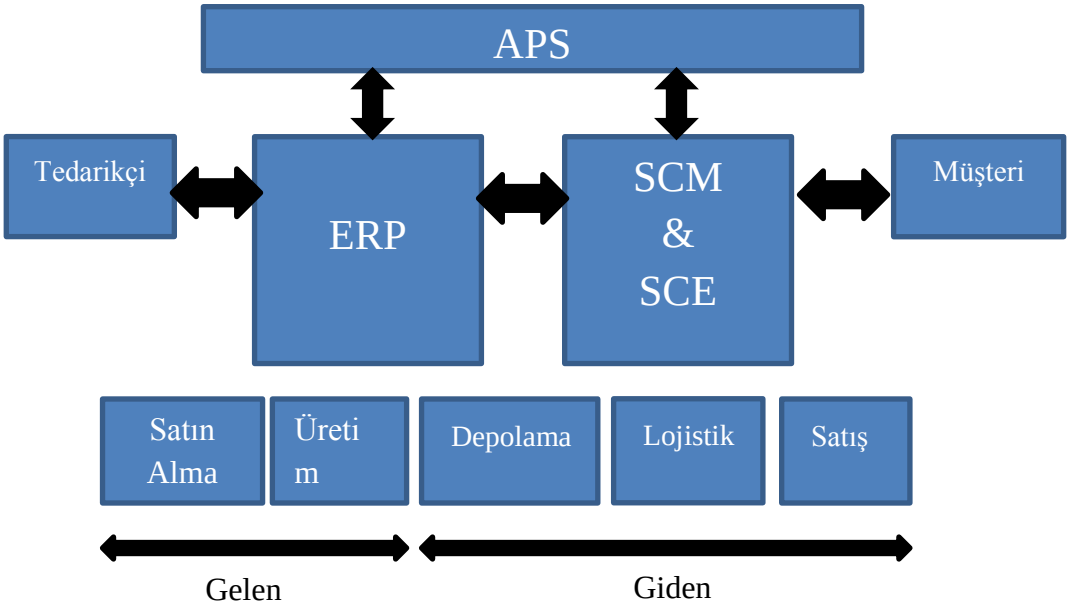
SCM kendi başına ERP II ile büyük bir ihtiyaç listesi çıkarmaktadır. SCM öncelikle mevcut iş sisteminizi tedarik zincirine entegre etmenizi gerektirir. İş dinamiğinde yapılacak hatalar SCM ile birleştiğinde kamçı etkisini (Bullwhip effect) ciddi anlamda arttıracaktır (Moller, 2005) .

SCM, hammaddeden başlayarak müşteriye gönderilen ürüne kadar planlama ve kontrollerin yapılması gerekliliğini sunun bir sistem çözümüdür. Bu sebeple tedarikçi ve müşteri ilişkilerini yukarı ve aşağı doğru daha az maliyetle yönetme stratejisini ortaya koyar.

Yazılım endüstrisi bu ihtiyaçları giderebilmek için ERP üzerine Gelişmiş planlama ve Zamanlama (Advanced Planning and Scheduling - APS) modülü geliştirmiştir. Bu modül, matematiksel programlamalar ve genetik algoritmaların arıtılmış bir çözümü olarak tedarik zinciri ağının içindeki parçaları birleştirmek için aslında SCM'in merkezine oturtulmuştur. APS, MRP/II ve kapasite planlamadan aldığı bilgileri gerçek zamanlı yorumlayarak tedarik zinciri merkez kontrolünün işini kolaylaştırmaktadır. Böylece kapasite planlama sistemin olmazsa olmazı durumuna gelmektedir.

Ayrıca APS sistemlerinin ERP ve MES arasındaki entegrasyonu sağladığı yöntemlerde mevcuttur (Botta-Genoulaz, Millet ve Grabot, 2005 ) .Bu sayede üretimden anlık olarak veriler çekilmekte ve daha doğru kararlar verilmektedir. Buda SCM için daha doğru terminler demektir.

**Şekil 2.1 Tedarik Zinciri**




*Kaynak:* Advanced scheduling problem using constraint programming tecniques in SCM environment , Y.Yun , M.Gen ,2002

İşletmelerin ERP II ile ortak-ticaret alanı oluşturmaları ve APS kullanmaları, doğru ve toleransları yüksek olan bir kapasite planlama yapmak zorunda bırakmaktadır. Halihazırda işletmeler mevcut ERP sistemleri içinde kapasite planlamalarını dahi yapmakta zorlanmaktadır. Çünkü göz önünde bulundurulması gereken çok fazla faktör bulunmaktadır. Genel itibari ile baktığımızda kapasite planlaması çok fazla kısıttı olan karmaşık bir optimizasyon problemidir. Ortak-ticaretin gelişmesiyle sürekli olarak sisteme sipariş bilgilerinin girişleri olacaktır . Müşteriler platform üzerinde siparişleri girmeden önce teslim zamanlarına bakmak istemektedir. Hatta buna göre tedarikçi tercih etmektedirler. Bu sistem beraberinde hızlı ve detaylı, tolere edilmiş teslim süreleri ortaya çıkarma ihtiyacını doğurmuştur. Bunu yapabilmek içinde aynı özelliklerde kapasite planlamasının yapılması ihtiyacı doğmuştur. Statik kararlar veren bazı kapasite planlama metodları farklı durumları göz önünde bulunduramayacağı için işletmeye en çok faydayı sağlama kuralını pas geçebilir. Ancak sezgisel yöntemler bunu aşmada kullanılabilir.

## 2.2. ÜRETİM

İşletmeler ürettikleri ürüne veya tercih ettikleri üretim yöntemine bağlı olarak farklı üretim stratejileri uygularlar. Üretim basit olarak ; hammaddelerin işlenip satışa hazır hale getirilmesidir. Gerçek yaşamda işler bu kadar kolay olmamaktadır. Firmaların bazı tercihler yapması gerekmektedir. Genel olarak üretim tipleri aşağıda belirtilmiştir.

### 2.2.1. SİPARİŞ İÇİN ÜRETİM YAPMA (MAKE-TO-ORDER (MTO) )

Genel olarak müşterilerin verdiği siparişler birbirinden farklı özelliklere sahiptir. Bu sebeple müşteri siparişini verdiği anda planlamaya alınır. Sipariş gelene kadar hangi materyallere ihtiyaç olduğu belirlenemez. Bu sebeple buradaki başarı anahtarı ihtiyaçların zamanında gelmesi ve üretim hattını buna göre planlamasıdır. Kapasite planlamasının gereksinimler belirlendikten sonra hemen yapılması gerekir. Planlamada yapılacak hatalar veya gecikmeler üretimin performansını düşürecektir. MTO sistemlerinde adaptasyonu ve entegrasyonu iyi sağlanmış bir kapasite planlama maliyet değerlerini de düşürecektir (Choy ve diğ. 2011) .

### 2.2.2. STOK İÇİN ÜRETİM YAPMA (MAKE-TO-STOCK (MTS) )

İtme yöntemi olarak gösterilen bu modelde işletme stoka üretim yapar. Daha sonra satışlar bu stoklar üzerinden gerçekleştirilir. Bu tip organizasyonlarda kapasiteyi ve planlamayı yönetmek bir MTO sistemine göre biraz daha kolaydır. Talepler ve ne gibi kaynaklara ihtiyaç olduğu bellidir. Ancak bu sistemleri yönetmesindeki zorluk, parti büyüklüğü yöntemleri ve çok sayıda planı aynı anda planlamaktır. Bu denli büyük problemleri çözmek için zaman ve yöntemlerinde doğru seçilmesi gerekir.

### 2.2.3. SİPARİŞE ÜRETME (BUILD-TO-ORDER (BTO) )

Basitçe bakıldığında yalın üretim (lean manufacturing) ve çevik üretimin (agile manufacturing) ana karakteristiklerini barındırır. Aslında MTO (sipariş odaklı) ve MTS (tahmin odaklı) stratejilerinin ikisini de barındırır. Standart üretilen ve müşteri taleplerine göre değişmeyecek olan komponentlerin kısa dönemli tahminler yoluyla, müşteri tercihlerine göre değişen parçaların talep alındıktan sonra, nihai ürününde bu sebeple talep edildikten sonra üretilmesi yöntemine dayanır. Bu sistemin yürüyebilmesini sağlayan ana faktör tedarik zinciridir. Yapılan tahminler ve gelen

taleplere göre, müşteriye sunulan çıktıdan satın alınan veya üretilen kaynaklara kadar detaylı ve güçlü bir akışın olması gerekir. Çünkü pazarda sağlanacak performansı etkileyen en önemli faktör tedarik zinciridir (Yimer ve Demirli, 2010) .

## 2.3. KAPASİTE PLANLAMA (SCHEDULING )

Kapasite planlama üretim yapan işletmeler için üretkenlik ve verimlilik açısından önemlidir. İşletmeler için kapasite planlama yapmak hangi operasyonu, ne zaman, hangi makine ve/veya işçiler ile hangi ekipmanları kullanarak gerçekleştireceğine karar vermek ve en düşük maliyet ile en kısa sürede gerçekleştirmektir. Aynı zamanda verimliliği yükseltip maliyeti düşürmeye de odaklanır.

Kapasite planlama temel anlamda ikiye ayrılır Akış tipi – FSP (Flow Shop Problem) ve Atölye tipi - JSP (Jobshop Problem ) (Lin ve Jia-zhen, 2009) . Ayrıca Tek Makine ( single-machine) problemleri olarak bir dal daha eklenebilir (Choy ve diğ. 2011) .

### 2.3.1. ÜRETİM PLANLAMA YÖNTEMLERİ

#### 2.3.1.1. AKIŞ TİPİ ÜRETİM PLANLAMA (FLOW SHOP SCHEDULING)

Akış tipi üretim planlama problemi verilen n adet aynı sıralı işin verilen süreleri ile m adet makinaya planlanmasıdır (Akhshabia ve diğ. 2012) . Her iş sırası önceden belirlenmiş m adet işten oluşur ve her makinada bir defa işlem görür. Her işlem, işlem süresince kesintiye uğramadan gerçekleşir. Aralardaki taşıma süreleri ihmal edildiği varsayılır. Bir makine aynı anda birden fazla işlemi gerçekleştiremez ve bir işlem aynı anda sadece bir makinada işlem görür. Tüm işler için makine sıraları aynıdır ve hazırlık süreleri önceden bilinmektedir (Temiz, 2010 ).

Ayrıca pazardaki rekabet ortamından dolayı birçok işletme, klasik üretim sistemini daha yeni melez akış tipi üretim sistemlerine yükseltmektedir. Aradaki fark ise darboğaz oluşan noktalarda işi yapan paralel makineler koymaktır. Böylece bu iş noktalarında işlemler paralel makine sayısı kadar azalacaktır (Amin-Naseri ve Beheshti-Nia, 2007 ) .

### 2.3.1.2. ATÖLYE TİPİ ÜRETİM (JOB SHOP SCHEDULING)

Bu tip üretimde önceden belirlenmiş sıraları belirli bir dizi operasyon geçici olarak oluşturulur. Ürünler çeşitlilik gösterir ve hepsi aynı rotaya sahip değildir. Mevcut üretim sisteminde bu operasyonlar kaynak kısıtları göz önünde bulundurularak planlaması yapılır. Operasyonlar yerleştirilirken aynı işi yapan benzer operasyon merkezlerine farklı süreler ile yüklenebilir. Buna Esnek Atölye tipi problem denir ( Flexible Job Shop Problem) (Mesghouni ve diğ. 2004) .

Bu tip problemlerde n tane iş m adet farklı makinada işlenecektir. Her iş o kadar operasyondan oluşur ve hangi makinada ne kadar süreceği belirlenmiştir. Bu sebeple çok fazla iş ve makine üzerinde kısıtlar vardır. Problemin ana amaç fonksiyonlarından birisi toplam süreci ve her iş için gereken süreyi kısaltmaktır (Qing-dao-er-ji ve Wang, 2011).

### 2.3.2. KAPASİTE PLANLAMA PARAMETRELERİ - TERİMLERİ

Kapasite planlama optimizasyonunun içinde çok sayıda kısıt ve birden fazla amaç fonksiyonu bulunur. Bu formülleri daha anlamlı hale getirmek için fonksiyonlar üzerinde hesaplanmış bazı parametreler bulunmaktadır.

*Uygunluk*; belirlenen zaman aralığında mevcut kaynağın birim zamanda ne kadar kullanılabilir olduğunu gösterir. Uygunluk değerlerine göre yapılmamış planlama ihtimal dahilinde değildir[16].

*Takvim*; planlamanın kaynakların uygunluk dağılımını gösteren bir komponentidir. Her kaynak kendine ait bir takvimi vardır ve genel takvimden kendine etki eden faktörlerde vardır. (genel tatiller, kutlamalar gibi.)

*Olası en erken başlama;* kaynakların uygunluğuna göre belirlenen bir işe başlanabilecek olan en erken tarih. Bunun için takvim bağlı kısıtlar, operasyon bağımlı kısıtlar, makina bağımlı kısıtlar ve işçi bağımlı kısıtlar olabilir.

*Uygulama Penceresi ;* bir işin olası en erken başlama zamanı ile en geç bitiş zamanı arasındaki periyodu verir. Aynı zamanda işin ne kadar esnetilebileceğini gösterir (Cox, 2005 ) .

*Uygulanabilir Plan*; her işin, yapılabilmesi için uygun işçi veya makine, belirlenen aralıklarda takvime uygun kaynaklar, bağımlı olduğu tüm operasyonların bitmiş olması kısıtlarını sağlamasıdır. Mümkün plan kısıttı diğer birçok kısıttı içinde bulundurur ve cezai şartı olmamalıdır (Cox, 2005 ).

*En Geç Bitiş;* bir işin olası en son bitiş tarihidir. Bu tarih dışarıdan bir kısıt olarak verilebilir veya program içinde de hesaplanabilir. Eğer program içinde hesaplanıyorsa işin mümkün plan dahilinde olması beklenir ancak dışarıdan sabit olarak veriliyorsa ve iş bu tarihe kadar planlanmaz ise uygulanabilir değildir.

*Ağ;* tüm işlerin önceliklerinin ve bağlantılarının tanımlandığı yerdir. Düğümler ile birbirine bağlanmış bir grafik teorisidir (Cox, 2005 ).

*Öncelik Bağlantısı ;* iki düğüm arasındaki zaman kısıt ağıdır. Eğer B işi , A işi bitmeden başlayamaz ise bu iki iş arasında bir öncelik bağlantısı tanımlanır. Bu durumda A işi ata , B işi de ardıldır (Cox, 2005 ).

*Kaynak;* satın alınan ve operasyon içerisinde kullanılan hammadde veya yarı mamuldür. Olası çözümler kaynak ihtiyaçlarına dikkat etmek zorundadır. ERP sistemlerinde bu bilgi MRP çalışmasından alınır.

*Toplam Üretim Süresi(Makespan Time);* planlama işleminin en erken başlama tarihi ile en geç biten işin bitiş tarihi arasında geçen süredir. Genelde kapasite planlama problemleri bu süreyi minimize etmeye çalışır.

*Toplam Operasyon Süreleri;* planlanan tüm operasyonların sürelerinin toplamıdır. Amaç fonksiyonunda minimize edilir. Atama yapılan işlerin en kısa hangi makine ve/veya işçi tarafından yapılması gerektiğini amaçlanmaktadır.

*Gecikme;* operasyonun olası en erken başlama tarihinden ne kadar geç başladığı arasındaki süre farkıdır. İki taraflı etkisi vardır, bekleme maliyeti ve diğer işlere öncelik verilmesi gerektiği.

*Ortalama Gecikme Süresi;* gecikme süresinin mevcut talebin tüm operasyonlarının sürelerinin toplamına olan oranıdır.

*Boşta Kalma Süresi;* makine veya işçilerin iki operasyon arasındaki atıl süresidir. Başlangıç için atanacak hiçbir iş olmadığından ilk operasyonun beklenmesi ve son operasyondan sonra olası iş olmadığı için bu aralardaki boşta kalma süresi dahil edilmez.

*Fason Maliyeti;* mevcut işletme kapasitesinin yetersiz gelmesi veya belirlenmiş işlemlerin dışarıda yapılması durumlarında kapasiteye eklenmesi gereken maliyet değeridir. Sonsuz kapasiteli kabul edilebilir veya olası kapasite tanımı yapılabilir.

*Erken bitirme;* talebin belirlenen tarihten önce arz edilmesi durumunda ortaya çıkacak stokta bekletme süresi ve maliyetini içerir.

*Planlanma Yüzdesi;* planlama sonucunda uygulanabilir planlamaların, tüm planlanacak olan işlere oranlanmasıdır.

## 2.4. GENETİK ALGORİTMALAR

John H. Holland tarafından 1975 yılındaki "*Adaptation in Natural and Artificial Systems: An Introductory Analysiswith Applications to Biology, Control, and Artificial Intelligence"* adlı kitabında bahsettiği genetik algoritma, doğadaki evrimleşmeyi ve seleksiyonu temel alan sezgisel bir arama tekniğidir. Holland, doğadaki olayların sitemlere adapte edilmesi konusunu araştırmıştır.

Bu sezgisel yöntem optimizasyon ve çözüm kümesi arayışlarında çokça kullanılır. Daha büyük sınıf olan evrimsel algoritmanın bir alt sınıfıdır.

Evrimsel algoritmalar kompleks optimizasyon problemlerini çözmek için doğadaki seleksiyonu taklit eder. Olası çözüm kümesini içinde barındıran büyük bir popülasyondan oluşur. Büyük boyutlu ve karmaşık olan yavaş, zorlu, kırılgan veya mevcut analitik teknikler ile formüle edilmesi zor problemleri çözmek için genetik algoritmalar ve evrimsel programlamayı yapay zeka sistemlerinin adaptasyon davranışlarına, çoklu kriter ve amaç fonksiyonları üzerinde sistematik alternatif çözüm tarayıcısı olarak kullanılmasını doğurmuştur (Cox, 2005 ).

Genetik Algoritmalar kalıtım, mutasyon, seleksiyon ve çaprazlama gibi doğadan ilham alınan tekniklerini kullanarak verilen optimizasyon probleminin çözüm kümesini oluşturmaya odaklanır.

Genetik algoritmanın işleyişine geçmeden önce kısa bir terminoloji tanımı yapılmalıdır.

Alel; gen üzerinde belirli bir yerde veri taşıyan yapıya denir. Binary sunumlarda bu 0-1 olur. Problem çözümlerine göre sayılardan veya objelerden oluşabilir.

Kromozom (Birey); problemin olası çözümünü içinde barındıran alelden oluşmuş yapıdır. Kromozomun üzerindeki alel sırası problem çözümlerinde önemlidir. Bunu locuslar belirler.

Genom; kromozomların belirli bir sunum şeklidir. Genelde genom bir kromozomdan oluşur ancak çoklu kısıtların ve amaç fonksiyonlarının olduğu problemlerde birden fazla kromozomdan oluşabilir (Cox, 2005 ).

Locus; bir kromozomun genom üzerindeki belirli bir pozisyondur.

Popülasyon; yapay aramanın yapıldığı tüm genomların bulunduğu topluluktur. Genetik algoritmada yapılacak olan bütün işlemler popülasyon üzerinde yapılır ve popülasyona ekleme çıkarmalar yapılarak seleksiyon sağlanır.

Tavlama; genetik algoritmada popülasyonun mevcut durumunu değiştirerek çözüme doğru yaklaşması işlemidir. Genetik algoritmalarda bu işlem için mutasyon, yeni popülasyon oluşturma ve seleksiyon işlemleri ile gelecek nesiller çözüme doğru yaklaştırılır (Cox, 2005 ).

Uygulanabilir Çözüm; genetik algoritmalar mutlaka karşılığı mümkün olan çözümler bulmalıdır. Aksi halde en iyi fit değeri verse bile kullanılamaz. Çözümün mümkün olup olmadığını kısıt fonksiyonları belirler. Kısıt fonksiyonları içerisinde mevcut kaynaklar ve operasyonlar arasındaki ilişkiler bulunmaktadır.

Amaç Fonksiyonu; bireyin hedeflenen çözüme ne kadar yakın olduğunu bulmak için kullanılan ölçme yöntemidir. Tek bir fonksiyondan oluşacağı gibi kolektif amaç fonksiyonu da olabilir. Genetik algoritmalar keşif odaklıdır. Bu keşifte sonuca ne kadar yakınsadığı amaç fonksiyonu ile kontrol edilir.

Jenerasyon; genetik algoritmanın oluşturduğu her yeni popülasyon bir jenerasyon olarak adlandırılır. Her jenerasyonda sistem yeni adaylar üretir ve bu işlem sonlandırma kısıntına kadar devam eder. Maksimum jenerasyon sayısı da ayrıca parametre olarak verilir.

Sonlandırma Kısıt; genetik algoritmanın uygun çözüm aramayı ne zaman sonlandırması gerektiğini belirten kısıttır. Maksimum jenerasyon sayısının kısıt olarak verileceği gibi başarı olarak kabul edilebilecek fit değeri veya algoritmanın çalışma süreside kısıt olarak verilebilir. Ayrıca bu kısıtların hepsinin verilip herhangi biri sağlandığında durması da sağlanabilir (Cox, 2005 ).

Genetik Algoritmanın çalışma prensibi temel olarak;

1- İlk popülasyonu oluştur
2- Fit değerleri hesapla
3- Doğal seleksiyonu yap
4- Çaprazlama ve mutasyon yap
5- Onarım fonksiyonlarını çalıştır.
6- Fit değerlerini hesapla
7- Sonlandırma kriteri sağlanıyor mu kontrol et, eğer değilse 3. aşamaya geri dön

Genetik algoritma; oluştur ve test et paradigmasının bir formudur. Bu metot çok fazla olası çözümü oluşturup her birinin kabul edilebilir çözümler olup olmadığını araştırır. Doğadaki evrimi taklit ederek karmaşık optimizasyon problemleri için yeni çözümler ortaya çıkarmaya çalışır. Bunu yaparken her seferinde bir önceki jenerasyondan daha iyi sonuçlar ortaya çıkarmak için genetik operatörler kullanır.

### 2.4.1. GEN TASARIMI

Genetik Algoritmanın verimliliğini ve gücünü kromozomun tasarımı belirler. Çünkü problemin nasıl tanımlanacağı ve çözüme nasıl ulaşılacağı kısmı bu aşamada yapılır. Hatalı tasarımlar problemin çözüm kümesinin bulunamamasına neden olabilir.

Pelin Alkan ve Hüseyin Başlıgil (2009) yaptıkları planlama çözümünde n işleri, m makine sayısını belirtmek koşuluyla n X m boyutunda bir matris tanımlayarak x(i,j) {0,1} şeklinde bir kısıt eklemişlerdir . X. Cai ve K.N. Li (1997) farklı becerilerdeki

işçilerin planlanması probleminin çözümü için oluşturdukları metotta makine ve işleri ikili modda karşılıkları çıkarılmış ve genom tasarımı bu ikili kodları barındıracak şekilde tasarlamışlardır. Alebachew D. Yimer ve Kudret Demirli (2010) iki aşamalı tedarik zinciri problemini çözmek için önerdikleri sistemde tamsayı, ondalıklı sayı ve ikili kod barındıran bir genotip kullanmışlardır. Byung Joo Park, Hyung Rim Choi ve Hyun Soo Kim (2003) atölye tipi planlama problemlerini çözerken oluşturdukları melez modelde iş sayısı j ile makine sayısı m çarpımları uzunluğunda bir gen tanımı yapmışlardır. Bu yöntemde sıralama önemlidir ve kromozom üzerinde ki sıralamaya göre hangi işin hangi makinaya atandığı bulunur. Li Lin ve Huo Jia-zhen (2009) çelik boru üretimindeki planlama problemlerinin çözümü için önerdikleri yöntemde kromozom yapısını iki yarı bölüm gibi düşünerek ilk bölümde işlerin sırasını, ikinci bölümde ise hangi makinada yapılacağı bilgisini saklayacak şekilde tasarlamışlardır. Mostafa Akhshabi, Javad Haddadniab ve Mohammad Akhshabi (2012) akış tipi planlama problemlerini çözebilmek için tasarladıkları paralel genetik algoritmada kromozom içerisinde her işe bir id vererek yerleştirilmiş, kromozomun içerisinde paralel yürütebilmek için işlemcilere ayırabilmeyi sağlayan -1 değeri konulmuştur.

### 2.4.2. BAŞLANGIÇ POPÜLASYONU

Genetik algoritmalarda çözüm kümesinin aranacağı popülasyonun ilk olarak oluşturulması gerekir. İlk popülasyon genelde rassal yöntemler ile oluşturulur. Buradaki önemli olan nokta popülasyonun sayısıdır. Eğer popülasyon sayısı küçük olursa genetik algoritma araştırması lokal minimum veya lokal maksimumlara takılabilir. Eğer popülasyon sayısı gereğinden büyük olursa çok fazla işlem yapmak gerekeceğinden çözümün bulunması için gereken süre uzayacaktır.

Manas Kumar Maiti (2008) kredi bağlantılı promosyon taleplerinin stok modeli ile planlanması için kullandığı değişken popülasyon sayısı gibi algoritmalarda kullanılabilir. Bu yöntemde eğer maksimum fit değeri ile ortalama fit değeri arasındaki fark verilen parametredeki değerden daha düşükse yeni bir jenerasyon üretilmiştir. Ayrıca yaşlı olan bireyler ölüp yerine popülasyon büyüklüğünü korumak için yeni çocuklar oluşturulmuştur.

### 2.4.3. EVRİM ( GENETİK OPERATÖRLER)

Popülasyonun optimal çözüme yakınsaması için uygulanan bir dizi operasyondur. Bunun içinde seleksiyon, çaprazlama, mutasyon ve onarım operatörleri bulunur. Bu sayede her yeni jenerasyonda popülasyona yeni bireyler katılır, yaşı dolan veya evrim sonucu amaçtan uzaklaşan fit değere sahip birey popülasyondan atılır.

#### 2.4.3.1. ÇAPRAZLAMA

Genetik algoritmalarda yeni popülasyonun oluşturulması için iki bireyden yeni bir birey oluşturulur ve buna çaprazlama denir. Buradaki amaç popülasyondaki çeşitliliği arttırıp yeni çözümler oluşturabilmektir. Literatürde tek noktalı, çok noktalı, değişken nokta sayılı, rastlantısal, tersine çevirme gibi çok çeşitli çaprazlama teknikleri bulunmaktadır. Çaprazlama operatörü probleme göre belirlemek daha doğrudur. Aksi takdirde onarım operatörüne çok iş düşmektedir.

Amaç fonksiyonu birden fazla optimizasyon içeriyorsa çaprazlamada buna bağlı olan kriterleri göz önünde bulundurarak yeni birey üretilebilir. Amaç fonksiyonlarının birisi süreyi minimize etmek isterken diğeri maliyetleri düşürmek istiyorsa, süreyi minimize eden en iyileri bulup maliyeti minimize eden en iyi diğer bireyleri bulup bunlardan yeni bireyler türetmek her iki amaç fonksiyonunu optimale doğru yaklaştıracaktır (Alcan ve Başlıgıl, 2009 ).

Çok basit bir şekilde 1-0 rassallığına dayanarak 1 ise 1. bireyden 0 ise 2. bireyden alelin kopyalanması ile yeni birey oluşturulabilir (Cai ve Li, 1997 ).

Ortalama çaprazlama yöntemi ile iki aile bireyinin alel değerlerinin ortalaması alınarak yeni bir birey oluşturulabilir. Konveks çaprazlamada ise [0.05,0.45] aralığında rastgele değer seçilerek ikinci bireye ağırlıklandırma yapılır. Çoğunlukla ilk bireyden alel alınır (Yimer ve Demirli, 2010).

#### 2.4.3.2. MUTASYON

Mutasyon operatörü, belirlenen oran doğrultusunda aleller üzerinde değişiklikler yaparak bizi yakınsayamadığımız çözümlere götürmek için kullanılır. Önemli olan mutasyon oranıdır. Çünkü mutasyon oranı çok düşük olduğunda gerekli optimal sonuca

sıçramalar yapamazsınız, benzer şekilde yükse olduğunda da optimal sonuçtan kolaylıkla uzaklaşabilirsiniz. Genetik algoritmaların mutasyon oranını belirlemek için çeşitli bulanık mantıklar geliştirilmiştir. Basit olarak eğer ortalama fit değerin altında değerler oluşuyorsa mutasyon oranı arttırılır, ortalama değerin üzerinde ise mutasyon oranı düşürülür.

Kromozomun içindeki belirlenen orana göre alel atıp sonra olmayan değerleri rastgele tekrar koyarak yeni birey oluşturma yöntemleri de geliştirilmiştir (Cai ve Li, 1997).

Çevirme mutasyon operatörü ise boş bir genom alıp içini ana bireyden belirlenen orana göre verilerin yerlerini değiştirerek yeni bireyler oluşturmaktadır. Değiş-tokuş çaprazlama yöntemi ana bireyin birebir kopyasını alıp belirlenen oranda rassal olarak iki alel seçer ve bunların yerini değiştirir. Kombine mutasyon ağırlıklandırma metodu ile hangi alellerin mutasyona uğrayıp uğramayacağına karar verir. Sınır mutasyon ise rassal olarak seçilen kromozomların belirli locuslardaki verilerin en düşük ve en yüksek değerlerini alarak yeni bireye rassal olarak yükler. Bunun amacı çözüm kümesinin sınırlarına ulaşarak çeşitliliği sağlamaktır (Yimer ve Demirli,2010).

Mutasyon operasyonu çaprazlama gibi probleme göre belirlenmelidir. Eğer kromozom içerisinde genomun belirli bir bölgesiyle veya kimlik bilgisini taşıyan bölgeleri üzerinde mutasyon yapmak ardından onarım operatörünün çalışmasını gerektirir. Çünkü bu şekildeki değişiklikler genelde uygun olmayan çözümler doğurur.

#### 2.4.3.3. ONARIM

Çaprazlama ve mutasyon operasyonları bizi optimal çözüme doğru bazen sıçrama yaparak yakınsatır. Ancak bu operatörler iyi kurgulanmadığında çok fazla gen bozulmasına neden olmaktadır. Onarım bu şekilde bozulan kromozomları tamir etmeye odaklanır.

Bazen problemin doğası gereği bozulması ve düzeltilmesi gerektiği durumlarda oluşabilir. Kusum Deep ve Manoj Thakur (2007) önerdikleri mutasyon operasyonu ile daha az onarım operatörü çalışmasını sağlamışlardır .

Kapasite planlama problemlerinde genom tasarımında işleri makinelere veya işçilere yükleme şeklinde bir yapı varsa ve mutasyon atamalara yapıldığında önerilen yöntem

bazı işçileri veya makineleri boşta tutmak ve mutasyon sonrası bozulmaları kısıtlamalara dikkat ederek bu boştaki iş merkezlerine yükleme yapmak önerilen yöntemler arasındadır (Cai ve Li, 1997).

Kapasite planlama problemlerinde operasyonların birbirlerine bağımlılık durumları göz önünde bulundurulmalı ve bozulmalar onarılmalıdır. Makinelerin işleri yapıp yapamayacağı kısıtlarına göre onarım yapılmalıdır. Bir işi aynı anda iki makinaya atama veya aynı makinaya aynı anda iki farklı işi yükleme hataları da onarılmalıdır (Choy ve diğ., 2011).

#### 2.4.3.4. SELEKSİYON

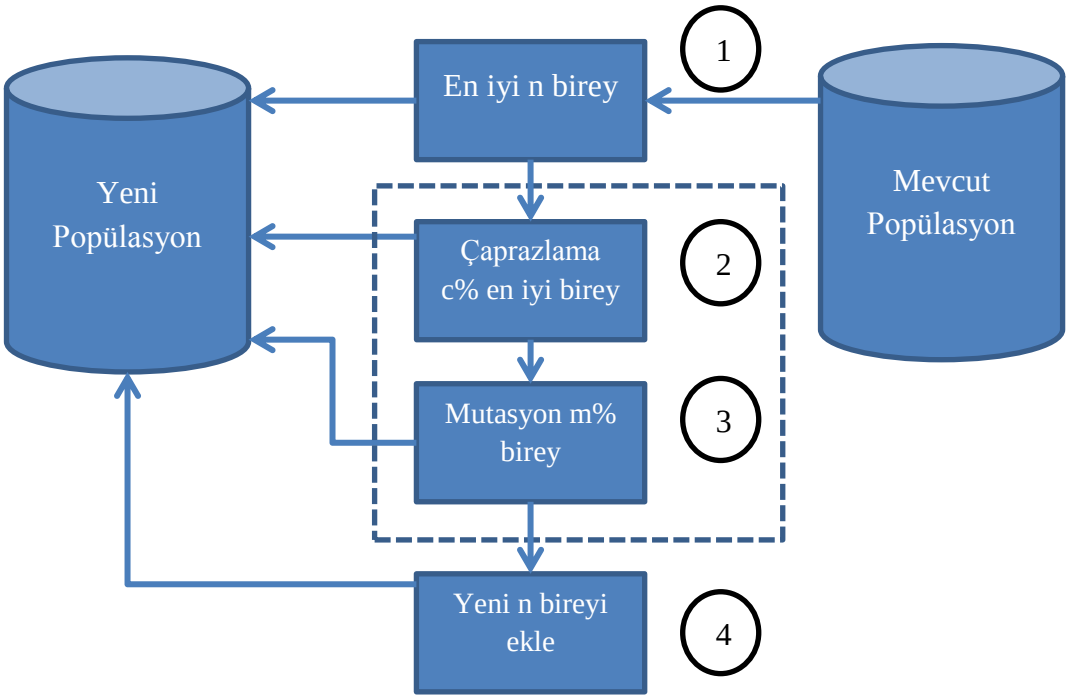
Her jenerasyonda bir önceki nesilden gelen bireylerin tamamı aktarılmaz. Çeşitliliği sağlamak için bazı bireylerin yok olması ve yerine yeni bireylerin oluşturulması gerekir. En çok kabul gören yöntemlerin başında elitizm gelir. Popülasyonda belirlenen orandaki en iyi bireylerin yeni jenerasyona doğrudan alınması yöntemine dayanır. Böylece küçük oranda mutasyonlar ile iyi olan bireyler daha iyi hale getirilebilir ve iyi çözümler popülasyonda saklanmış olur. Rulet çarkı ve elitizmin birlikte kullanıldığı yöntemlerde önerilmiştir (Yimer ve Demirli,2010).

Yarışma seçim yönteminde rastgele iki farklı birey seçilir, hangisinin fit değeri amaç fonksiyonunda daha iyi ise yeni popülasyona o aktarılır. Ayrıca en iyi bireyler aktarıldıktan sonra geri kalanların rastgele aktarıldığı tohum seçim yöntemi de mevcuttur (Park ve diğ. 2003 ).

Değişken popülasyon sayısı çözümlerinde yeni gelen bireylerin hepsinin popülasyona dahil edilmediği, bunlar arasında sıralama yöntemi ile seçimler yapılarak popülasyona dahil edildiği öneriler mevcuttur (Maiti, 2008 ).

En düşük ve en yüksek fit değerlerine sahip bireyleri popülasyon içinde tutmak, yapılan araştırmalara göre çeşitliliği arttırıyor fakat değişkenliğini algoritma içinde kontrol etmek gerekmektedir (Elamin, 2006 ).

**Şekil 2.2 Yeni popülasyonun oluşturulması**




(Cox, 2005 )

## 2.5. KARINCA KOLONİ ALGORİTMASI

Doğada topluluk halinde yaşayıp karşılaştıkları problemleri birlikte çözen hayvanlar sosyal hayvanlar olarak adlandırılır. Bu çeşitli canlıların birbirleri ile iletişim kurup problem çözme teknikleri incelenerek gerçek dünya problemlerinin çözümlerinde kullanılmaktadır.

Karıncalar bu şekilde yaşamlarını sürdüren sosyal hayvanlardır. Karıncalar görme yetenekleri tam olarak yoktur. Ancak buna rağmen yiyecek ile yuva arasında en kısa yolu her zaman bulabilmektedirler. Hatta çevre koşullarında değişiklik olduğunda ve belirledikleri yol artık en kısa yol değilse yeni kısa yolu da bulabilmektedirler.

Karıncalar belirlenen yoldan geçerken feromon adında bir koku bırakır. Bu koku belirli bir süre kalır ve yavaşça yok olur. Doğal olarak kısa olan yolda bu koku daha fazla

kalmaktadır. Yolun ayrıldığı noktalarda karıncalar feromon maddesinin yoğunluğuna bakar ve bunu göz önünde bulundurarak ağırlıklandırma yaptıktan sonra rassal bir karar verir. Bunun sebebi hepsinin aynı yolda gitmesini engellemek ve eğer keşfedilmemiş daha kısa bir yol var ise onu bulmaktır (Elamin, 2006 ).

Çalışma prensibi gereği paralelizmi destekler buda daha düşük maliyet ve zamandan kazanç sağlar. Problem karmaşıklaşmaya başladığında optimal çözüme ulaşmak bazen uzayabilir ancak bu durum genel olarak optimizasyon problemlerinin yakınsamalarında karşılaşılır. Paralel işlemciler ile karınca koloni algoritmasının daha hızlı çözümüne dair araştırmalar bulunmaktadır (Ling ve diğ. 2009 ).

Olası yolun geçtiği tüm noktaların birbirlerine göre feromon değerlerinin tutulduğu bir feromon matrisi vardır. Bu değerler karıncalar tarafından bırakılır ve buharlaşama oranıyla düşürülür.

Karınca koloni optimizasyonunun çalışma prensibi;

1- başlangıç durumunu oluşturmak
2- her karınca belirlenmiş veya rassal olarak belirlenen bir başlangıç noktasına gelir
3- feromon matrisine göre ağırlıklandırma yapıp yol belirler
4- lokal feromon matrisini günceller
5- fit değerleri hesaplanır
6- en iyi fit değerine göre global feromon matrisi güncellenir
7- sonlandırma kriterine ulaşılamadıysa 2. aşamaya geri dönülür

2 den başlayarak 5. Aşamada dahil olmak üzere bu işlemler her karınca için yapılır.

Gezgin satıcı problemlerinin çözümü için iyi bir yöntemdir. Gezgin satıcı problemi; n adet şehrin hepsini en kısa yolun bulunarak ziyaret edilmesidir. Karınca koloni algoritması özellikle sıralamaya dayalı problemlerin çözümünde etkilidir.

Yapılan araştırmalara göre iki düğüm arasında görüle bilirlik kuralları tanımlaması yapıldığında algoritma optimal sonuca daha hızlı yaklaştığı kanıtlanmıştır. Amaç fonksiyonuna göre bir düğümden diğerine geçişte, hangisinin fit değerini amaç

fonksiyonuna uygun olarak daha fazla yaklaştıracağını bulup ağırlıklandırma üzerine bir faktör daha eklenir. Amaç fonksiyonunu ters yönde etkileyen düğümlerin seçim şansı vardır, ama diğerlerine göre daha düşüktür (R.F. Tavares Neto n, M.Godinho Filho, 2011 ).

Karınca koloni algoritmasının sıralama üzerine getirdiği çözüm planlama problemlerinin çözümüne uyarlanmaktadır. Tek makine üzerine işlerin sıralanması konusunda yapılmış çözümler bulunmaktadır. Önerilen yöntemde j işi alınır ve i sırasında nasıl olacağına karar verilir. Feromon matrisinden yola çıkarak bir faktör belirlenir ve problemin başında belirtilen α kuvveti alınır. Amaç fonksiyonuna ne kadar uygun olduğuna karar verilir ve bununda β kuvveti alınır. Böylece arama uzayında amaç fonksiyonuna uygun sıralamanın yapılması için olasılık arttırılır (Merkle ve Middendorf, 2000).

İşlerin farklı makinalar üzerine yüklenmesi ve sıralanması problemlerinin çözümünde yapılan araştırmada makinalar için bir süper düğüm oluşturulmuş ve karıncaların öncelikle burayı ziyaret etmesi için algoritma düzenlenmiştir. Normal düğümlere, süper düğümler üzerinden dağılan karıncalar işlerin hangi makinalarda yapılacağına bu şekilde karar vermektedir (T. Keskinturk, M. B.Yildirim, M. Barut, 2010). Ayrıca global feromon güncellemesi yapılarak çözüme diğer karıncalarında yakınsaması sağlanmıştır (Zhuo ve diğ. 2007).

Gezgin satıcı problemlerinde yapay karıncalara doğada olmayan ancak araştırmalar sonucu davranışlarını iyileştirmek için bir sonraki kısa yolu ve daha önce ziyaret edilip edilmediğinin bilgisini vermemizi sağlayan bazı yeni özellikler eklenmiştir. Dorigo ve Gambardella önerdikleri Yapay Karınca Sisteminde (Ant Colony System- ACS) ziyaret edilen noktaların olasılığını 0 olarak hesaplamayı ve daha kısa olan bir sonraki noktanın uzaklık değerlerinin belli bir katsayı ile çarpılarak olasılığını arttırmaya yönelik bir görünürlük formülü uygulamıştır. Bu yöntem ile ziyaret edilen noktaların olasılıktan kaldırılmasını ve kısa olan bir sonraki noktanın olasılığının yükselmesi sağlanmıştır (Dorigo ve Gambardella, 1996 ) .

M-MMAS algoritması mevcut karınca koloni algoritmasının farklı bir yorumudur. Mevcut karınca koloni algoritmasının lokal aramalarını biraz daha genişleten, seçilen

rotanın ağırlıklarını toplayarak oluşturan bir planlama yorumudur. Araştırmada görünürlük fonksiyonun değişmesinin problem çözümündeki hızının kıyaslanması yapılmıştır (Rajendran ve Ziegler, 2001).

J. Heinonen ve F. Pettersson atölye tipi planlamada önerdikleri yöntem; mevcut karınca koloni algoritması çalıştıktan sonra işlem sonrası algoritması geliştirmişlerdir. Bu algoritma ile karmaşık süre hesaplamaları bu işlem üzerinde yapılmıştır. İşlem sonrası algoritma tamamlandıktan sonra fit değerleri hesaplanır ve değerlendirilir. Böylece karınca koloni algoritması karmaşık süre bazlı hareket etmediği için problemin optimal sonuca ulaşmasında hız kazanılır (Heinonen ve Pettersson, 2007).

# 3. VERİ VE YÖNTEM

Kapasite planlama problemi sadece atölye veya akış tipi üretimden oluşmamaktadır. Benzer şekilde sadece üretim hattı veya hücresel imalattan oluşmaz. Gerçek dünya problemlerine baktığımız zaman birçok sistemin birleşmesi ile üretim yapılmaktadır. Üretim sistemlerinin yalnızca birisini ele alıp çözmek istediğimizde problemin karmaşıklığı ortaya çıkmaktadır. Bunların iç içe çalıştığı kompleks bir sistem gerçek dünya problemlerini çözmede bize yardımcı olabilir. Zaten kapasite planlamanın karmaşık ve zor oluş nedeni buradan kaynaklanmaktadır. Literatürde kapasite planlama NP-Zor olarak tanımlanmıştır. NP, belirsiz Turing Makinası ile çok terimli zamanda çözülebilen karar problemlerini içeren karmaşıklık sınıfıdır. NP-Zor, en az NP problemleri kadar zor olan problemlerin bulunduğu sınıfa denir (Garey ve Johnson, 1979 ).

Kapasite planlama problemlerini basit anlamda incelediğimizde, işi yapan merkezlerin (işçi veya makine ) üzerine yapılacak işleri hangi sırayla yüklememiz gerektiğini bulmaktır. Sıralama yapılırken MRP sisteminden gelen veriler göz önünde bulundurulmalıdır. MRP verilerinin içerisinde operasyonun gerçekleşmesi için gerekli olan hammadde ve diğer operasyonlar arasındaki ilişkisi bulunmaktadır. MRP den gelen veriler ile kapasite planlamanın ağ, öncelik bağlantısı ve kaynak bilgileri oluşturulur.

Problem çözümü için önerilen yöntem ise Genetik Algoritma ile Karınca Koloni algoritmasının birleşimi ile yeni bir melez model oluşturmaktır. Bu yeni modelin amacı genetik algoritma ve karınca koloni optimizasyonunun temellerini kullanarak kapasite planlama problemlerinin çözümüne uygun bir yapı oluşturabilmektir.

## 3.1. VERİ

Önerilen yöntem Java 7 üzerinde yazılmıştır. Problem proje üzerinde 3 ana paket oluşturulmuştur. BaseUtils paketi üzerinde problem tanımında kullanılacak temel yapılar yer almaktadır. Solver paketi, yapacakları işleme göre ayrılmış sınıfları barındırmaktadır. Main paketi ise problemi tanımlayıp çözümü için sistemin tetiklendiği yerdir. BaseUtils paketi bu noktada hangi verileri sakladığı anlatılacaktır. Solver paketi için detaylı bilgi bulgular bölümünde yer almaktadır.

### 3.1.1. MAKİNE

Kapasite planlamasının yapılacağı iş merkezleri program üzerinde Machine sınıfı içerisinde saklanmaktadır. Hangi operasyonların bu iş merkezinde yapılabilme durumu varsa bir liste içinde tutulmaktadır. Bu makine için kurulum tiplerinin arasındaki geçişte kullanılmak üzere bir matris bulunmaktadır. Problem tanımlanırken bu matrisin verilmesi gerekmektedir. Ayrıca makineler arasında daha kolay iletişim sağlamak için kimlik numarası ve isim belirtilebilir. Ayrıca makinelerin fason veya işletme içi makineler olup olmadığı önemlidir. Hesaplamalar yapılırken fason iş merkezleri kapasitesi sonsuz kabul edilebildiği durumlar vardır. Operasyonlar arasındaki geçişlerde makine boşta kalma sürelerini hesaplamaktadır ve bunu belirlenen maliyet ile çarpmaktadır. Mutasyon oranı ve feremon matrisinide barındırmaktadır.

### 3.1.2. OPERASYONLAR

Operasyon sınıfı içinde makine üzerindeki planlama işlemine göre ne kadar zaman alacağına dair tüm hesaplamaların yapılması için gerekli olan birçok değişken tanımlanmıştır. Maliyete etki eden parametreleri de barındırmaktadır. Ayrıca üzerinde planlama durumunu belirten statüler bulunmaktadır. Bazı durumlarda mevcut operasyonlar devam edebilir ve kapasite planlamanın bunu göz önünde bulundurması gerekir. Tamamlanma yüzdesi ve mevcut devam eden operasyon olup olmadığı bilgisi belirtilir. Böylece ilk operasyonlar daima devam eden operasyonlar olmaktadır.

### 3.1.3. İLİŞKİLER VE KAYNAKLAR

Planlama yapılırken bir operasyonun ne zaman başlayabileceği bilgisine gerekli kaynakların en geç hazır olduğu zaman bulunarak ulaşılır. Bir operasyonun gerçekleşmesi için başka bir operasyonun bitmiş olması gerekebilir. MRP bize bu temel bilgileri sunmaktadır. MRP verilerinin saklanması için kaynakların tutulduğu bir yapı oluşturulmuştur. Bu yapıda operasyonlar arası ilişkiler, hammadde ihtiyaçları, alternatif operasyonlar bilgisi ve sipariş teslim zamanları gibi bilgiler bulunmaktadır.

Ayrıca planlanabilecek olan operasyonların bilgilerinin tutulduğu planlanabilir sınıfı oluşturulmuştur. Operasyonun en erken başlama süresini, hangi makine üzerinde ve hangi sırada olduğu bilgileri saklanmaktadır.

### 3.1.4. TEST VERİLERİ

Tezin araştırmasında kullanılmak üzere 2 farklı ürün temel alınarak örnek hazırlanmıştır. Bu ürünlerin operasyonlarının süreleri ve sıralaması Tablo 3.1 ve Tablo 3.2' de verilmiştir. A ürün iki, B ürün ise üç operasyondan oluşmaktadır.

**Tablo 3.1 A ürünü rotası**

| A Operasyon Adı | Süre(Saat) | İş Merkezi | Hazırlık Tipi |
|---|---|---|---|
| Operasyon 1 | 7 | M1,M2,M3,M4 | A1 |
| Operasyon 2 | 12 | M1,M2,M3,M4 | A2 |

**Tablo 3.2 B ürünü rotası**

| Operasyon Adı | Süre(Saat) | İş Merkezi | Hazırlık Tipi |
|---|---|---|---|
| Operasyon 1 | 7 | M1,M2,M3,M4 | B1 |
| Operasyon 2 | 12 | M1,M2,M3,M4 | B2 |
| Operasyon 3 | 8 | M1,M2,M3,M4 | B3 |

İş merkezlerinin hepsinin kullanılabilir olması karmaşıklık seviyesini arttırmak amaçlıdır. Operasyon alternatifleri olarak her operasyonun iş merkezlerine yüklenişi ilişkiler objesinde belirtilmiştir. Her operasyonun kendine ait bir ana operasyon kodu bulunmakta ve alternatiflerin kendine ait ayrıca bir operasyon kodu bulunmaktadır. Her alternatif tanımlaması iş merkezleri üzerine bir operasyon eklemek demektir. Örnek olarak B ürünü için tanımlama yapılırken sisteme olası 12 operasyon yüklenmektedir. Hesaplamalar yapılırken 12 operasyonda dikkate alınmaktadır. Sipariş sayısında 1 adet eklememiz demek 12 olası operasyon daha eklediğimiz anlamına gelir.

Problem boyutlarının değişmesi ile algoritmaların nasıl tepki verdiği bilgisine erişebilmek için üç farklı veri seti hazırlanmıştır. Bu veri setlerinde ürünler belirlenmiş ve teslim tarihleri ayarlanmıştır. Test parametrelerinden ilki olan planlama başlangıç tarihi; 2013.01.01 00:00:00 olarak ayarlanmıştır. X, Y ve Z olarak adlandırılan setlerin detayları Tablo 3.3, Tablo 3.4 ve Tablo 3.5'de verilmiştir.

**Tablo 3.3 1. sipariş listesi**

| Ürün | Teslim Tarihi |
|---|---|
| B | 2013.01.02 06:00:00 |
| B | 2013.01.03 12:00:00 |
| A | 2013.01.04 08:00:00 |
| A | 2013.01.05 04:00:00 |

**Tablo 3.4 2. sipariş listesi**

| Ürün | Teslim Tarihi |
|---|---|
| B | 2013.01.02 06:00:00 |
| B | 2013.01.03 12:00:00 |
| A | 2013.01.04 08:00:00 |
| A | 2013.01.05 04:00:00 |
| B | 2013.01.06 10:00:00 |
| A | 2013.01.07 06:00:00 |
| B | 2013.01.08 12:00:00 |
| A | 2013.01.09 18:00:00 |
| A | 2013.01.10 14:00:00 |

**Tablo 3.5 3. sipariş listesi**

| Ürün | Teslim Tarihi |
|---|---|
| B | 2013.01.02 06:00:00 |
| B | 2013.01.03 12:00:00 |
| A | 2013.01.04 08:00:00 |
| A | 2013.01.05 04:00:00 |
| B | 2013.01.06 10:00:00 |
| A | 2013.01.07 06:00:00 |
| B | 2013.01.08 12:00:00 |
| A | 2013.01.09 18:00:00 |
| A | 2013.01.10 14:00:00 |
| B | 2013.01.11 10:00:00 |
| A | 2013.01.12 06:00:00 |
| B | 2013.01.13 02:00:00 |

Amaç fonksiyonu üzerinde sistem tasarımına dayalı olarak birçok parametre verilebilir. Kapasite problemleri çözümünde iki tipte sayısal tabanlı skor oluşturulabilir. Bunların ilk süre tabanlı, ikincisi ise maliyet tabanlı. Her operasyonun ve iş merkezinin üzerinde aktivitelerin maliyetleri belirlenirse sistem fit değeri hesaplamasını yaparken maliyet değerlerini de çıkartıp amaç fonksiyonuna kullanıma hazır hale getirecektir, aksi durumda 0 olarak hesaplanmaktadır. Testler yapılırken amaç fonksiyonu temelinde süreler kullanılmıştır.

TotalFitness= (ScheduledPercentage)*
(Earliness - 5*DiffJobEndDelivery - 2*FlowTime - 2*IdleTimeTotal) ;

Earliness:teslimden ne kadar önce bitirdiği

DiffJobEndDelivery: Teslime ne kadar geciktiği

FlowTime: Toplam çevrim süresi

IdleTimeTotal: atıl zaman

**Denklem 3.1 Amaç fonksiyonu**

Buradaki öncelikli amaç Earliness ile işlerin teslimden önce bitmesini sağlamaktır. İkinci olarak DiffJobEndDelivery yüksek bir negatif katsayı verilerek önceliğin siparişin yetişmesi ilkesi doğrultusunda skor hesaplamaya yönlendirilmiştir. Çevrim süresine negatif katsayı verilmesindeki sebep işlerin tüm iş merkezlerine dağılmasını sağlamaktır. IdleTimeTotal ile operasyonlar arasında boş süreleri yani makinelerin boşta beklemelerini azaltmaya yönlendirmek için negatif katsayı ile amaç fonksiyonuna eklenmiştir. ScheduledPercentage; sisteme yüklenen toplam operasyonların planlanan operasyonlar ile olan oranını veriyor. Amaç fonksiyonu ana çarpanı olarak yazılmasındaki sebep alınacak en iyi puanın tüm operasyonlar çizelgelendiğinde ortaya çıkmasını sağlamaktır. SetupTime amaç fonksiyonuna özellikle eklenmemiştir. Çünkü kurulum süreleri operasyon sürelerine etki edecek ve dolayısıyla toplam çevrim süresini uzatacaktır. Amaç fonksiyonunda çevrim süresi negatif faktörde olduğu için algoritma kurulum süresini azaltmaya çalışacaktır.

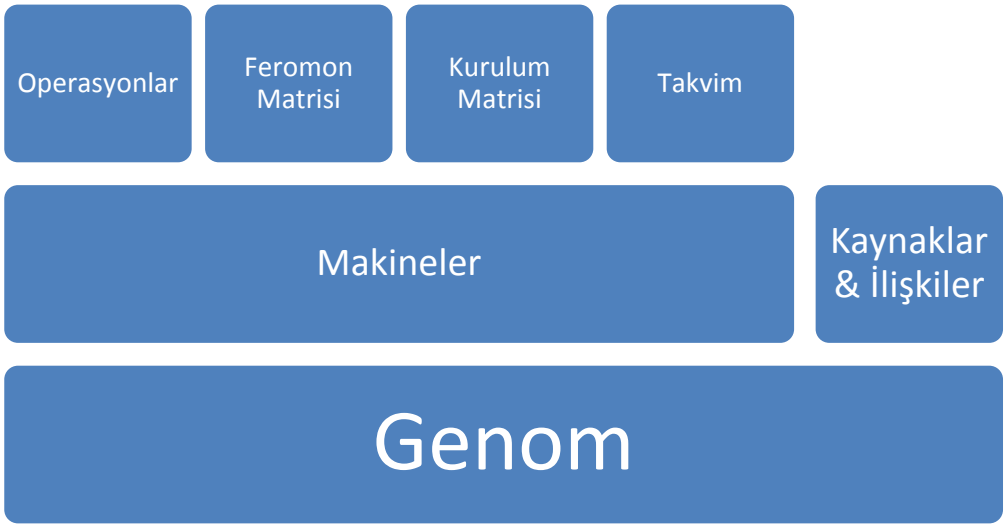
## 3.2.YÖNTEM

Genetik algoritma yöntemimizin temelini oluşturmaktadır. Genetik operatörlerden mutasyon içerisine karınca koloni optimizasyon algoritması entegre edilerek melez model oluşturulmuştur.

### 3.2.1. PROBLEMİN KODLANMASI

Kapasite planlamasının uygulanacağı her makine bir alel olarak ele alınmıştır. Genom ise bu makinelerin toplamlarından oluşmaktadır. Problemde tanımlanan tüm makineler bir genom üzerinde bulunmaktadır. Problem çözümündeki yöntemden ötürü makinelerin sırası önemli değildir. Her bir makine içerisinde planlanması muhtemel olan operasyonları barındırır. Ayrıca kaynaklar genel bir ilişki verisi olduğu için genom bazında belirtilir. Makinelerin hepsi kendine özgü feromon matrisi barındırır.

**Şekil 3.1 Genom tasarımı**




Feromon matrisinde bir operasyondan sonra hangi operasyonun gelmesi gerektiğinin feromon değerini tutmaktadır. Kurulum matrisi ise operasyonların kurulumları arasında geçecek olan hazırlık sürelerini tutmaktadır. Hazırlık tipi n olacak şekilde nXn matrisinde bir yapıya sahiptir.

**Tablo 3.6 Kurulum Matrisi**

| | A1 | A2 | B1 | B2 | B3 |
|---|---|---|---|---|---|
| A1 | 0 | 3 | 2 | 1 | 1 |
| A2 | 2 | 0 | 3 | 0 | 3 |
| B1 | 1 | 1 | 0 | 2 | 2 |
| B2 | 3 | 2 | 5 | 0 | 3 |
| B3 | 2 | 3 | 4 | 3 | 0 |

Eğer kurulumda optimizasyon yapılmak istenirse amaç fonksiyonuna kurulum sürelerini minimize edecek bir parametre daha eklemek gerekir. Ayrıca karınca koloni algoritmasının görünürlük fonksiyonunda değişiklik yapmak çözüme daha hızlı ulaşmayı sağlayacaktır. Test için tanımlanan operasyonların hepsinin kurulum maliyeti 15 olarak kabul edilmiştir. Bu değer yukarıdaki matrise göre operasyonun kurulum süresi belirlendikten sonra çarpılır ve kurulum maliyeti ilgili operasyon için bulunur.

İlk popülasyon oluşturulurken problem tanımından yeni bir genom türetilir. Yeni genom üzerinde rastlantısal sıralamalar yapılır ve fit değerleri hesaplanıp popülasyon içerisine yerleştirilir.

Yeni türetilecek birey sayısı, çaprazlama oranı ile hesaplanmış birey sayısıdır. En az 1 bireye karşılık gelmesi için düzenleme yapılmıştır. Çaprazlama oranı en çok %50 olarak verilmelidir. Aksi halde popülasyon içerisinde çok fazla yeni birey olduğu için tecrübe kazanması uzun sürecek ve problem çözüm kümesine yakınsaması zorlaşacaktır. Yeni bireylerin üzerindeki makinelerin barındırdığı feromon matrisi mevcut tecrübeyi taşıyacaktır ancak birçok makinenin taşınan tecrübelerinin entegre hale gelmesi ilk jenerasyonda zordur.

Yapılan testlerde popülasyon sayısı sipariş sayısının iki katı olarak verilmiştir.

### 3.2.2. ALGORİTMA

Problem çözümünde öncelikle kodlama işlemi yapılarak kullanılacak veriler hazırlanır. İlk popülasyon oluşturulur ve fit değerleri hazırlanarak algoritma çalışmaya başlar. Çaprazlama operatörü uygulanır ve popülasyon içinden en kötü olan bireyler yeni bireyler ile yer değiştirilir. Yeni bireyler ile birlikte oluşan popülasyon gen operatörlerine tabi tutulur.

Gen operatörleri içinde öncelikle evrim operatörü gerçekleştirilir. Evrimin lokomotifi olan mutasyon operatörü iki paralel katmandan oluşur. Bu katmanların birisi karınca koloni algoritması temelli, diğeri ise genetik algoritmanın temel mutasyon operatörüdür. Oluşan yeni genomlara onarım operatörü uygulanır.

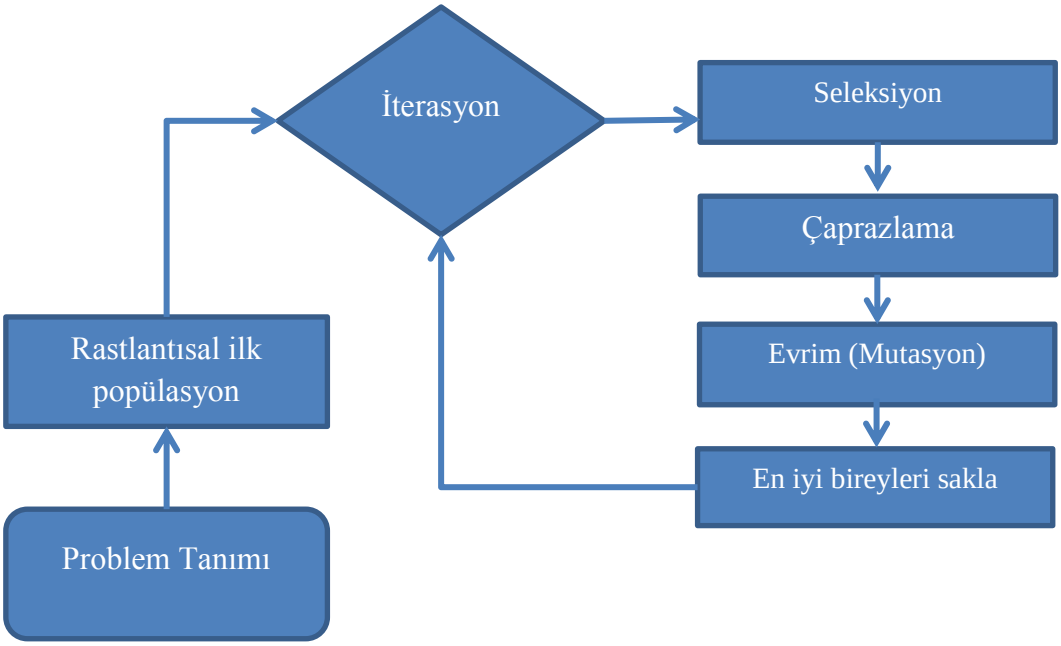
Artık genom üzerindeki sıralamaların karşılıklarını bulmak için planlama operatörü devreye girebilir. Planlama bilgileri hesaplanır ve operasyonların hangi iş merkezlerinde ne zaman işlem göreceği veya göremeyeceği belirlenir. Amaç fonksiyonunda kullanılacak değerler hazırlanır ve amaç fonksiyonu hesaplanır.

Yineleme sayısına göre algoritma durur. Ayrıca arka planda kaç jenerasyon iyileşme kaydedilmeden geçtiği sayılır. Belirlenen değerlere göre bu sayıda durdurma kriteri olarak kullanılır. Algoritmanın pseudo kodu aşağıda verilmiştir.

**Şekil 3.2 Algoritma Pseudo kodu**

```
yeni popülasyon oluştur();
planlama ve fit değerlerini hesapla();
while (iterasyon limiti)
        çaprazlama();
        while(popülasyon sayısı)
                evrim();
                onarım();
                planlama();
                fit değerini hesapla();
        endwhile;
        en iyi bireyleri sakla();
        iyileşmeyi kontrol et ve gerekirse hesaplamayı bitir();
endwhile;
```

**Şekil 3.3 Temel Algoritma**



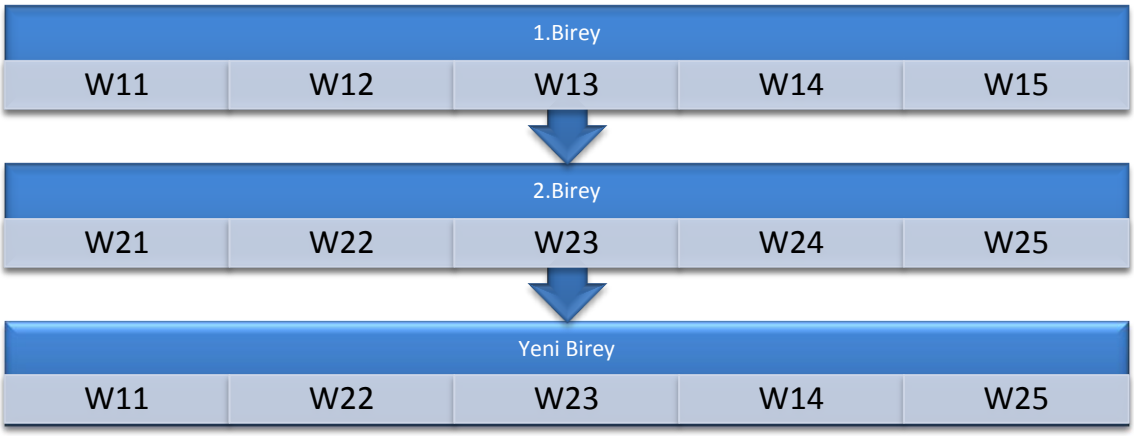
### 3.2.3. ÇAPRAZLAMA OPERATÖRÜ

Çaprazlama operatörü popülasyona yeni bireyler kazandırmak için uygulanır. Algoritma içerisinde çaprazlama oranı parametre olarak fonksiyona verilir. Bu oran dahilinde öncelikle kaç yeni birey oluşmasına karar verir ve popülasyon içinde en düşük fit değerine sahip yeni gelecek birey sayısı kadar mevcut birey çıkarılır.

Problem çözümünde genomun tasarımında iş merkezleri alellere yerleştirilmişti. İş merkezleri üzerinde operasyonlar, feromon matrisi ve planlamaya dair bazı bilgiler bulunmaktadır. İlişkili olan operasyonlarda eğer farklı iş merkezlerinde farklı sıralamalar ile birleştirilerek yorumlandığı zaman daha iyi kombinasyonlar oluşturabilir. Bu fikir ile popülasyondan seçilen rastgele iki genom alınır. Yeni boş bir genom oluşturulur. Bütün genomlarda aynı locus üzerinde aynı iş merkezi bulunacağı için olası çözüm üzerinde büyük bozulma yaşanmadan hangi locus'a hangi bireyin iş merkezi taşınacağı rastlantısal olarak belirlenir. Taşınan iş merkezi kendi feromon matrisi ile taşınacağı için mevcut sıralamayı da beraberinde getirecektir.

Bu yöntem ile lokal maksimum veya minimumlardan farklı noktalara sıçramalar yapılmaya çalışılmıştır. Mevcut bireyler bulundukları yerdeki lokal minimum veya maksimumlara takılabilir ve buradan kurtulması feromon matrisine bağlı olduğu için düşük ihtimaldir. İş merkezleri arasında ilişkiler olduğu için operasyonların birbirlerini beklemesi veya alternatif operasyonlar olması yeni sıralama ile farklı çözüm oluşturacaktır.

**Şekil 3.4 Çaprazlama Operasyonu**




Eğer iş merkezleri bir hücresel imalat merkezine ait ise bu merkeze ait tüm iş merkezleri bir bireyden alınarak taşınması daha olumlu sonuçlar üretecektir. Eğer sadece hücresel imalat merkezine ait iş merkezleri planlanıyorsa buna gerek yoktur.

### 3.2.4. MUTASYON OPERATÖRÜ

Mutasyon operatörü genetik algoritmalarda amaç fonksiyonuna uygun olarak sıçramalar yapabilmek için geliştirilmiştir. Bazı kurallar dahilinde yapılan işlemlerin haricinde çözüm uzayında belirli bir lokalde arama yaparken genlerin değişmesi ile yeni bir çözüm arayışına girmektir. Mutasyon operatörü temel teoride genlerin üzerindeki bilgilerin değişmesi ile olur. Bu değişim sonucunda çözüm bireyi uygun çözüm olmaktan çıkabilir bu sebeple onarım operatörüne ihtiyaç duyar. Mutasyon operatörü için çok çeşitli yöntemler vardır.

Kapasite planlama problemlerinde amacımız iş merkezleri üzerindeki işleri sıraya dizmek olduğu göz önünde bulundurulduğunda mutasyon operatörü operasyonları rastlantısal olarak seçip yer değiştirmektedir. Seçilen bir operasyonu alıp farklı iki nokta arasına koymakta algoritma içerisinde parametreye bağlanıp denenmiştir. Tartışma bölümünde bunların sonuçlarından bahsedilecektir.

Mutasyon işlemini gerçekleştirirken temel olarak ACO'dan faydalanılmıştır. ACO kullanımı bir sonraki bölümde detaylı olarak anlatılacaktır.

Mutasyon işleminin en önemli parametresi mutasyon oranıdır. Kullanılan algoritma içerisinde iki parametre bulunmaktadır. MR olarak ilk parametre mutasyon oranını vermektedir. MDR ise mutasyon oranının her yeni jenerasyonda azaltılması için gerekli parametredir. Her yeni jenerasyonda aşağıdaki formül ile MR belirlenir.

$$MR \begin{cases} MR - MDR & MR - MDR > MRMIN \\ MRMIN & MR - MDR < MRMIN \end{cases}$$

**Denklem 3.2. Mutasyon oranının jenerasyon bazında değişimi**

MRMIN ve MRMAX değerleri parametre olarak verilir ve en küçük ve en büyük mutasyon oranlarını belirler. Buradaki amaç her jenerasyonda iyileşmeler olduğu kabul etmek ve mutasyon oranını azaltarak daha küçük hamleler ile daha iyi sonuçları bulmaya çalışmaktır. Ancak bu yöntemde popülasyon içerisindeki fit değerleri dikkate alınmadığı için fit değeri düşük ve yüksek olan tüm genomlar aynı mutasyon oranı ile işlem görmektedir. Bu durumu düzeltmek için bir formül geliştirilmiştir.

Önerilen formül bir önceki jenerasyondaki bütün fit değerlerin saklanması ve istatistik bilgilerinin çıkarılmasına dayanmaktadır. Öncelikle popülasyonun ortalaması bulunur. Popülasyon ortalaması hesaplanırken ortalamayı saptıracak uç verileri engellemek için bayes doğrulaması uygulanmıştır. Standart sapma değeri de bulunarak seçilen bireyin bir önceki nesilde ortalamanın ne kadar altında veya üstünde olduğuna dair bir bilgi bulunur ve MR değerine çarpan olarak eklenir. Bu yöntem ile ortalamanın altında kalan bireylere ki standart sapma ile yapılan hesaplamaya göre ne kadar uzaklaştılar ise daha fazla mutasyona uğramalarını, benzer şekilde ortalamanın üzerinde ise daha az mutasyona uğramaları sağlanmıştır. Ana mutasyon oranının kontrolünde uygulanan yöntem burada da uygulanmıştır.

$$MR = MR - \left( (MR) \frac{f_{fit}(genome) - \mu_L}{\sigma_L} \right)$$

$$MR \begin{cases} MR & MRMIN < MR < MRMAX \\ MRMIN & MRMIN > MR \\ MRMAX & MR > MRMAX \end{cases}$$

**Denklem 3.3 Genom bazında mutasyon oranının belirlenmesi**

Belirli bir jenerasyondan sonra ACO uygulamak bazı durumlarda fayda sağlamamaktadır. Bu neden ile parametre olarak verilen jenerasyondan sonra ACO mutasyonu kaldırılıp sadece genetik mutasyon yapılması için algoritma geliştirilmiştir. İstenirse popülasyon iyileşme gösteremediği durumlarda devreye alınabilecek şekilde tasarlanmıştır. Yani popülasyondan daha iyi yeni bireyler daha fazla türeyemiyorsa sadece mutasyon yaparak aramaları gerçekleştirmek performans açısından daha iyi sonuç verecektir.

### 3.2.5. KARINCA KOLONİ OPTİMİZASYONU

GA ile oluşturulan melez modelde ACO her iş merkezi üzerinde ayrı olarak işlem yaparak operasyonları sıraya dizmeye çalışır. ACO algoritmasının temelini oluşturan feromon matrisi iş merkezi üzerinde bulunur. Böylece bir iş merkezi üzerindeki operasyon sıralaması çaprazlamaya tabi tutulsa bile taşınır ve sıralama temel anlamda korunur.

İş merkezi üzerinde ilk operasyonun nasıl seçileceğini hesaplayabilmek için feromon matrisi üzerinde bir hesaplama yapılır ancak bu hesaplama her operasyonun matriste kendisiyle kesiştiği noktadan hesaplanır ve güncellenir. Standart uygulanan yöntemlerde rastgele bir operasyon seçilerek sıralamaya başlanır ancak bu yöntem sadece başlandığı noktaya geri dönmesi esas olan sıralamalarda işe yaramaktadır.

Operasyonlar sıralanırken uygulanan yöntem daima bir önceki operasyon referans alınarak bir sonraki operasyonu feromon matrisi üzerinde belirlenen ağırlığı alınır. Bu değer görünürlük fonksiyonu ile işlenir. Eğer operasyon sıralamada daha önce yer aldıysa 0 olarak alınır. Tüm olası operasyonlar için bu işlem yapılıp değerler toplanır.

Rastgele seçilen bir değerin hangi noktaya geldiği bulunur ve operasyon sıralamaya alınır.

Bu işlem sırasında sezgisel yöntemi güçlendirmek için görünürlük çarpanı iki adet olarak belirlenmiştir. İlk görünürlük çarpanı operasyonların seviyelerine göre öncelik tanıması esasına dayanır. İkinci görünürlük ise planlama baz tarihi dikkate alınarak ilgili teslim süresine ne kadar uzaklıkta olduğudur. Bu iki kriter aslında operasyonların iş merkezlerine yerleştirilirken dikkat edilen en büyük iki etmendir. Bu iki fonksiyon; atölye tipi iş merkezlerinde aynı operasyonları yapan iş merkezlerinin öncelikli olarak düşük seviyedeki işleri yapmasını ve hem akış hem de atölye tipi üretim için teslim süresi yakın olan operasyonun daha önce yapılmasını sağlamaktadır. Teslim süresi üzerinde hesaplama yapılırken kurulum süreleri dikkate alınmamıştır.

$$P_{ij} = \tau(i,j)^{\alpha}.\left(\frac{1}{Lv_o{}^2}.\frac{1}{(D_t-(S_t+O_t))}\right)^{\beta}$$

**Denklem 3.4 Görünürlük Fonksiyonu**

Zaman birimleri kapasite planlamadaki ölçütlere göre saat veya dakika olarak alınabilir. Operasyon sürelerinin çok uzun olduğu durumlarda gün olarak alınabilir. $O_t$ siparişin teslimi için gereken tüm operasyonların, olası toplam sürelerinin en uzunu olarak alınmalıdır. Araştırma dâhilinde yapılan testlerde dakika olarak alınmıştır.

Karınca sayısı problem uzayının araştırılmasında önemli bir faktördür. Çok sayıda olması yapılan aramaların daha kısa jenerasyonda sonuç vermesine gidebilir ancak iyileşmeler ve yeni bireyler göz önünde bulundurulduğunda performans açısından ciddi anlamda kayba neden olacaktır.

Önerilen yöntem üzerinde karınca sayısının planlanması gereken operasyon sayısı kadar olması öngörülmüştür. Mutasyon işlemi yapılacağı zaman tüm iş merkezlerinde birer karınca gezisi düzenlenir. Yani iş merkezi sayısı kadar karınca bulundukları iş merkezindeki operasyonları sıralamaya çalışır. Bir genom üzerinde bu işlem tamamlanınca onarım, planlama, fit değer hesaplaması yapılır ve genom ayrı bir popülasyonun içinde tutulur. Parametre olarak verilen karınca sayısı kadar bu işlem tekrarlanır. Oluşan yeni bireylerden birer kopya alınır ve genetik mutasyona tabi tutulur.

Mutasyon operatörü bu aşamada ACO'dan gelen yeni bireylerin kopyalarına uygulanır. Amaç mevcut ACO çözümünü mutasyon ile daha iyi hale getirebilmektir. Mutasyona uğrayan kopya bireylerde oluşturulan popülasyona eklenir. Fit değerlerine göre en iyi olan birey mutasyonun sonucu olarak gerçek popülasyona gönderilir. Karınca sayısı kadar işlem yapılmasındaki amaç mevcut feromon matrisi bazlı seçimlerde daha fazla olası durumu değerlendirmektir.

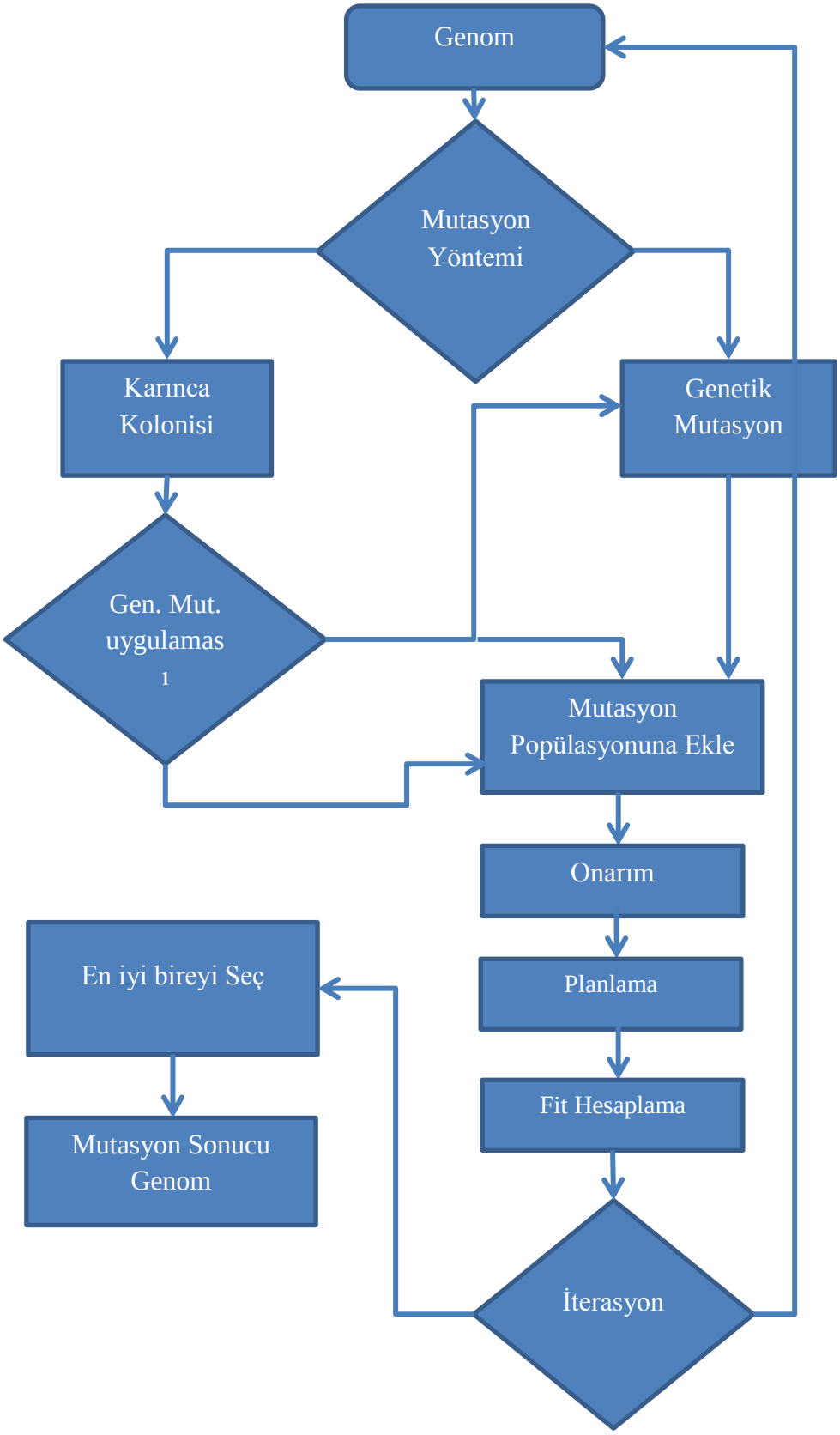
Bulunan en iyi genom sonucu ile iş merkezleri üzerindeki feromon matrisleri belirlenen buharlaşma (Q) değeri ile önce buharlaştırılır, sonra artırım (rho) değeri ile arttırılır. GA içerisindeki popülasyon incelenir ve en iyi sonucu veren birey dikkate alınarak global feromon güncellemesi yapılır. Global feromon güncellemesi Grho ile yapılır ve buharlaşma uygulanmaz.

Belirlenen parametre ile ana popülasyon belirli bir jenerasyona ulaştığı zaman ACO feromon matrisindeki güncellemelerden ötürü işlevselliğini yitirmeye başlayacağı için yerine sadece genetik mutasyona bırakmaktadır.

Mutasyon algoritması basit haliyle;

1- Mutasyon uygulanacak Genomun hangi yöntem ve mutasyon oranı ile işleneceği bilgisi alınır
2- Genomun bir kopyası alınır ve mutasyon uygulanmaya başlar
    a. Sadece karınca kolonisi uygulanacak ise sonrasında birey popülasyona eklenir
    b. Genetik mutasyon uygulanacak ise sonrasında birey popülasyona eklenir
    c. Her iki yöntemde uygulanacak ise; önce karınca kolonisi uygulanır ve bir kopyası alınıp genetik mutasyona gönderilir, orijinal birey popülasyona eklenir. Genetik mutasyona uğrayan kopya birey popülasyona eklenir
3- Onarım, planlama ve fit hesaplamalarından sonra birey karşılaştırılabilir hale gelir.
4- Belirlenen mutasyon popülasyon sayısı kadar iterasyon uygulanır ve her iterasyon sonucu çıkan birey mutasyon popülasyonuna eklenir.
5- Popülasyon belirlenen sayıya ulaşınca en iyi bireyi mutasyon sonucu olarak geri gönderilir.

**Şekil 3.5 Mutasyon Algoritması**

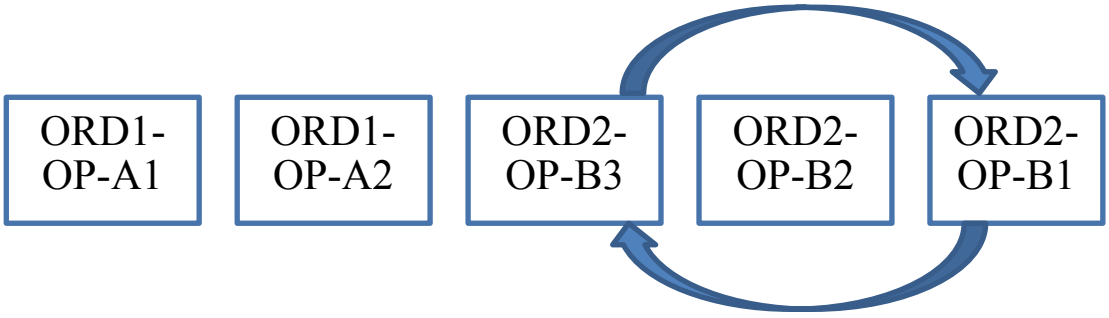


### 3.2.6. ONARIM

Uygulanan mutasyon operasyonları sonucunda aynı iş merkezi üzerinde yapılması gereken operasyonların sıralamalarında bozulmalar gerçekleşebilir. Bu bozulmaları gidermek amacıyla onarım algoritması oluşturulmuştur.

Algoritma tasarımında gen üzerindeki verilerin bozulması engellenecek şekilde tasarım yapılmıştır. Herhangi bir verinin bozulup kaybolması ve yeniden oluşturulması gerekmemektedir. Bu sayede karmaşık bir kontrol algoritmasına ihtiyaç kalmamaktadır.

**Şekil 3.6 Örnek Onarım**




Şekil 3.5 de görüldüğü gibi eğer bir iş merkezi üzerinde birbirleri ile ilişkilendirilmiş operasyonların sıralamaları, operasyon üzerinde bulunan mrp seviyesine göre uygun değilse yer değiştirilir. Çünkü ORD2-OP-B1 operasyonu işlem görmeden ORD-OP-B2 işlem göremez.

### 3.2.7. PLANLAMA

Genetik algoritma tasarımı yapılırken yapılan evrim operasyonlarında zamanı genin üzerine yerleştirmek ciddi bir problemdir. Yapılan tüm hesaplamalarda mutasyonda, çaprazlamada ve ACO üzerinde zaman parametresini dikkate alarak hesaplama yapmak performans anlamında büyük kayıplara yol açar. Çünkü bir operasyonun başlaması ve bitişi birçok hesaplama ile belirlenir. Operasyon tatil öncesi başlamış ve bitmemiş ise tatil süresini dikkate alacak şekilde hesaplama yapılmalıdır. Ayrıca bakım planlamaları, sabit duraklama zamanları gibi operasyon bitiş süresini hesaplarken dikkate alınması

gereken çok fazla parametre bulunmaktadır. Bu sebeple süre hesaplamaları işler sıraya dizildikten sonra planlama operatörü ile yapılmıştır.

Planlama operatörü; iş merkezleri üzerine sıralanmış olan operasyonların kısıtları sağlayıp sağlamadığını kontrol ederek uygun çözüm olup olmadığı araştırılır. Belirlenen başlangıç tarihini temel alarak iş merkezleri üzerindeki ilk planlanabilir işlerin en erken başlangıç tarihleri alınır ve sıralanır. Başlangıç tarihi hesaplanırken kaynaklar ve ilişkiler sorgulanır, eğer operasyonun başlaması başka operasyonlara bağlı ise bu operasyonların bitiş zamanı, gerekli malzeme var ise bu malzemelerin tedarik zamanları karşılaştırılır ve en geç olanı alınır. Tüm iş merkezlerindeki planlanmamış ilk operasyonların hepsi için bu hesaplama yapılır. En erken başlanabilmesi olası olan operasyon planlandı olarak işaretlenir ve süre hesaplamaları gerçekleştirilir. Eğer operasyon yarım kalmış bir operasyonun devamı ise tamamlanma oranı dikkate alınır. Operasyon planlandıktan sonra aynı işlemler iş merkezleri üzerinde planlanabilir hiçbir iş kalmadığı sürece devam edilir.

Algoritma içerisinde her iş merkezinin bir takvimi olması ve bunu hesaplamalarında dikkate alarak operasyon bitiş sürelerini hesaplaması sağlanmıştır. Takvim içerisine iş merkezlerinin çalışma saatleri, tatil günleri, bakım planları gibi çevrimdışı olduğu zamanlarda belirtilmiştir.

Planlama operatöründen sonra operasyonların planlanıp planlanamadığı belirlenir. Tüm operasyonların tarihleri hesaplanmış olur.

### 3.2.8. FİT HESAPLAMA

Planlama operatöründen sonra amaç fonksiyonunda kullanılacak değerler artık hesaplanabilir. Fit değeri hesaplanırken hem süre temelli değerler hem de maliyet odaklı değerler hesaplanmıştır. Amaç fonksiyonu işletme önceliklerine göre belirlenebilir. Tüm değerler için amaç fonksiyonundaki katsayıları parametre olarak alınmıştır. Eğer bir değerin faktör parametresi 0 olarak verilmiş ise o değerin hesaplamaları işlem kaybı olmaması açısından yapılmaz.

Planlanma oranı; problem içerisinde tanımlanan operasyonların yüzde olarak kaçının planlandığını bulur. Yapılan testlerde fit değerinin ana çarpanlarından biri olarak yazılmıştır.

Sipariş Teslim Gecikmesi ve Erken Bulundurma Süresi; performans kazancı için aynı anda hesaplanmıştır. Bir siparişin teslim süresi ilişki olarak problem üzerinde tanımlanmış ise operasyonun bitiş tarihi ile arasında kalan süre hesaplanır. Eğer çıkan fark negatif ise sipariş gecikmiş demektir ve sipariş teslim gecikmesi değerine eklenir, aksi durumda erken bulundurma süresine eklenir. Her iki değer için maliyet karşılıkları ile hesaplamalar yapılabilir.

Operasyon ve kurulum maliyetleri; aynı anda hesaplanır. Kurulum maliyeti; mevcut operasyonun kurulum tipi ile önceki operasyonun kurulum tipleri arasında makine üzerinde tanımlanmış matristen bulunan değer ile süresi hesaplanıp maliyet faktörü ile çarpılarak bulunur. Operasyonun maliyeti; makine üzerindeki birim süre ile operasyon üzerinde makineye özel tanımlanmış olan birim maliyet çarpılarak hesaplanır. Ayrıca hesaplanması gereken birim temelli veya tek seferlik bir maliyet var ise operasyon içerisinde iki adet aktivite tanımlanmıştır. Kurulum maliyeti de operasyon maliyetine eklenir. Kurulum maliyeti için ayrıca değer saklanır.

Fason Maliyeti; iş merkezleri tipinde fason olarak belirlenmiş olan operasyonların maliyetleri hesaplanarak bu değere eklenir.

Atıl Zaman; iki operasyon arasında iş merkezi boşta kalmış ve operasyon planlaması çeşitli sebeplerden dolayı gecikmiş ise planlamada negatif faktör olarak şekilde hesaplanır. Genel üretim verimliliğini etkilediği için yüksek bir faktör verilir. İlk operasyonlar için atıl zaman hesaplaması dikkate alınmamıştır.

Tüm Operasyonların Toplam Süresi; tüm operasyonların gerçekleşen süreleri toplanarak genel anlamda iş merkezleri üzerinde toplamda ne kadar yük olduğu hesaplanmıştır.

Gecikmeler; operasyonun olası en erken başlama tarihi ile planlama sonucunda hesaplanan başlangıç tarihi arasındaki fark olacak şekilde hesaplanmıştır. Stokta bekletme veya özel bir maliyet faktörü ile çarpılarak amaç fonksiyonuna eklenebilir.

Ayrıca gecikmelerin süre toplamı tüm operasyonların sürelerine oranlanarak ortalama gecikme süresi hesaplanmıştır.

Çevrim Zamanı; iş merkezleri üzerindeki en erken başlama zamanı ile en geç bitiş zamanı arasındaki fark olarak hesaplanmıştır. Birçok kapasite planlama probleminde minimize edildiğinde iş merkezleri üzerine işlerin daha dengeli yüklenmesini sağlar. Aksi durumlarda maliyet temelli bir amaç fonksiyonunda en ucuz iş merkezine işler yetiştiği müddetçe yükleme yapabilir. Bu bazı zamanlarda istenen bir durumda olabilir.

### 3.2.9. DEĞERLENDİRME

Fit değerleri hesaplandıktan sonra tüm popülasyon aldıkları değere göre sıralanır ve aralarında en iyi puana sahip olan bireyler en iyi popülasyona girebiliyorsa burada saklanır. Bunun amacı herhangi bir operatör uygulandığı anda bozulmalara maruz kalmasını engellemektir.

Parametre olarak verilebileceği gibi sistem içerisinde toplam jenerasyonun yarısı olarak kabul edilen bir noktadan sonra sistem ACO algoritmasını durdurur ve sadece genetik mutasyonu aktif hale getirir.

İyileşme olmadan geçen geçen jenerasyon sayısı ayrıca hesaplanır. Ne zaman daha iyi bir birey keşfedilirse bu değer sıfırlanır. Eğer popülasyon içerisinde belirlenen sayıdan fazla jenerasyonda iyileşme kaydedilemiyorsa döngü kırılmakta ve çözüm arayışı sonlandırılmaktadır.

Testler yapılırken popülasyon sayısı ve mutasyon popülasyon sayısı sipariş sayısının iki katı olarak verilmiştir. İterasyon sayısı 500, melez mutasyonu bırakıp sadece genetik mutasyona yönelmesi için iterasyon limiti 100 olarak ayarlanmıştır. Her mutasyon tipi için 3 set üzerinde üçer defa çalıştırılmış ve en her seferinde en iyi 3 sonuç elitizim ile alınmıştır.

# 4. BULGULAR

Çalışma öncelikle genetik algoritma ile başlamıştır. Genetik algoritma küçük çaplı problemleri çözmek için uygun bir performansa sahip ancak problem boyutu büyüdükçe çok sayıda yenileme yapılarak çözüme ulaşılabilmektedir. Bu sebeple daha kararlı hareket edebilmek için ACO algoritması mutasyon temeli olarak kullanılmıştır. ACO içerisinde rastlantısal kararlar verilirken görünürlük fonksiyonu sayesinde amaca daha hızlı ulaşıldığı görülmüştür. Yapılan algoritma esnek tasarlanmıştır ve gerektiğinde belirli operatörler devre dışı bırakılmıştır. Hangi operatörlerin daha iyi sonuçlar verdiği bulunmak için testler yapılmıştır. 3 sipariş setinde genetik algoritma, karınca kolonisi ve melez model ayrı ayrı çalıştırılmış ve sonuçları eklerde verilmiştir.

**Tablo 4.1 Test sonuçları skor karşılaştırma tablosu**

| Yöntem – veri | Test 1 | | | Test 2 | | | Test 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| Sıra | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 |
| GA-1 | 126 | 126 | 126 | 128 | 128 | **130** | 127 | 127 | 129 |
| ACO-1 | 127 | 127 | 128 | 126 | 126 | **129** | 124 | 126 | 128 |
| M-1 | 128 | 129 | **130** | 128 | 128 | 129 | 124 | 124 | 128 |
| GA-2 | 243 | 246 | 250 | 231 | 233 | 240 | 254 | **273** | 274 |
| ACO-2 | 233 | 244 | 247 | 204 | 224 | 242 | 239 | 249 | **273** |
| M-2 | 281 | 289 | **291** | 265 | 274 | 278 | 242 | 247 | 274 |
| GA-3 | 109 | 147 | 164 | 160 | 170 | 177 | 146 | 148 | **248** |
| ACO-3 | 178 | 184 | 188 | 123 | 145 | **220** | 149 | 171 | 183 |
| M-3 | 306 | 311 | 338 | 313 | 325 | **368** | 204 | 237 | 350 |

Skor bazında değerlendirme yaptığımız zaman küçük problemleri çözmede ufak farklılıklarda olsa sonuçlar birbirine yakın çıkmıştır ve sonuçlarda bir istikrar söz konusudur. Çıkan sonuçların standart sapmaları çok düşüktür. Orta ölçekli bir problemde melez yöntem farkını hissettirmektedir. Üstelik bazı test sonuçlarında yalın yöntemlerde lokal maksimumlara takılmaların özellikle karınca kolonisinde daha fazla

olduğunu görüyoruz. Problem boyutu büyüdükçe ve karmaşıklaştıkça genetik algoritma sonuçları ortalamanın çok altında kalmakla beraber yalnızca bir iyi sonuç üretebilmiştir. Karınca kolonisi biraz daha istikrarlı bir eğilim göstermiş ancak iyi sonuç bulmakta gene başarılı olamamıştır. Melez yöntem en iyi skoru almakla beraber son test haricinde çıkan sonuçlar istikrarlıdır.

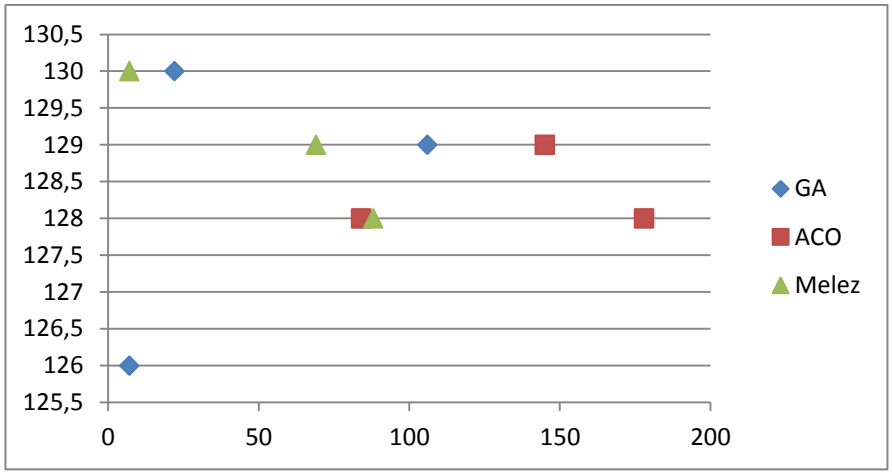
**Tablo 4.2 Test sonuçları süre ve iterasyon karşılaştırma tablosu**

| Yöntem – veri | Test 1 | | Test 2 | | Test 3 | |
|---|---|---|---|---|---|---|
| | E.S.İ. | Süre(sn) | E.S.İ. | Süre(sn) | E.S.İ. | Süre(sn) |
| GA-1 | 7 | **10** | 22 | 11 | 106 | 18 |
| ACO-1 | 84 | **18** | 145 | 26 | 178 | 29 |
| M-1 | 7 | **12** | 69 | 19 | 88 | 29 |
| GA-2 | 20 | **18** | 85 | 27 | 195 | 44 |
| ACO-2 | 30 | **83** | 94 | 120 | 100 | 131 |
| M-2 | 107 | 130 | 100 | 130 | 1 | **82** |
| GA-3 | 20 | 25 | 106 | 42 | 12 | **24** |
| ACO-3 | 79 | 313 | 62 | **299** | 169 | 490 |
| M-3 | 153 | 431 | 132 | 424 | 8 | **224** |

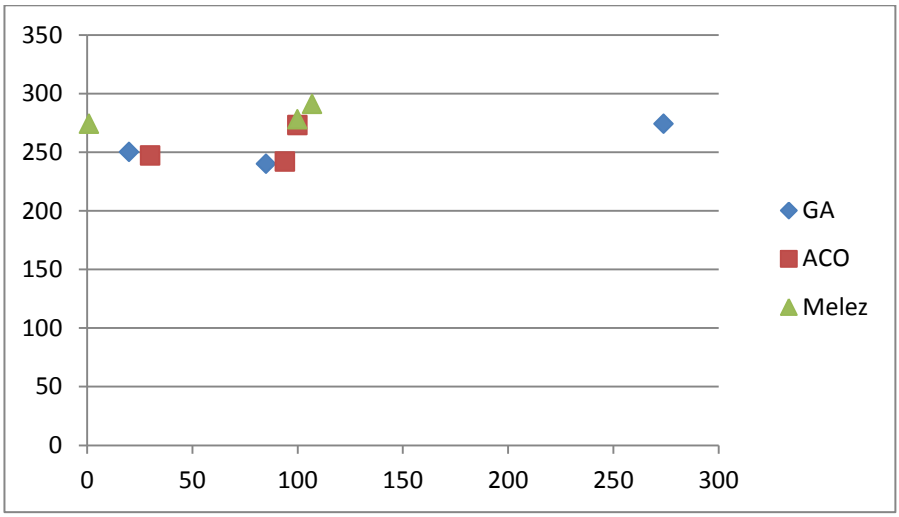
ESİ: en iyi skorun bulunduğu iterasyon

Küçük ölçekli problemlerin çözümlerinin bulunmasında genetik algoritmanın tek başına karınca kolonisinden daha iyi olduğu görülmektedir. Melez yöntem neredeyse genetik algoritmaya yakındır ancak genetik algoritma kadar iyi değildir. Orta ölçekli bir problemde karınca kolonisi çok fazla iterasyondan sonra iyi sonuçlara ulaşabilmektedir. Genetik algoritma ise son testte çok fazla iterasyon yapmıştır. Büyük ölçekli bir problemde genetik algoritma en hızlılarıdır. Sonrasında karınca kolonisi takip etmektedir. Melez yöntem büyük problemlerde çok daha uzun sürmektedir. Ancak son testte görülmektedir ki melez yöntem 8. iterasyonda en iyi sonucu bulmuştur.

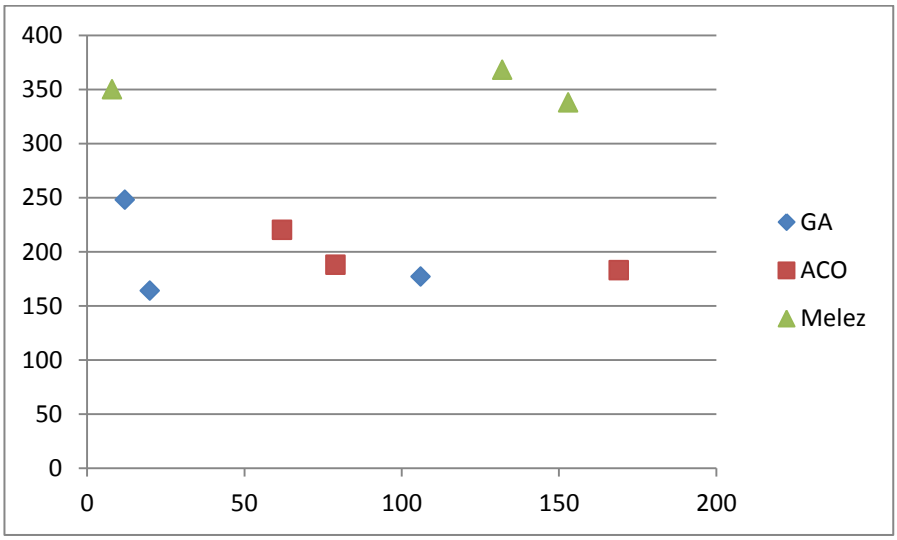
**Şekil 4.1 1. veri seti test sonuçları**




**Şekil 4.2 2. veri seti test sonuçları**




**Şekil 4.3 3. veri seti test sonuçları**



Grafikler dikkat edilmesi gereken nokta; eğer daha sol tarafta ise daha az iterasyon yaptığı anlamına gelir, yukarıda olması zaten bizim istediğimiz puanın yüksek olması anlamına gelir. Beklenti sol ve üst bölgede bulunan noktalardan seçim yapmaktır. 1. veri setinde genetik algoritma ile melez model birbirine yakın değerler üretmiş ama noktaların birbirine dikey anlamda uzak olması genetik algoritmadan çıkacak sonuçların her zaman iyi bir eğilim göstermeyeceğinin kanıtıdır.

2. veri setinde ise değerler birbirine yakın gözükmektedir. Genetik algoritmanın çok fazla iterasyon yapan bir çözümü de burada görülmektedir. Melez model en iyi sonuçları vermiş ve noktaların grafik üzerinde dikey doğrultuda birbirine yakın olması kararlı çözümlerin bulunduğunu gösterir.

3. veri setinde çok belirgin olarak melez modelin daha iyi sonuçlar verdiği görülür. Genetik algoritma ve karınca koloni optimizasyonu dikeyde daha yakın değerler çıkarmıştır. İterasyon sayılarına baktığımız zaman genetik algoritma daha düşük iterasyonlar ile sonucu bulmuştur. Karınca koloni optimizasyonu ise daha yüksek iterasyonlarda benzer sonuçları yakalayabilmiştir. Buna zaman boyutu da eklendiğinde genetik algoritma daha iyi kalmaktadır. Melez model ise diğer yalın modellerden daha iyi sonuçlar üretmiştir ve önceki veri setlerinde olduğu gibi burada da kararlılık ile sonuca gitmiştir. Melez modelin dezavantajı iterasyon sayısının fazla olması ve zaman boyutunda daha uzun sürmesidir. Ancak sistemden kararlı ve iyi veriler istiyorsak bizim için ideal bir seçim olacaktır.

# 5. TARTIŞMA

Melez modelin çözüm olarak önerilmesindeki amaç belirli problemlerin çözümünde uzmanlaşmış iki algoritmadan kapasite planlamasına uygun çözüm üreten bir algoritma geliştirmektir. Yapılan test sonuçlarında GA, tek başına uzun jenerasyonlar sonucunda iyi puanlar almayı başarmıştır. Ayrıca GA genel olarak çok istikrarlı sonuçlar sunamamaktadır. Test sonuçlarında görüldüğü üzere problem boyutları büyüdükçe başarısız olmaktadır.

ACO ise bulabildiği en iyi sonuçları çok daha kısa yenilemede bulabilmiş ancak sonuçlar küçük ölçekli problemlerde GA kadar iyi olmamıştır. ACO işlem süresi genetik algoritmaya göre daha uzun sürmüştür. Bunun sebeplerinin başında ACO üzerinde bulunan feromon ağırlıklandırması ile rastlantısallığın sağlanmasıdır. Daha erken bulduğu çözümü iyileştiremediği için kısıntının etkisiyle toplam jenerasyon sayısı GA'ya göre daha azdır. Bunun sebebi feromon matrisi belirli bir noktadan sonra yeni çözümler üretmekte daha kısır kalmasından kaynaklanmaktadır.

Melez modelimizde ise ACO ilk iyileşmeleri sağlamakta, üzerine genetik mutasyon uygulanması ile çıkan çözümden daha iyi çözümler aramaktadır. Melez modelde iterasyon sayısı son test haricinde daha uzun sürmüştür. Bunun sebebi her jenerasyonda ACO ile belirli bir istikrarlı çözüm sağlanmakta ve sonrasında iyileştirmeler aranmaktadır. Bu iyileşmeler çözüm kümesinde aranırken feromon matrisine bağımlı olmadığı için sıçramaları daha bağımsız ve rastlantısal yapmaktadır. Böylece lokal maksimum veya minimumlardan uzaklaşma sağlanabilmektedir. Bunun dezavantajı süre ikili mutasyondan dolayı ciddi şekilde uzamaktadır.

Mutasyonun kendine ait bir popülasyon oluşturması algoritmanın daha garanti bir şekilde ilerlemesini sağlamaktadır. Mevcut feromon matrisi ile yeni genom oluşturulur ancak bu oluşum sırasında çeşitli rastlantılar sonucunda iyi bireylerden kötü bireyler oluşabilir. Benzer şekilde mevcut feromon matrisi ile ufak farklar ile çok iyi bireylerde oluşabilir. Amaç mevcut bireye mutasyon uygularken bir defa değil mutasyon popülasyonu olan sayı kadar mutasyon yapıp küçük bir popülasyon oluşturmaktır. ACO'dan gelen bireylerin üzerinde belirlenen mutasyon oranı kadar mutasyon operatörü uygulayınca bir o kadar daha birey türetilmiş olur. Böylece hem ACO ile

oluşturulan hem de mutasyon uygulanmış bireylerin oluşturduğu popülasyon içerisinden en iyi birey seçilip mutasyon operatörünün sonucu olarak döndürülür. Böylelikle bir bireyden mevcut feromon matrisi ve mutasyon oranı ile olası en iyi birey ortaya çıkarılmaya çalışılır.

Küçük ölçekli problemlerde genetik algoritma eşit orandaki değişkenlik ile problem çözümlerinde daha iyi sonuçlar çıkarmaktadır. Bunun sebebi genetik mutasyonun bağımsız rastlantılar ile hareket etmesidir. Problem ölçeği büyüdükçe karınca kolonisinin daha iyi sonuçlar verdiği görülmektedir. 3. veri setinde ACO, GA algoritmadan daha iyi sonuçlar vermiştir. Bunun sebebi üzerindeki görünürlük fonksiyonunun seçimlerdeki etkisidir. Önerilen melez model her durumda iyi sonuçlar üretmeyi başarmıştır. Ancak zaman maliyetini yükseltmektedir.

Önemli bir noktada önerilen melez model içerisinde zamanın planlama fonksiyonu ile hesaplanması ve operasyonların yerleştirilmesi. Bu yöntem ile zamanın nasıl hesaplancağı sistem üzerinde tanımlanabilmektedir. Tatil günleri, bakım planları, çalışma saatleri gibi kritik bilgiler takvime yüklenip buradan hesaplanabilir. Böylelikle operasyon süreleri daha detaylı hesaplanmakta ve gerçeğe daha yakın sonuçlar üretmektedir.

Kullanılan mutasyon popülasyonu özelliği mevcut feromon matrisini ve mevcut bireyi en iyi çözüm kümesine mevcut noktadan ne kadar yaklaşabileceğini sorgulamaktadır. Bu özellik sayesinde feromon matrisini veya bireyin mevcut durumunu olasılık dahilinde kötüye götürmek yerine daha iyiye götürmeyi başarmıştır.

# 6. SONUÇ

Problem çözümlerinde yöntem seçmenin bazı kriterleri bulunmaktadır. Bunlardan en önemlisi algoritmanın ne kadarlık bir güven aralığında çözümleri bulduğudur. Gerçek hayat problemlerinde birden fazla defa algoritmanın çalıştırılması söz konusu olmayabilir. Sadece bir kere çalıştırılmalı ve onda da istikrarlı sonuçlar üretebilir olmalıdır.

Bir diğer kriterde zamandır. Algoritmanın çalışmasına karşılık gelen bir zaman ve işlem maliyeti bulunmaktadır. Eğer çok hızlı kararların verilmesi gerektiği durumlarda gene güven aralığı dahilinde hızlı olan sistemleri tercih etmek gerekmektedir.

Yapılan testlerin karşılaştırılmasında görülmüştür ki; önerilen yöntem zaman maliyeti açısından çok uygun olmayabilir ancak istikrarlı ve doğru sonuçlar açısından kesinlikle yalın yöntemlerden daha iyi kalmaktadır. Problem boyutlarının büyümesi ile diğer yöntemler belirli lokal maksimum veya minimumlara bağlı kalmaktadır. Genetik algoritma sıçramalar yapmaya uygun bir yapıda olsa da öğrenme ve tecrübe edinme olmadığı için her seferinde rastlantısal kararlar vererek çeşitli çözümleri denemektedir. Karınca kolonisi ise feromon matrisi ile tecrübe ve öğrenme karakteristiklerini sağlamakta ancak bir aşamadan sonra çözüm kümesinde bulunduğu noktanın etrafını araştırmaktadır.

Önerilen yöntem ile karınca kolonisinin genetik mutasyon ile farklı çözüm kümelerine sıçramalar yapması sağlanmıştır. Büyük ölçekli problemlerde karınca kolonisinin görünürlük fonksiyonu ve tecrübe avantajları ile çözüm arayışında hızlı yol alınırken, genetik mutasyon ile mevcut durum iyileştirmeleri sorgulanmaktadır. Önerilen yöntemde ayrıca planlama operatörü entegre edilerek zaman parametresinin karmaşık yapısından biraz da olsa ana yöntemler ayrıştırılmıştır. Literatürdeki bazı çözümlerde zaman boyutunda yerleştirmeler yapılmakta ve onarım algoritmalarına ciddi iş yükü doğurmaktadır.

# 7. KAYNAKÇA

***Kitaplar***

Earl Cox, 2005, *Fuzzy Modeling and Genetic Algorithms for Data Mining and Exploration, Morgan* Kaufmann

***Süreli Yayınlar***

Al-Mashari, M., Al-Mudimigh, A., & Zairi, M. (2003). *Enterprise resource planning: A taxonomy of critical factors.*

F. Robert Jacobs, F.C. Weston Jr. . *Enterprise resource planning (ERP)—A brief history*, 2007.

Robert C. Beatty, Craig D. Williams (2006). *ERP II: best practices for successfully implementing an ERP upgrade. Commun*.

F.D.Ted, Weston, Jr. (2003). *ERP II: The extended enterprise system. Business Horizons*.

Charles Møller, 2005, ERP *II: a conceptual framework for next-generation enterprise systems?* .

V. Botta-Genoulaz, P.-A. Millet, B. Grabot, 2005, A survey on the recent research literature on ERP systems.

K.L. Choy, Y.K. Leung, H.K.H. Chow, T.C. Poon, C.K. Kwong, G.T.S. Ho, S.K. Kwok , 2011, *A hybrid scheduling decision support model for minimizing job tardiness in a make-to-order based mould manufacturing environment* .

Alebachew D. Yimer, Kudret Demirli, 2010, *A genetic approach to two-phase optimization of dynamic supply chain scheduling*.

Li Lin, Huo Jia-zhen, 2009, Multi-*Objective Flexible Job-Shop Scheduling Problem in Steel Tubes Production*.

Mostafa Akhshabi, Javad Haddadnia, Mohammad Akhshabi, 2012, *Solving flow shop scheduling problem using a parallel genetic*.

Young-SU Yun, Mitsuo Gen, 2002, *Advanced scheduling problem using constraint programming techniques in SCM environment*.

Temiz Izzettin, 2010, *Cok Kriterli Permutasyon Akıs Tipi Cizelgeleme*.

Mohammad Reza Amin-Naseri, Mohammad Ali Beheshti-Nia, 2007, *Hybrid flow shop scheduling with parallel batching,*

Khaled Mesghouni, Slim Hammadi, Pierre Borne, 2004, *Evolutionary algorithms for job-shop scheduling,*

R. Qing-dao-er-ji, Yuping Wang, 2011, *A new hybrid genetic algorithm for job shop scheduling problem.*

Pelin Alcan, Hüseyin Baslıgil, 2009, A *genetic algorithm application using fuzzy processing times.*

X. Cai, K.N. Li, A, 1997, *genetic algorithm for scheduling staff of mixed skills under multi-criteria.*

Byung Joo Park, Hyung Rim Choi, Hyun Soo Kim, 2003, *A hybrid genetic algorithm for the job shop scheduling problems*.

Manas Kumar Maiti, 2008, *A fuzzy genetic algorithm with varying population size to solve an inventory model with credit-linked promotional demand in an imprecise planning horizon*.

Kusum Deep, Manoj Thakur, 2007, *A new mutation operator for real coded genetic algorithms.*

Dr. E. Elamin Elnima Ali, 2006, *A Proposed Genetic Algorithm Selection Method*.

Akyol Sinem ve Alataş Bilal, 2012, Güncel *Sürü Zekâsı Optimizasyon Algoritmaları*.

Chen Ling, Sun Hai-Ying, Wang Shu, 2009, *A parallel ant colony algorithm on massively parallel processors and its convergence analysis for the travelling salesman problem.*

R.F. Tavares Neto n, M. Godinho Filho, 2011, Literature *review regarding Ant Colony Optimization applied to scheduling problems: Guidelines for implementation and directions for future research.*

Daniel Merkle, Martin Middendorf, 2000, *An Ant Algorithm with a New Pheromone Evaluation Rule for Total Tardiness Problems*.

Timur Keskinturk, MehmetB. Yildirim, MehmetBarut, 2010, *An ant colony optimization algorithm for load balancing in parallel machines with sequence-dependent setup times*.

Xiao-Lan Zhuo, Jun Zhang, MIEEE ve Wei-neng Cheng, 2007, *A New Pheromone Design in ACS for Solving JSP*.

Chandrasekharan Rajendran, Hans Ziegler, 2001, *Ant-colony algorithms for permutation flowshop scheduling to minimize makespan total flowtime of jobs*.

J. Heinonen, F. Pettersson, 2007, *Hybrid ant colony optimization and visibility studies applied to a job-shop scheduling problem*.

Marco Dorigo, Luca Maria Gambardella, 1996, *Ant colonies for the traveling salesman problem*.

***Diğer Yayınlar***

Michael R. Garey ve David S. Johnson, *NP-hard*, http://en.wikipedia.org/wiki/NP-hard [erişim 4 Ocak 2013]

# EKLER

**EK : TEST SONUÇLARI**

Genetik algoritma 1. veri seti 1. test sonuçları

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 126 | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 16 | |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 1 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 17:00:00 | 2013.01.02 06:00:00 | 1 | |
| | 2 | 18 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 43 | 2013.01.01 09:00:00 | 2013.01.02 00:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 32 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 3 | 3 | 2013.01.01 21:00:00 | 2013.01.02 06:00:00 | 2 | |
| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
| 2. 126 | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 43 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 42 | 2013.01.05 04:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 1 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 16:00:00 | 2013.01.02 05:00:00 | 1 | |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 16 | |
| | 2 | 3 | 2013.01.01 21:00:00 | 2013.01.02 06:00:00 | 2 | |
| | 3 | 18 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 3 | 32 | 2013.01.01 09:00:00 | 2013.01.01 22:00:00 | 31 | 2013.01.04 08:00:00 |

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 1. 126 | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 16 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 16:00:00 | 2013.01.02 05:00:00 | 16 | |
| | 2 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 3 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 43 | 2013.01.01 17:00:00 | 2013.01.02 06:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 18 | 2013.01.01 00:00:00 | 2013.01.01 11:00:00 | 17 | 2013.01.03 12:00:00 |
| | 3 | 32 | 2013.01.01 11:00:00 | 2013.01.02 02:00:00 | 31 | 2013.01.04 08:00:00 |

Iteration:1 Best Value:109.0

Iteration:2 Best Value:110.0

Iteration:3 Best Value:115.0

Iteration:7 Best Value:126.0

Total Iteration Count:108

1. En iyi değerin amaç fonksiyon parametreleri

TotalProcessCompleteTime: 107.0 | AllOperationTardiness: 28.0 | AvgOperationsTardiness: 0.2616822429906542 | FlowTime: 30.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 15.0 | SetupCost: 810000.0 | OperationCost: 3312000.0 | TotalOperationCost: 4122000.0 | DiffJobEndDelivery: 0.0 | Earliness: 186.0 | ScheduledPercentage: 1.0

Genetik algoritma 1. veri seti 2. test sonuçları

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 128 | 0 | 21 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 35 | 2013.01.01 09:00:00 | 2013.01.01 22:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 22 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 23 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 16 | |
| | 1 | 9 | 2013.01.01 21:00:00 | 2013.01.02 06:00:00 | 2 | |
| | 2 | 49 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 10 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 2 | 11 | 2013.01.01 17:00:00 | 2013.01.02 05:00:00 | 1 | |
| | 3 | 40 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 52 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 42 | 2013.01.05 04:00:00 |
| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
| 2. 128 | 0 | 45 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 20 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 1 | 36 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 37 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 9 | 2013.01.01 21:00:00 | 2013.01.02 06:00:00 | 2 | |
| | 2 | 27 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 50 | 2013.01.01 09:00:00 | 2013.01.01 22:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 28 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 13 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 3 | 14 | 2013.01.01 16:00:00 | 2013.01.02 04:00:00 | 1 | |
| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
| 1. 130 | 0 | 6 | 2013.01.01 00:00:00 | 2013.01.01 11:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 45 | 2013.01.01 11:00:00 | 2013.01.01 19:00:00 | 0 | |
| | 0 | 46 | 2013.01.01 19:00:00 | 2013.01.02 08:00:00 | 42 | 2013.01.05 04:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 24 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 37 | 2013.01.01 09:00:00 | 2013.01.02 00:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 38 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 11 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 1 | |
| | 3 | 13 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 28 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 3 | 29 | 2013.01.01 16:00:00 | 2013.01.02 06:00:00 | 16 | |

Iteration:1 Best Value:112.0
Iteration:4 Best Value:114.0
Iteration:10 Best Value:128.0
Iteration:22 Best Value:130.0
Total Iteration Count:123

1. En iyi değerin amaç fonksiyon parametreleri

TotalProcessCompleteTime: 107.0 | AllOperationTardiness: 20.0 | AvgOperationsTardiness: 0.18691588785046728 | FlowTime: 32.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 15.0 | SetupCost: 810000.0 | OperationCost: 3312000.0 | TotalOperationCost: 4122000.0 | DiffJobEndDelivery: 0.0 | Earliness: 194.0 | ScheduledPercentage: 1.0

Genetik algoritma 1. veri seti 3. test sonuçları

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 127 | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 17:00:00 | 2013.01.02 06:00:00 | 16 | |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 43 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 3 | 3 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 18 | 2013.01.01 09:00:00 | 2013.01.01 18:00:00 | 17 | 2013.01.03 12:00:00 |
| | 3 | 32 | 2013.01.01 18:00:00 | 2013.01.02 07:00:00 | 31 | 2013.01.04 08:00:00 |
| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
| 2. 127 | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 32 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 18 | 2013.01.01 21:00:00 | 2013.01.02 06:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 3 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 43 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 17:00:00 | 2013.01.02 06:00:00 | 16 | |
| | 3 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
| 1. 129 | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 32 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 16 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 16:00:00 | 2013.01.02 05:00:00 | 16 | |
| | 2 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 3 | 3 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 18 | 2013.01.01 09:00:00 | 2013.01.01 18:00:00 | 17 | 2013.01.03 12:00:00 |
| | 3 | 43 | 2013.01.01 18:00:00 | 2013.01.02 07:00:00 | 42 | 2013.01.05 04:00:00 |

Iteration:1 Best Value:110.0
Iteration:3 Best Value:112.0
Iteration:5 Best Value:119.0
Iteration:27 Best Value:123.0
Iteration:66 Best Value:124.0
Iteration:106 Best Value:129.0
Total Iteration Count:207

TotalProcessCompleteTime: 103.0 | AllOperationTardiness: 27.0 | AvgOperationsTardiness: 0.2621359223300971 | FlowTime: 31.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 11.0 | SetupCost: 594000.0 | OperationCost: 3312000.0 | TotalOperationCost: 3906000.0 | DiffJobEndDelivery: 0.0 | Earliness: 191.0 | ScheduledPercentage: 1.0

Genetik algoritma 2. veri seti 1. test sonuçları

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 231 | 0 | 105 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 0 | 53 | 2013.01.01 22:00:00 | 2013.01.02 17:00:00 | 0 | |
| | 0 | 54 | 2013.01.02 17:00:00 | 2013.01.03 20:00:00 | 53 | |
| | 0 | 55 | 2013.01.03 20:00:00 | 2013.01.04 15:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 69 | 2013.01.04 15:00:00 | 2013.01.05 18:00:00 | 68 | 2013.01.07 06:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 32 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 94 | 2013.01.01 23:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 1 | 95 | 2013.01.03 11:00:00 | 2013.01.06 02:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 2 | 16 | 2013.01.01 15:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 2 | 42 | 2013.01.01 23:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 2 | 17 | 2013.01.02 07:00:00 | 2013.01.02 20:00:00 | 16 | |
| | 2 | 43 | 2013.01.02 20:00:00 | 2013.01.03 10:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 68 | 2013.01.03 10:00:00 | 2013.01.04 01:00:00 | 0 | |
| | 2 | 106 | 2013.01.04 01:00:00 | 2013.01.05 16:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 3 | 3 | 2013.01.01 22:00:00 | 2013.01.02 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 18 | 2013.01.02 09:00:00 | 2013.01.02 20:00:00 | 17 | 2013.01.03 12:00:00 |
| | 3 | 80 | 2013.01.02 20:00:00 | 2013.01.03 23:00:00 | 79 | |
| | 3 | 81 | 2013.01.03 23:00:00 | 2013.01.04 18:00:00 | 80 | 2013.01.08 12:00:00 |
| | | | | | | |
| 2. 233 | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 3 | 2013.01.01 08:00:00 | 2013.01.01 19:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 42 | 2013.01.01 19:00:00 | 2013.01.02 03:00:00 | 0 | |
| | 0 | 79 | 2013.01.02 03:00:00 | 2013.01.02 19:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 80 | 2013.01.02 19:00:00 | 2013.01.03 20:00:00 | 79 | |
| | 0 | 81 | 2013.01.03 20:00:00 | 2013.01.04 15:00:00 | 80 | 2013.01.08 12:00:00 |
| | 0 | 55 | 2013.01.04 15:00:00 | 2013.01.05 10:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 105 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 1 | 53 | 2013.01.01 22:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 1 | 43 | 2013.01.02 14:00:00 | 2013.01.03 05:00:00 | 42 | 2013.01.05 04:00:00 |
| | 1 | 32 | 2013.01.03 05:00:00 | 2013.01.03 18:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 54 | 2013.01.03 18:00:00 | 2013.01.04 19:00:00 | 53 | |
| | 2 | 68 | 2013.01.01 00:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 2 | 16 | 2013.01.01 17:00:00 | 2013.01.02 02:00:00 | 0 | |
| | 2 | 17 | 2013.01.02 02:00:00 | 2013.01.02 16:00:00 | 16 | |
| | 2 | 18 | 2013.01.02 16:00:00 | 2013.01.03 02:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 69 | 2013.01.03 02:00:00 | 2013.01.04 05:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 106 | 2013.01.04 05:00:00 | 2013.01.05 20:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 3 | 94 | 2013.01.01 22:00:00 | 2013.01.03 10:00:00 | 0 | |
| | 3 | 95 | 2013.01.03 10:00:00 | 2013.01.06 01:00:00 | 94 | 2013.01.09 18:00:00 |
| | | | | | | |
| | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 42 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 0 | 43 | 2013.01.01 16:00:00 | 2013.01.02 07:00:00 | 42 | 2013.01.05 04:00:00 |
| | 0 | 69 | 2013.01.02 07:00:00 | 2013.01.03 09:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 54 | 2013.01.03 09:00:00 | 2013.01.04 10:00:00 | 53 | |
| | 0 | 81 | 2013.01.04 10:00:00 | 2013.01.05 05:00:00 | 80 | 2013.01.08 12:00:00 |
| | 1 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 16 | |
| 1. 240 | 1 | 18 | 2013.01.01 21:00:00 | 2013.01.02 07:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 53 | 2013.01.02 07:00:00 | 2013.01.03 01:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 80 | 2013.01.03 01:00:00 | 2013.01.04 02:00:00 | 79 | |
| | 1 | 55 | 2013.01.04 02:00:00 | 2013.01.04 21:00:00 | 54 | 2013.01.06 10:00:00 |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 68 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 2 | 3 | 2013.01.01 23:00:00 | 2013.01.02 10:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 105 | 2013.01.02 10:00:00 | 2013.01.03 09:00:00 | 0 | |
| | 2 | 32 | 2013.01.03 09:00:00 | 2013.01.04 00:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 106 | 2013.01.04 00:00:00 | 2013.01.05 15:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 15:00:00 | 2013.01.02 05:00:00 | 1 | |
| | 3 | 94 | 2013.01.02 05:00:00 | 2013.01.03 17:00:00 | 0 | |
| | 3 | 95 | 2013.01.03 17:00:00 | 2013.01.06 08:00:00 | 94 | 2013.01.09 18:00:00 |

Iteration:1 Best Value:149.0
Iteration:5 Best Value:193.0
Iteration:17 Best Value:214.0
Iteration:20 Best Value:250.0
Total Iteration Count:121

TotalProcessCompleteTime: 433.0 | AllOperationTardiness: 455.0 | AvgOperationsTardiness: 1.0508083140877598 | FlowTime: 123.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 43.0 | SetupCost: 2322000.0 | OperationCost: 1.404E7 | TotalOperationCost: 1.6362E7 | DiffJobEndDelivery: 1.0 | Earliness: 501.0 | ScheduledPercentage: 1.0

Genetik algoritma 2. veri seti 2. test sonuçları

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 243 | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 32 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 3 | 2013.01.01 23:00:00 | 2013.01.02 10:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 79 | 2013.01.02 10:00:00 | 2013.01.03 01:00:00 | 0 | |
| | 0 | 54 | 2013.01.03 01:00:00 | 2013.01.04 03:00:00 | 53 | |
| | 1 | 106 | 2013.01.04 03:00:00 | 2013.01.05 17:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 42 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 16:00:00 | 2013.01.02 05:00:00 | 16 | |
| | 2 | 18 | 2013.01.02 05:00:00 | 2013.01.02 16:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 69 | 2013.01.02 16:00:00 | 2013.01.03 19:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 43 | 2013.01.03 19:00:00 | 2013.01.04 08:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 81 | 2013.01.04 08:00:00 | 2013.01.05 03:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 53 | 2013.01.01 00:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 2 | 94 | 2013.01.01 16:00:00 | 2013.01.03 04:00:00 | 0 | |
| | 2 | 80 | 2013.01.03 04:00:00 | 2013.01.04 05:00:00 | 79 | |
| | 3 | 55 | 2013.01.04 05:00:00 | 2013.01.04 23:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 68 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 23:00:00 | 2013.01.02 13:00:00 | 1 | |
| | 3 | 105 | 2013.01.02 13:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 3 | 95 | 2013.01.03 11:00:00 | 2013.01.06 00:00:00 | 94 | 2013.01.09 18:00:00 |
| | | | | | | Sipariş Teslimi |
| 2. 246 | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 42 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 16:00:00 | 2013.01.02 05:00:00 | 16 | |
| | 0 | 53 | 2013.01.02 05:00:00 | 2013.01.03 00:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 32 | 2013.01.03 00:00:00 | 2013.01.03 15:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 54 | 2013.01.03 15:00:00 | 2013.01.04 16:00:00 | 53 | |
| | 0 | 106 | 2013.01.04 16:00:00 | 2013.01.06 06:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 1 | |
| | 1 | 3 | 2013.01.01 23:00:00 | 2013.01.02 08:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 18 | 2013.01.02 08:00:00 | 2013.01.02 19:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 95 | 2013.01.02 19:00:00 | 2013.01.05 10:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 2 | 31 | 2013.01.01 15:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 2 | 69 | 2013.01.01 23:00:00 | 2013.01.03 00:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 105 | 2013.01.03 00:00:00 | 2013.01.03 22:00:00 | 0 | |
| | 2 | 81 | 2013.01.03 22:00:00 | 2013.01.04 17:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 55 | 2013.01.04 17:00:00 | 2013.01.05 12:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 94 | 2013.01.01 00:00:00 | 2013.01.02 12:00:00 | 0 | |
| | 3 | 43 | 2013.01.02 12:00:00 | 2013.01.03 01:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 79 | 2013.01.03 01:00:00 | 2013.01.03 17:00:00 | 0 | |
| | 3 | 80 | 2013.01.03 17:00:00 | 2013.01.04 19:00:00 | 79 | |
| | | | | | | Sipariş Teslimi |
| | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 94 | 2013.01.01 08:00:00 | 2013.01.02 20:00:00 | 0 | |
| | 0 | 18 | 2013.01.02 20:00:00 | 2013.01.03 07:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 54 | 2013.01.03 07:00:00 | 2013.01.04 08:00:00 | 53 | |
| | 0 | 106 | 2013.01.04 08:00:00 | 2013.01.05 23:00:00 | 105 | 2013.01.10 14:00:00 |
| | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 1 | 3 | 2013.01.01 22:00:00 | 2013.01.02 07:00:00 | 2 | 2013.01.02 06:00:00 |
| 1. 250 | 1 | 17 | 2013.01.02 07:00:00 | 2013.01.02 22:00:00 | 16 | |
| | 1 | 80 | 2013.01.02 22:00:00 | 2013.01.03 22:00:00 | 79 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 69 | 2013.01.03 22:00:00 | 2013.01.05 01:00:00 | 68 | 2013.01.07 06:00:00 |
| | 1 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 2 | 53 | 2013.01.01 15:00:00 | 2013.01.02 10:00:00 | 0 | |
| | 2 | 105 | 2013.01.02 10:00:00 | 2013.01.03 08:00:00 | 0 | |
| | 2 | 55 | 2013.01.03 08:00:00 | 2013.01.04 03:00:00 | 54 | 2013.01.06 10:00:00 |
| | 2 | 81 | 2013.01.04 03:00:00 | 2013.01.04 22:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 42 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 3 | 43 | 2013.01.01 16:00:00 | 2013.01.02 07:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 32 | 2013.01.02 07:00:00 | 2013.01.02 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 3 | 68 | 2013.01.02 21:00:00 | 2013.01.03 13:00:00 | 0 | |
| | 3 | 95 | 2013.01.03 13:00:00 | 2013.01.06 03:00:00 | 94 | 2013.01.09 18:00:00 |

Iteration:1 Best Value:96.0
Iteration:3 Best Value:170.0
Iteration:8 Best Value:207.0
Iteration:41 Best Value:213.0
Iteration:47 Best Value:231.0
Iteration:85 Best Value:240.0
Total Iteration Count:186

TotalProcessCompleteTime: 433.0 | AllOperationTardiness: 431.0 | AvgOperationsTardiness: 0.9953810623556582 | FlowTime: 128.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 43.0 | SetupCost: 2322000.0 | OperationCost: 1.404E7 | TotalOperationCost: 1.6362E7 | DiffJobEndDelivery: 4.0 | Earliness: 516.0 | ScheduledPercentage: 1.0

Genetik algoritma 2. veri seti 3. test sonuçları

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 254 | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 43 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 42 | 2013.01.05 04:00:00 |
| | 0 | 31 | 2013.01.01 21:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 0 | 69 | 2013.01.02 06:00:00 | 2013.01.03 07:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 105 | 2013.01.03 07:00:00 | 2013.01.04 06:00:00 | 0 | |
| | 0 | 106 | 2013.01.04 06:00:00 | 2013.01.05 20:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 1 | 53 | 2013.01.01 22:00:00 | 2013.01.02 17:00:00 | 0 | |
| | 1 | 32 | 2013.01.02 17:00:00 | 2013.01.03 08:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 80 | 2013.01.03 08:00:00 | 2013.01.04 09:00:00 | 79 | |
| | 1 | 81 | 2013.01.04 09:00:00 | 2013.01.05 04:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 2 | 2 | 2013.01.01 15:00:00 | 2013.01.02 04:00:00 | 1 | |
| | 2 | 3 | 2013.01.02 04:00:00 | 2013.01.02 13:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 18 | 2013.01.02 13:00:00 | 2013.01.02 22:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 54 | 2013.01.02 22:00:00 | 2013.01.04 00:00:00 | 53 | |
| | 2 | 55 | 2013.01.04 00:00:00 | 2013.01.04 19:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 68 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 3 | 94 | 2013.01.01 23:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 3 | 95 | 2013.01.03 11:00:00 | 2013.01.06 02:00:00 | 94 | 2013.01.09 18:00:00 |
| | | | | | | |
| 2. 273 | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 08:00:00 | 2013.01.01 20:00:00 | 1 | |
| | 0 | 3 | 2013.01.01 20:00:00 | 2013.01.02 06:00:00 | 2 | |
| | 0 | 105 | 2013.01.02 06:00:00 | 2013.01.03 05:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 54 | 2013.01.03 05:00:00 | 2013.01.04 06:00:00 | 53 | |
| | 0 | 106 | 2013.01.04 06:00:00 | 2013.01.05 20:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 1 | 53 | 2013.01.01 15:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 1 | 18 | 2013.01.02 06:00:00 | 2013.01.02 17:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 80 | 2013.01.02 17:00:00 | 2013.01.03 18:00:00 | 79 | |
| | 1 | 55 | 2013.01.03 18:00:00 | 2013.01.04 13:00:00 | 54 | 2013.01.06 10:00:00 |
| | 2 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 43 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 17 | 2013.01.01 21:00:00 | 2013.01.02 11:00:00 | 16 | |
| | 2 | 31 | 2013.01.02 11:00:00 | 2013.01.02 21:00:00 | 0 | |
| | 2 | 32 | 2013.01.02 21:00:00 | 2013.01.03 11:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 95 | 2013.01.03 11:00:00 | 2013.01.06 00:00:00 | 94 | 2013.01.09 18:00:00 |
| | 3 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 68 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 3 | 94 | 2013.01.01 23:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 3 | 69 | 2013.01.03 11:00:00 | 2013.01.04 12:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 81 | 2013.01.04 12:00:00 | 2013.01.05 05:00:00 | 80 | 2013.01.08 12:00:00 |
| | | | | | | |
| | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 43 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 42 | 2013.01.05 04:00:00 |
| | 0 | 3 | 2013.01.01 21:00:00 | 2013.01.02 07:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 106 | 2013.01.02 07:00:00 | 2013.01.03 22:00:00 | 105 | 2013.01.10 14:00:00 |
| | 0 | 80 | 2013.01.03 22:00:00 | 2013.01.04 23:00:00 | 79 | |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 1 | |
| | 1 | 31 | 2013.01.01 23:00:00 | 2013.01.02 07:00:00 | 0 | |
| 1. 274 | 1 | 18 | 2013.01.02 07:00:00 | 2013.01.02 16:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 32 | 2013.01.02 16:00:00 | 2013.01.03 07:00:00 | 31 | 2013.01.04 08:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 54 | 2013.01.03 07:00:00 | 2013.01.04 07:00:00 | 53 | |
| | 1 | 81 | 2013.01.04 07:00:00 | 2013.01.05 00:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 2 | 105 | 2013.01.01 09:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 2 | 94 | 2013.01.02 07:00:00 | 2013.01.03 19:00:00 | 0 | |
| | 2 | 95 | 2013.01.03 19:00:00 | 2013.01.06 10:00:00 | 94 | 2013.01.09 18:00:00 |
| | 3 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 17 | 2013.01.01 15:00:00 | 2013.01.02 05:00:00 | 16 | |
| | 3 | 53 | 2013.01.02 05:00:00 | 2013.01.03 00:00:00 | 0 | |
| | 3 | 68 | 2013.01.03 00:00:00 | 2013.01.03 15:00:00 | 0 | |
| | 3 | 55 | 2013.01.03 15:00:00 | 2013.01.04 10:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 69 | 2013.01.04 10:00:00 | 2013.01.05 11:00:00 | 68 | 2013.01.07 06:00:00 |

Iteration:1 Best Value:145.0
Iteration:5 Best Value:231.0
Iteration:49 Best Value:239.0
Iteration:122 Best Value:251.0
Iteration:188 Best Value:254.0
Iteration:195 Best Value:274.0
Total Iteration Count:296

TotalProcessCompleteTime: 428.0 | AllOperationTardiness: 430.0 | AvgOperationsTardiness: 1.0046728971962617 | FlowTime: 130.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 38.0 | SetupCost: 2052000.0 | OperationCost: 1.404E7 | TotalOperationCost: 1.6092E7 | DiffJobEndDelivery: 1.0 | Earliness: 539.0 | ScheduledPercentage: 1.0

Genetik algoritma 3. veri seti 1. test sonuçları

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 116 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 0 | 17 | 2013.01.02 06:00:00 | 2013.01.02 19:00:00 | 16 | |
| | 0 | 18 | 2013.01.02 19:00:00 | 2013.01.03 05:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 131 | 2013.01.03 05:00:00 | 2013.01.04 04:00:00 | 0 | |
| | 0 | 143 | 2013.01.04 04:00:00 | 2013.01.05 17:00:00 | 142 | |
| | 0 | 55 | 2013.01.05 17:00:00 | 2013.01.06 12:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 144 | 2013.01.06 12:00:00 | 2013.01.07 13:00:00 | 143 | 2013.01.13 02:00:00 |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 79 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 1 | 31 | 2013.01.01 23:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 1 | 80 | 2013.01.02 07:00:00 | 2013.01.03 08:00:00 | 79 | |
| | 1 | 3 | 2013.01.03 08:00:00 | 2013.01.03 19:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 81 | 2013.01.03 19:00:00 | 2013.01.04 13:00:00 | 80 | 2013.01.08 12:00:00 |
| | 1 | 54 | 2013.01.04 13:00:00 | 2013.01.05 15:00:00 | 53 | |
| | 1 | 132 | 2013.01.05 15:00:00 | 2013.01.07 06:00:00 | 131 | 2013.01.12 06:00:00 |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 142 | 2013.01.01 08:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 2 | 2 | 2013.01.02 07:00:00 | 2013.01.02 21:00:00 | 1 | |
| | 2 | 105 | 2013.01.02 21:00:00 | 2013.01.03 19:00:00 | 0 | |
| | 2 | 117 | 2013.01.03 19:00:00 | 2013.01.05 08:00:00 | 116 | |
| | 2 | 106 | 2013.01.05 08:00:00 | 2013.01.06 22:00:00 | 105 | 2013.01.10 14:00:00 |
| | 2 | 118 | 2013.01.06 22:00:00 | 2013.01.08 01:00:00 | 117 | 2013.01.11 10:00:00 |
| | 3 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 53 | 2013.01.01 15:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 3 | 43 | 2013.01.02 07:00:00 | 2013.01.02 20:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 32 | 2013.01.02 20:00:00 | 2013.01.03 09:00:00 | 31 | 2013.01.04 08:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 94 | 2013.01.03 09:00:00 | 2013.01.04 21:00:00 | 0 | |
| | 3 | 69 | 2013.01.04 21:00:00 | 2013.01.05 23:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 95 | 2013.01.05 23:00:00 | 2013.01.08 12:00:00 | 94 | 2013.01.09 18:00:00 |
| | | | | | | |
| | | | | | | |
| | 0 | 105 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 0 | 31 | 2013.01.01 22:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 0 | 54 | 2013.01.02 06:00:00 | 2013.01.03 08:00:00 | 53 | |
| | 0 | 55 | 2013.01.03 08:00:00 | 2013.01.04 01:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 142 | 2013.01.04 01:00:00 | 2013.01.05 00:00:00 | 0 | |
| | 0 | 32 | 2013.01.05 00:00:00 | 2013.01.05 13:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 69 | 2013.01.05 13:00:00 | 2013.01.06 14:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 118 | 2013.01.06 14:00:00 | 2013.01.07 17:00:00 | 117 | 2013.01.11 10:00:00 |
| | 1 | 53 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 1 | 1 | 2013.01.01 15:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 23:00:00 | 2013.01.02 13:00:00 | 1 | |
| | 1 | 3 | 2013.01.02 13:00:00 | 2013.01.02 22:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 18 | 2013.01.02 22:00:00 | 2013.01.03 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 79 | 2013.01.03 09:00:00 | 2013.01.04 03:00:00 | 0 | |
| | 1 | 80 | 2013.01.04 03:00:00 | 2013.01.05 04:00:00 | 79 | |
| | 1 | 95 | 2013.01.05 04:00:00 | 2013.01.07 19:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 116 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 22:00:00 | 2013.01.02 12:00:00 | 16 | |
| | 2 | 132 | 2013.01.02 12:00:00 | 2013.01.04 02:00:00 | 131 | 2013.01.12 06:00:00 |
| | 2 | 43 | 2013.01.04 02:00:00 | 2013.01.04 17:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 68 | 2013.01.04 17:00:00 | 2013.01.05 08:00:00 | 0 | |
| | 2 | 117 | 2013.01.05 08:00:00 | 2013.01.06 21:00:00 | 116 | |
| | 2 | 144 | 2013.01.06 21:00:00 | 2013.01.08 00:00:00 | 143 | 2013.01.13 02:00:00 |
| | 3 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 131 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 3 | 94 | 2013.01.02 06:00:00 | 2013.01.03 18:00:00 | 0 | |
| | 3 | 42 | 2013.01.03 18:00:00 | 2013.01.04 02:00:00 | 0 | |
| | 3 | 106 | 2013.01.04 02:00:00 | 2013.01.05 15:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 81 | 2013.01.05 15:00:00 | 2013.01.06 10:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 143 | 2013.01.06 10:00:00 | 2013.01.08 01:00:00 | 142 | |
| | | | | | | |
| | | | | | | |
| | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 0 | 31 | 2013.01.01 22:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 0 | 32 | 2013.01.02 06:00:00 | 2013.01.02 19:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 17 | 2013.01.02 19:00:00 | 2013.01.03 09:00:00 | 16 | |
| | 0 | 18 | 2013.01.03 09:00:00 | 2013.01.03 18:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 80 | 2013.01.03 18:00:00 | 2013.01.04 20:00:00 | 79 | |
| | 0 | 131 | 2013.01.04 20:00:00 | 2013.01.05 18:00:00 | 0 | |
| | 0 | 132 | 2013.01.05 18:00:00 | 2013.01.07 08:00:00 | 131 | 2013.01.12 06:00:00 |
| | 1 | 105 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 1 | 79 | 2013.01.01 22:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 1 | 94 | 2013.01.02 14:00:00 | 2013.01.04 02:00:00 | 0 | |
| | 1 | 106 | 2013.01.04 02:00:00 | 2013.01.05 17:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 95 | 2013.01.05 17:00:00 | 2013.01.08 08:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 142 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 2 | 53 | 2013.01.01 22:00:00 | 2013.01.02 13:00:00 | 0 | |
| | 2 | 116 | 2013.01.02 13:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 2 | 117 | 2013.01.03 11:00:00 | 2013.01.05 01:00:00 | 116 | |
| | 2 | 118 | 2013.01.05 01:00:00 | 2013.01.06 04:00:00 | 117 | 2013.01.11 10:00:00 |
| | 2 | 55 | 2013.01.06 04:00:00 | 2013.01.06 21:00:00 | 54 | 2013.01.06 10:00:00 |
| | 2 | 144 | 2013.01.06 21:00:00 | 2013.01.08 00:00:00 | 143 | 2013.01.13 02:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 3 | 68 | 2013.01.01 17:00:00 | 2013.01.02 08:00:00 | 0 | |
| | 3 | 3 | 2013.01.02 08:00:00 | 2013.01.02 19:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 69 | 2013.01.02 19:00:00 | 2013.01.03 22:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 43 | 2013.01.03 22:00:00 | 2013.01.04 12:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 143 | 2013.01.04 12:00:00 | 2013.01.06 02:00:00 | 142 | |
| | 3 | 54 | 2013.01.06 02:00:00 | 2013.01.07 03:00:00 | 53 | |
| | 3 | 81 | 2013.01.07 03:00:00 | 2013.01.07 20:00:00 | 80 | 2013.01.08 12:00:00 |

Iteration:1 Best Value:-167.0
Iteration:3 Best Value:-127.0
Iteration:5 Best Value:-79.0
Iteration:6 Best Value:78.0
Iteration:20 Best Value:164.0
Total Iteration Count:121

TotalProcessCompleteTime: 660.0 | AllOperationTardiness: 1221.0 | AvgOperationsTardiness: 1.85 | FlowTime: 176.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 51.0 | SetupCost: 2754000.0 | OperationCost: 2.1924E7 | TotalOperationCost: 2.4678E7 | DiffJobEndDelivery: 30.0 | Earliness: 666.0 | ScheduledPercentage: 1.0

Genetik algoritma 3. veri seti 2. test sonuçları

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 17:00:00 | 2013.01.02 07:00:00 | 16 | |
| | 0 | 18 | 2013.01.02 07:00:00 | 2013.01.02 18:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 68 | 2013.01.02 18:00:00 | 2013.01.03 09:00:00 | 0 | |
| | 0 | 54 | 2013.01.03 09:00:00 | 2013.01.04 10:00:00 | 53 | |
| | 0 | 55 | 2013.01.04 10:00:00 | 2013.01.05 04:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 69 | 2013.01.05 04:00:00 | 2013.01.06 05:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 81 | 2013.01.06 05:00:00 | 2013.01.06 22:00:00 | 80 | 2013.01.08 12:00:00 |
| | 0 | 144 | 2013.01.06 22:00:00 | 2013.01.08 01:00:00 | 143 | 2013.01.13 02:00:00 |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 1 | 53 | 2013.01.01 09:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 1 | 32 | 2013.01.02 00:00:00 | 2013.01.02 13:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 79 | 2013.01.02 13:00:00 | 2013.01.03 06:00:00 | 0 | |
| | 1 | 2 | 2013.01.03 06:00:00 | 2013.01.03 19:00:00 | 1 | |
| | 1 | 3 | 2013.01.03 19:00:00 | 2013.01.04 04:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 142 | 2013.01.04 04:00:00 | 2013.01.05 02:00:00 | 0 | |
| | 1 | 106 | 2013.01.05 02:00:00 | 2013.01.06 17:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 143 | 2013.01.06 17:00:00 | 2013.01.08 05:00:00 | 142 | |
| | 2 | 131 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 2 | 42 | 2013.01.01 22:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 2 | 132 | 2013.01.02 06:00:00 | 2013.01.03 21:00:00 | 131 | 2013.01.12 06:00:00 |
| | 2 | 43 | 2013.01.03 21:00:00 | 2013.01.04 11:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 117 | 2013.01.04 11:00:00 | 2013.01.05 23:00:00 | 116 | |
| | 2 | 118 | 2013.01.05 23:00:00 | 2013.01.07 00:00:00 | 117 | 2013.01.11 10:00:00 |
| | 3 | 105 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 3 | 94 | 2013.01.01 22:00:00 | 2013.01.03 10:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 116 | 2013.01.03 10:00:00 | 2013.01.04 08:00:00 | 0 | |
| | 3 | 80 | 2013.01.04 08:00:00 | 2013.01.05 09:00:00 | 79 | |
| | 3 | 95 | 2013.01.05 09:00:00 | 2013.01.07 22:00:00 | 94 | 2013.01.09 18:00:00 |
| | | | | | | |
| | | | | | | |
| | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 1 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 0 | 142 | 2013.01.01 17:00:00 | 2013.01.02 16:00:00 | 0 | |
| | 0 | 105 | 2013.01.02 16:00:00 | 2013.01.03 14:00:00 | 0 | |
| | 0 | 94 | 2013.01.03 14:00:00 | 2013.01.05 04:00:00 | 0 | |
| | 0 | 143 | 2013.01.05 04:00:00 | 2013.01.06 18:00:00 | 142 | |
| | 0 | 144 | 2013.01.06 18:00:00 | 2013.01.07 20:00:00 | 143 | 2013.01.13 02:00:00 |
| | 1 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 31 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 16:00:00 | 2013.01.02 05:00:00 | 16 | |
| | 1 | 18 | 2013.01.02 05:00:00 | 2013.01.02 15:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 32 | 2013.01.02 15:00:00 | 2013.01.03 06:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 3 | 2013.01.03 06:00:00 | 2013.01.03 16:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 43 | 2013.01.03 16:00:00 | 2013.01.04 06:00:00 | 42 | 2013.01.05 04:00:00 |
| | 1 | 69 | 2013.01.04 06:00:00 | 2013.01.05 07:00:00 | 68 | 2013.01.07 06:00:00 |
| | 1 | 95 | 2013.01.05 07:00:00 | 2013.01.07 22:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 2 | 79 | 2013.01.01 15:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 2 | 2 | 2013.01.02 06:00:00 | 2013.01.02 20:00:00 | 1 | |
| | 2 | 80 | 2013.01.02 20:00:00 | 2013.01.03 21:00:00 | 79 | |
| | 2 | 81 | 2013.01.03 21:00:00 | 2013.01.04 14:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 117 | 2013.01.04 14:00:00 | 2013.01.06 05:00:00 | 116 | |
| | 2 | 118 | 2013.01.06 05:00:00 | 2013.01.07 06:00:00 | 117 | 2013.01.11 10:00:00 |
| | 3 | 116 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 53 | 2013.01.01 22:00:00 | 2013.01.02 13:00:00 | 0 | |
| | 3 | 131 | 2013.01.02 13:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 3 | 54 | 2013.01.03 11:00:00 | 2013.01.04 12:00:00 | 53 | |
| | 3 | 55 | 2013.01.04 12:00:00 | 2013.01.05 05:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 106 | 2013.01.05 05:00:00 | 2013.01.06 18:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 132 | 2013.01.06 18:00:00 | 2013.01.08 09:00:00 | 131 | 2013.01.12 06:00:00 |
| | | | | | | |
| | | | | | | |
| | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 79 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 0 | 142 | 2013.01.01 23:00:00 | 2013.01.02 21:00:00 | 0 | |
| | 0 | 94 | 2013.01.02 21:00:00 | 2013.01.04 09:00:00 | 0 | |
| | 0 | 143 | 2013.01.04 09:00:00 | 2013.01.05 21:00:00 | 142 | |
| | 0 | 144 | 2013.01.05 21:00:00 | 2013.01.07 00:00:00 | 143 | 2013.01.13 02:00:00 |
| | 0 | 81 | 2013.01.07 00:00:00 | 2013.01.07 17:00:00 | 80 | 2013.01.08 12:00:00 |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 116 | 2013.01.01 08:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 1 | 54 | 2013.01.02 07:00:00 | 2013.01.03 09:00:00 | 53 | |
| | 1 | 18 | 2013.01.03 09:00:00 | 2013.01.03 18:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 80 | 2013.01.03 18:00:00 | 2013.01.04 18:00:00 | 79 | |
| | 1 | 95 | 2013.01.04 18:00:00 | 2013.01.07 08:00:00 | 94 | 2013.01.09 18:00:00 |
| | 1 | 118 | 2013.01.07 08:00:00 | 2013.01.08 11:00:00 | 117 | 2013.01.11 10:00:00 |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 53 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 2 | 2 | 2013.01.01 23:00:00 | 2013.01.02 13:00:00 | 1 | |
| | 2 | 3 | 2013.01.02 13:00:00 | 2013.01.03 00:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 17 | 2013.01.03 00:00:00 | 2013.01.03 15:00:00 | 16 | |
| | 2 | 117 | 2013.01.03 15:00:00 | 2013.01.05 05:00:00 | 116 | |
| | 2 | 69 | 2013.01.05 05:00:00 | 2013.01.06 08:00:00 | 68 | 2013.01.07 06:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 2 | 106 | 2013.01.06 08:00:00 | 2013.01.07 21:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 32 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 3 | 131 | 2013.01.01 21:00:00 | 2013.01.02 19:00:00 | 0 | |
| | 3 | 43 | 2013.01.02 19:00:00 | 2013.01.03 08:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 55 | 2013.01.03 08:00:00 | 2013.01.04 03:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 68 | 2013.01.04 03:00:00 | 2013.01.04 19:00:00 | 0 | |
| | 3 | 105 | 2013.01.04 19:00:00 | 2013.01.05 19:00:00 | 0 | |
| | 3 | 132 | 2013.01.05 19:00:00 | 2013.01.07 10:00:00 | 131 | 2013.01.12 06:00:00 |

Iteration:1 Best Value:-156.0
Iteration:5 Best Value:27.0
Iteration:10 Best Value:126.0
Iteration:72 Best Value:135.0
Iteration:89 Best Value:160.0
Iteration:101 Best Value:170.0
Iteration:106 Best Value:177.0
Total Iteration Count:207

TotalProcessCompleteTime: 659.0 | AllOperationTardiness: 1155.0 | AvgOperationsTardiness: 1.7526555386949925 | FlowTime: 179.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 50.0 | SetupCost: 2700000.0 | OperationCost: 2.1924E7 | TotalOperationCost: 2.4624E7 | DiffJobEndDelivery: 24.0 | Earliness: 655.0 | ScheduledPercentage: 1.0

Genetik algoritma 3. veri seti 3. test sonuçları

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 116 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 0 | 17 | 2013.01.02 06:00:00 | 2013.01.02 19:00:00 | 16 | |
| | 0 | 18 | 2013.01.02 19:00:00 | 2013.01.03 05:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 131 | 2013.01.03 05:00:00 | 2013.01.04 04:00:00 | 0 | |
| | 0 | 143 | 2013.01.04 04:00:00 | 2013.01.05 17:00:00 | 142 | |
| | 0 | 55 | 2013.01.05 17:00:00 | 2013.01.06 12:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 144 | 2013.01.06 12:00:00 | 2013.01.07 13:00:00 | 143 | 2013.01.13 02:00:00 |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 79 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 1 | 31 | 2013.01.01 23:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 1 | 80 | 2013.01.02 07:00:00 | 2013.01.03 08:00:00 | 79 | |
| | 1 | 3 | 2013.01.03 08:00:00 | 2013.01.03 19:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 81 | 2013.01.03 19:00:00 | 2013.01.04 13:00:00 | 80 | 2013.01.08 12:00:00 |
| | 1 | 54 | 2013.01.04 13:00:00 | 2013.01.05 15:00:00 | 53 | |
| | 1 | 132 | 2013.01.05 15:00:00 | 2013.01.07 06:00:00 | 131 | 2013.01.12 06:00:00 |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 142 | 2013.01.01 08:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 2 | 2 | 2013.01.02 07:00:00 | 2013.01.02 21:00:00 | 1 | |
| | 2 | 105 | 2013.01.02 21:00:00 | 2013.01.03 19:00:00 | 0 | |
| | 2 | 117 | 2013.01.03 19:00:00 | 2013.01.05 08:00:00 | 116 | |
| | 2 | 106 | 2013.01.05 08:00:00 | 2013.01.06 22:00:00 | 105 | 2013.01.10 14:00:00 |
| | 2 | 118 | 2013.01.06 22:00:00 | 2013.01.08 01:00:00 | 117 | 2013.01.11 10:00:00 |
| | 3 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 53 | 2013.01.01 15:00:00 | 2013.01.02 07:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 43 | 2013.01.02 07:00:00 | 2013.01.02 20:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 32 | 2013.01.02 20:00:00 | 2013.01.03 09:00:00 | 31 | 2013.01.04 08:00:00 |
| | 3 | 94 | 2013.01.03 09:00:00 | 2013.01.04 21:00:00 | 0 | |
| | 3 | 69 | 2013.01.04 21:00:00 | 2013.01.05 23:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 95 | 2013.01.05 23:00:00 | 2013.01.08 12:00:00 | 94 | 2013.01.09 18:00:00 |
| | | | | | | |
| | | | | | | |
| | 0 | 105 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 0 | 31 | 2013.01.01 22:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 0 | 54 | 2013.01.02 06:00:00 | 2013.01.03 08:00:00 | 53 | |
| | 0 | 55 | 2013.01.03 08:00:00 | 2013.01.04 01:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 142 | 2013.01.04 01:00:00 | 2013.01.05 00:00:00 | 0 | |
| | 0 | 32 | 2013.01.05 00:00:00 | 2013.01.05 13:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 69 | 2013.01.05 13:00:00 | 2013.01.06 14:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 118 | 2013.01.06 14:00:00 | 2013.01.07 17:00:00 | 117 | 2013.01.11 10:00:00 |
| | 1 | 53 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 1 | 1 | 2013.01.01 15:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 23:00:00 | 2013.01.02 13:00:00 | 1 | |
| | 1 | 3 | 2013.01.02 13:00:00 | 2013.01.02 22:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 18 | 2013.01.02 22:00:00 | 2013.01.03 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 79 | 2013.01.03 09:00:00 | 2013.01.04 03:00:00 | 0 | |
| | 1 | 80 | 2013.01.04 03:00:00 | 2013.01.05 04:00:00 | 79 | |
| | 1 | 95 | 2013.01.05 04:00:00 | 2013.01.07 19:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 116 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 22:00:00 | 2013.01.02 12:00:00 | 16 | |
| | 2 | 132 | 2013.01.02 12:00:00 | 2013.01.04 02:00:00 | 131 | 2013.01.12 06:00:00 |
| | 2 | 43 | 2013.01.04 02:00:00 | 2013.01.04 17:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 68 | 2013.01.04 17:00:00 | 2013.01.05 08:00:00 | 0 | |
| | 2 | 117 | 2013.01.05 08:00:00 | 2013.01.06 21:00:00 | 116 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 2 | 144 | 2013.01.06 21:00:00 | 2013.01.08 00:00:00 | 143 | 2013.01.13 02:00:00 |
| | 3 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 131 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 3 | 94 | 2013.01.02 06:00:00 | 2013.01.03 18:00:00 | 0 | |
| | 3 | 42 | 2013.01.03 18:00:00 | 2013.01.04 02:00:00 | 0 | |
| | 3 | 106 | 2013.01.04 02:00:00 | 2013.01.05 15:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 81 | 2013.01.05 15:00:00 | 2013.01.06 10:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 143 | 2013.01.06 10:00:00 | 2013.01.08 01:00:00 | 142 | |
| | | | | | | |
| | | | | | | |
| | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 0 | 31 | 2013.01.01 22:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 0 | 32 | 2013.01.02 06:00:00 | 2013.01.02 19:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 17 | 2013.01.02 19:00:00 | 2013.01.03 09:00:00 | 16 | |
| | 0 | 18 | 2013.01.03 09:00:00 | 2013.01.03 18:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 80 | 2013.01.03 18:00:00 | 2013.01.04 20:00:00 | 79 | |
| | 0 | 131 | 2013.01.04 20:00:00 | 2013.01.05 18:00:00 | 0 | |
| | 0 | 132 | 2013.01.05 18:00:00 | 2013.01.07 08:00:00 | 131 | 2013.01.12 06:00:00 |
| | 1 | 105 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 1 | 79 | 2013.01.01 22:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 1 | 94 | 2013.01.02 14:00:00 | 2013.01.04 02:00:00 | 0 | |
| | 1 | 106 | 2013.01.04 02:00:00 | 2013.01.05 17:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 95 | 2013.01.05 17:00:00 | 2013.01.08 08:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 142 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 2 | 53 | 2013.01.01 22:00:00 | 2013.01.02 13:00:00 | 0 | |
| | 2 | 116 | 2013.01.02 13:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 2 | 117 | 2013.01.03 11:00:00 | 2013.01.05 01:00:00 | 116 | |
| | 2 | 118 | 2013.01.05 01:00:00 | 2013.01.06 04:00:00 | 117 | 2013.01.11 10:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 2 | 55 | 2013.01.06 04:00:00 | 2013.01.06 21:00:00 | 54 | 2013.01.06 10:00:00 |
| | 2 | 144 | 2013.01.06 21:00:00 | 2013.01.08 00:00:00 | 143 | 2013.01.13 02:00:00 |
| | 3 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 3 | 68 | 2013.01.01 17:00:00 | 2013.01.02 08:00:00 | 0 | |
| | 3 | 3 | 2013.01.02 08:00:00 | 2013.01.02 19:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 69 | 2013.01.02 19:00:00 | 2013.01.03 22:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 43 | 2013.01.03 22:00:00 | 2013.01.04 12:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 143 | 2013.01.04 12:00:00 | 2013.01.06 02:00:00 | 142 | |
| | 3 | 54 | 2013.01.06 02:00:00 | 2013.01.07 03:00:00 | 53 | |
| | 3 | 81 | 2013.01.07 03:00:00 | 2013.01.07 20:00:00 | 80 | 2013.01.08 12:00:00 |

Iteration:1 Best Value:-118.0
Iteration:7 Best Value:-51.0
Iteration:8 Best Value:60.0
Iteration:12 Best Value:248.0
Total Iteration Count:113

TotalProcessCompleteTime: 660.0 | AllOperationTardiness: 1127.0 | AvgOperationsTardiness: 1.7075757575757575 | FlowTime: 179.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 51.0 | SetupCost: 2754000.0 | OperationCost: 2.1924E7 | TotalOperationCost: 2.4678E7 | DiffJobEndDelivery: 17.0 | Earliness: 691.0 | ScheduledPercentage: 1.0

Karınca kolonisi 1.veri seti 1.test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 127 | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 32 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 2 | 18 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 3 | 2013.01.01 09:00:00 | 2013.01.01 18:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 43 | 2013.01.01 18:00:00 | 2013.01.02 07:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 3 | 17 | 2013.01.01 17:00:00 | 2013.01.02 07:00:00 | 16 | |
| | | | | | | |
| 2. 127 | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 1 | |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 16 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 16:00:00 | 2013.01.02 06:00:00 | 16 | |
| | 2 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 18 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 43 | 2013.01.01 17:00:00 | 2013.01.02 06:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 3 | 2013.01.01 00:00:00 | 2013.01.01 10:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 32 | 2013.01.01 10:00:00 | 2013.01.02 01:00:00 | 31 | 2013.01.04 08:00:00 |
| | | | | | | |
| 1. 128 | | | | | | |
| | 0 | 18 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 32 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 1 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 2013.01.01 17:00:00 | 2013.01.02 07:00:00 | 1 | |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 16 | |
| | 3 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 3 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 2 | 2013.01.02 06:00:00 |

Iteration:1 Best Value:113.0
Iteration:3 Best Value:116.0
Iteration:9 Best Value:119.0
Iteration:11 Best Value:120.0
Iteration:26 Best Value:126.0
Iteration:84 Best Value:128.0
Total Iteration Count:185

TotalProcessCompleteTime: 108.0 | AllOperationTardiness: 26.0 | AvgOperationsTardiness: 0.24074074074074073 | FlowTime: 31.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 16.0 | SetupCost: 864000.0 | OperationCost: 3312000.0 | TotalOperationCost: 4176000.0 | DiffJobEndDelivery: 0.0 | Earliness: 190.0 | ScheduledPercentage: 1.0

Karınca kolonisi 1.veri seti 2.test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 126 | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 1 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 17:00:00 | 2013.01.02 07:00:00 | 1 | |
| | 1 | 18 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 32 | 2013.01.01 09:00:00 | 2013.01.01 22:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 3 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 43 | 2013.01.01 17:00:00 | 2013.01.02 08:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | | | | | | |
| 2. 126 | | | | | | |
| | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 18 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 32 | 2013.01.01 17:00:00 | 2013.01.02 08:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 17:00:00 | 2013.01.02 06:00:00 | 16 | |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 2 | 2 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 1 | |
| | 3 | 3 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 43 | 2013.01.01 09:00:00 | 2013.01.01 22:00:00 | 42 | 2013.01.05 04:00:00 |
| 1. 129 | | | | | | |
| | | | | | | |
| | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 1 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 17:00:00 | 2013.01.02 05:00:00 | 1 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 16 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 16 | |
| | 2 | 18 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 43 | 2013.01.01 09:00:00 | 2013.01.01 22:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |

Iteration:1 Best Value:111.0
Iteration:2 Best Value:120.0
Iteration:45 Best Value:125.0
Iteration:145 Best Value:129.0
Total Iteration Count:246

TotalProcessCompleteTime: 108.0 | AllOperationTardiness: 27.0 | AvgOperationsTardiness: 0.25 | FlowTime: 30.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 16.0 | SetupCost: 864000.0 | OperationCost: 3312000.0 | TotalOperationCost: 4176000.0 | DiffJobEndDelivery: 0.0 | Earliness: 190.0 | ScheduledPercentage: 1.0

Karınca kolonisi 1.veri seti 3.test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 124 | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 18 | 2013.01.01 08:00:00 | 2013.01.01 19:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 32 | 2013.01.01 19:00:00 | 2013.01.02 08:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 1 | |
| | 2 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 17:00:00 | 2013.01.02 06:00:00 | 16 | |
| | 3 | 3 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 43 | 2013.01.01 09:00:00 | 2013.01.01 22:00:00 | 42 | 2013.01.05 04:00:00 |
| | | | | | | |
| 2. 126 | | | | | | |
| | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 1 | 18 | 2013.01.01 00:00:00 | 2013.01.01 10:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 43 | 2013.01.01 10:00:00 | 2013.01.01 23:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 32 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 3 | 2013.01.01 21:00:00 | 2013.01.02 06:00:00 | 2 | |
| | 3 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 1 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 16:00:00 | 2013.01.02 06:00:00 | 1 | |
| 1. 128 | | | | | | |
| | | | | | | |
| | 0 | 18 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 32 | 2013.01.01 09:00:00 | 2013.01.02 00:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 1 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 17:00:00 | 2013.01.02 07:00:00 | 1 | |
| | 2 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 3 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |

Iteration:1 Best Value:114.0
Iteration:4 Best Value:117.0
Iteration:7 Best Value:118.0
Iteration:18 Best Value:119.0
Iteration:65 Best Value:120.0
Iteration:86 Best Value:123.0
Iteration:145 Best Value:124.0
Iteration:163 Best Value:126.0
Iteration:178 Best Value:128.0
Total Iteration Count:279

TotalProcessCompleteTime: 107.0 | AllOperationTardiness: 26.0 | AvgOperationsTardiness: 0.24299065420560748 | FlowTime: 31.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 15.0 | SetupCost: 810000.0 | OperationCost: 3312000.0 | TotalOperationCost: 4122000.0 | DiffJobEndDelivery: 0.0 | Earliness: 190.0 | ScheduledPercentage: 1.0

Karınca kolonisi 2.veri seti 1.test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 233 | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 0 | 18 | 2013.01.01 22:00:00 | 2013.01.02 08:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 43 | 2013.01.02 08:00:00 | 2013.01.02 23:00:00 | 42 | 2013.01.05 04:00:00 |
| | 0 | 80 | 2013.01.02 23:00:00 | 2013.01.03 23:00:00 | 79 | |
| | 0 | 69 | 2013.01.03 23:00:00 | 2013.01.05 00:00:00 | 68 | 2013.01.07 06:00:00 |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 1 | 3 | 2013.01.01 22:00:00 | 2013.01.02 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 53 | 2013.01.02 09:00:00 | 2013.01.03 01:00:00 | 0 | |
| | 1 | 31 | 2013.01.03 01:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 1 | 95 | 2013.01.03 11:00:00 | 2013.01.06 02:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 79 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 2 | 94 | 2013.01.01 23:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 2 | 32 | 2013.01.03 11:00:00 | 2013.01.04 02:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 81 | 2013.01.04 02:00:00 | 2013.01.04 21:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 55 | 2013.01.04 21:00:00 | 2013.01.05 14:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 105 | 2013.01.01 15:00:00 | 2013.01.02 13:00:00 | 0 | |
| | 3 | 106 | 2013.01.02 13:00:00 | 2013.01.04 02:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 54 | 2013.01.04 02:00:00 | 2013.01.05 03:00:00 | 53 | |
| | | | | | | |
| | | | | | | |
| | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| 2. 244 | 0 | 3 | 2013.01.01 08:00:00 | 2013.01.01 19:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 17 | 2013.01.01 19:00:00 | 2013.01.02 08:00:00 | 16 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 68 | 2013.01.02 08:00:00 | 2013.01.03 01:00:00 | 0 | |
| | 0 | 18 | 2013.01.03 01:00:00 | 2013.01.03 12:00:00 | 17 | |
| | 0 | 55 | 2013.01.03 12:00:00 | 2013.01.04 07:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 69 | 2013.01.04 07:00:00 | 2013.01.05 09:00:00 | 68 | 2013.01.07 06:00:00 |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 31 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 1 | 105 | 2013.01.01 16:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 1 | 80 | 2013.01.02 14:00:00 | 2013.01.03 14:00:00 | 79 | |
| | 1 | 81 | 2013.01.03 14:00:00 | 2013.01.04 09:00:00 | 80 | 2013.01.08 12:00:00 |
| | 1 | 106 | 2013.01.04 09:00:00 | 2013.01.06 00:00:00 | 105 | 2013.01.10 14:00:00 |
| | 2 | 53 | 2013.01.01 00:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 2 | 79 | 2013.01.01 16:00:00 | 2013.01.02 08:00:00 | 0 | |
| | 2 | 32 | 2013.01.02 08:00:00 | 2013.01.02 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 43 | 2013.01.02 21:00:00 | 2013.01.03 10:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 95 | 2013.01.03 10:00:00 | 2013.01.05 23:00:00 | 94 | 2013.01.09 18:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 3 | 94 | 2013.01.01 22:00:00 | 2013.01.03 10:00:00 | 0 | |
| | 3 | 54 | 2013.01.03 10:00:00 | 2013.01.04 11:00:00 | 53 | |
| | | | | | | |
| | | | | | | |
| | 0 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 0 | 105 | 2013.01.01 15:00:00 | 2013.01.02 13:00:00 | 0 | |
| | 0 | 106 | 2013.01.02 13:00:00 | 2013.01.04 04:00:00 | 105 | 2013.01.10 14:00:00 |
| | 0 | 80 | 2013.01.04 04:00:00 | 2013.01.05 06:00:00 | 79 | |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 53 | 2013.01.01 08:00:00 | 2013.01.02 00:00:00 | 0 | |
| 1. 247 | 1 | 94 | 2013.01.02 00:00:00 | 2013.01.03 12:00:00 | 0 | |
| | 1 | 79 | 2013.01.03 12:00:00 | 2013.01.04 04:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 54 | 2013.01.04 04:00:00 | 2013.01.05 05:00:00 | 53 | |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 2 | 3 | 2013.01.01 22:00:00 | 2013.01.02 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 69 | 2013.01.02 09:00:00 | 2013.01.03 12:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 81 | 2013.01.03 12:00:00 | 2013.01.04 07:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 43 | 2013.01.04 07:00:00 | 2013.01.04 20:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 55 | 2013.01.04 20:00:00 | 2013.01.05 15:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 3 | 18 | 2013.01.01 22:00:00 | 2013.01.02 08:00:00 | 17 | 2013.01.03 12:00:00 |
| | 3 | 31 | 2013.01.02 08:00:00 | 2013.01.02 16:00:00 | 0 | |

Iteration:1 Best Value:220.0
Iteration:30 Best Value:247.0
Total Iteration Count:131

TotalProcessCompleteTime: 430.0 | AllOperationTardiness: 476.0 | AvgOperationsTardiness: 1.1069767441860465 | FlowTime: 121.0 | IdleTimeTotal: 5.0 | IdleCost: 18000.0 | TotalOutSource: 0.0 | SetupTime: 40.0 | SetupCost: 2160000.0 | OperationCost: 1.404E7 | TotalOperationCost: 1.62E7 | DiffJobEndDelivery: 3.0 | Earliness: 514.0 | ScheduledPercentage: 1.0

Karınca kolonisi 2.veri seti 2.test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 204 | 0 | 79 | 2013.01.01 00:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 16:00:00 | 2013.01.02 06:00:00 | 1 | |
| | 0 | 54 | 2013.01.02 06:00:00 | 2013.01.03 07:00:00 | 53 | |
| | 0 | 18 | 2013.01.03 07:00:00 | 2013.01.03 16:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 80 | 2013.01.03 16:00:00 | 2013.01.04 16:00:00 | 79 | |
| | 0 | 81 | 2013.01.04 16:00:00 | 2013.01.05 11:00:00 | 80 | 2013.01.08 12:00:00 |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 68 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 1 | 94 | 2013.01.01 23:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 1 | 95 | 2013.01.03 11:00:00 | 2013.01.06 00:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 53 | 2013.01.01 08:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 2 | 3 | 2013.01.02 00:00:00 | 2013.01.02 11:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 105 | 2013.01.02 11:00:00 | 2013.01.03 09:00:00 | 0 | |
| | 2 | 43 | 2013.01.03 09:00:00 | 2013.01.03 22:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 106 | 2013.01.03 22:00:00 | 2013.01.05 12:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 32 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 31 | 2013.01.04 08:00:00 |
| | 3 | 16 | 2013.01.01 23:00:00 | 2013.01.02 09:00:00 | 0 | |
| | 3 | 17 | 2013.01.02 09:00:00 | 2013.01.02 23:00:00 | 16 | |
| | 3 | 69 | 2013.01.02 23:00:00 | 2013.01.04 00:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 55 | 2013.01.04 00:00:00 | 2013.01.04 19:00:00 | 54 | 2013.01.06 10:00:00 |
| | | | | | | |
| 2. 224 | | | | | | |
| | 0 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 0 | 69 | 2013.01.01 15:00:00 | 2013.01.02 18:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 32 | 2013.01.02 18:00:00 | 2013.01.03 07:00:00 | 31 | 2013.01.04 08:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 43 | 2013.01.03 07:00:00 | 2013.01.03 22:00:00 | 42 | 2013.01.05 04:00:00 |
| | 0 | 80 | 2013.01.03 22:00:00 | 2013.01.04 23:00:00 | 79 | |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 79 | 2013.01.01 08:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 1 | 94 | 2013.01.02 00:00:00 | 2013.01.03 12:00:00 | 0 | |
| | 1 | 54 | 2013.01.03 12:00:00 | 2013.01.04 13:00:00 | 53 | |
| | 1 | 55 | 2013.01.04 13:00:00 | 2013.01.05 08:00:00 | 54 | 2013.01.06 10:00:00 |
| | 2 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 2 | 53 | 2013.01.01 17:00:00 | 2013.01.02 08:00:00 | 0 | |
| | 2 | 105 | 2013.01.02 08:00:00 | 2013.01.03 08:00:00 | 0 | |
| | 2 | 106 | 2013.01.03 08:00:00 | 2013.01.04 23:00:00 | 105 | 2013.01.10 14:00:00 |
| | 2 | 81 | 2013.01.04 23:00:00 | 2013.01.05 18:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 1 | |
| | 3 | 3 | 2013.01.01 23:00:00 | 2013.01.02 08:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 17 | 2013.01.02 08:00:00 | 2013.01.02 22:00:00 | 16 | |
| | 3 | 18 | 2013.01.02 22:00:00 | 2013.01.03 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 3 | 95 | 2013.01.03 12:00:00 | 2013.01.06 01:00:00 | 94 | 2013.01.09 18:00:00 |
| | | | | | | |
| | | | | | | |
| | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 0 | 18 | 2013.01.01 22:00:00 | 2013.01.02 08:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 105 | 2013.01.02 08:00:00 | 2013.01.03 07:00:00 | 0 | |
| | 0 | 81 | 2013.01.03 07:00:00 | 2013.01.04 02:00:00 | 80 | 2013.01.08 12:00:00 |
| | 0 | 106 | 2013.01.04 02:00:00 | 2013.01.05 17:00:00 | 105 | 2013.01.10 14:00:00 |
| 1. 242 | 1 | 53 | 2013.01.01 00:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 16:00:00 | 2013.01.02 06:00:00 | 1 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 3 | 2013.01.02 06:00:00 | 2013.01.02 15:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 80 | 2013.01.02 15:00:00 | 2013.01.03 16:00:00 | 79 | |
| | 1 | 55 | 2013.01.03 16:00:00 | 2013.01.04 11:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 69 | 2013.01.04 11:00:00 | 2013.01.05 12:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 2 | 94 | 2013.01.01 15:00:00 | 2013.01.03 03:00:00 | 0 | |
| | 2 | 43 | 2013.01.03 03:00:00 | 2013.01.03 16:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 54 | 2013.01.03 16:00:00 | 2013.01.04 17:00:00 | 53 | |
| | 3 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 1 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 3 | 31 | 2013.01.01 16:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 3 | 32 | 2013.01.02 00:00:00 | 2013.01.02 13:00:00 | 31 | 2013.01.04 08:00:00 |

Iteration:1 Best Value:195.0
Iteration:31 Best Value:224.0
Iteration:94 Best Value:242.0
Total Iteration Count:195

TotalProcessCompleteTime: 424.0 | AllOperationTardiness: 445.0 | AvgOperationsTardiness: 1.0495283018867925 | FlowTime: 114.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 34.0 | SetupCost: 1836000.0 | OperationCost: 1.404E7 | TotalOperationCost: 1.5876E7 | DiffJobEndDelivery: 9.0 | Earliness: 515.0 | ScheduledPercentage: 1.0

Karınca kolonisi 2.veri seti 3.test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 239 | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 0 | 105 | 2013.01.01 22:00:00 | 2013.01.02 20:00:00 | 0 | |
| | 0 | 18 | 2013.01.02 20:00:00 | 2013.01.03 06:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 32 | 2013.01.03 06:00:00 | 2013.01.03 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 81 | 2013.01.03 21:00:00 | 2013.01.04 16:00:00 | 80 | 2013.01.08 12:00:00 |
| | 0 | 55 | 2013.01.04 16:00:00 | 2013.01.05 11:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 1 | |
| | 1 | 3 | 2013.01.01 21:00:00 | 2013.01.02 08:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 79 | 2013.01.02 08:00:00 | 2013.01.03 01:00:00 | 0 | |
| | 1 | 95 | 2013.01.03 01:00:00 | 2013.01.05 14:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 94 | 2013.01.01 08:00:00 | 2013.01.02 20:00:00 | 0 | |
| | 2 | 53 | 2013.01.02 20:00:00 | 2013.01.03 12:00:00 | 0 | |
| | 2 | 80 | 2013.01.03 12:00:00 | 2013.01.04 14:00:00 | 79 | |
| | 2 | 106 | 2013.01.04 14:00:00 | 2013.01.06 05:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 43 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 68 | 2013.01.01 23:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 3 | 69 | 2013.01.02 14:00:00 | 2013.01.03 17:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 54 | 2013.01.03 17:00:00 | 2013.01.04 18:00:00 | 53 | |
| | | | | | | |
| | | | | | | |
| | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| 2. 249 | 0 | 43 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 42 | 2013.01.05 04:00:00 |
| | 0 | 68 | 2013.01.01 21:00:00 | 2013.01.02 13:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 17 | 2013.01.02 13:00:00 | 2013.01.03 02:00:00 | 16 | |
| | 0 | 54 | 2013.01.03 02:00:00 | 2013.01.04 04:00:00 | 53 | |
| | 0 | 95 | 2013.01.04 04:00:00 | 2013.01.06 19:00:00 | 94 | 2013.01.09 18:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 1 | |
| | 1 | 3 | 2013.01.01 23:00:00 | 2013.01.02 10:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 94 | 2013.01.02 10:00:00 | 2013.01.03 22:00:00 | 0 | |
| | 1 | 69 | 2013.01.03 22:00:00 | 2013.01.04 23:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 53 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 2 | 105 | 2013.01.01 23:00:00 | 2013.01.02 21:00:00 | 0 | |
| | 2 | 80 | 2013.01.02 21:00:00 | 2013.01.03 22:00:00 | 79 | |
| | 2 | 106 | 2013.01.03 22:00:00 | 2013.01.05 13:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 3 | 32 | 2013.01.01 09:00:00 | 2013.01.01 22:00:00 | 31 | 2013.01.04 08:00:00 |
| | 3 | 79 | 2013.01.01 22:00:00 | 2013.01.02 17:00:00 | 0 | |
| | 3 | 18 | 2013.01.02 17:00:00 | 2013.01.03 02:00:00 | 17 | 2013.01.03 12:00:00 |
| | 3 | 55 | 2013.01.03 02:00:00 | 2013.01.03 21:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 81 | 2013.01.03 21:00:00 | 2013.01.04 14:00:00 | 80 | 2013.01.08 12:00:00 |
| | | | | | | |
| | | | | | | |
| | 0 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 0 | 94 | 2013.01.01 15:00:00 | 2013.01.03 03:00:00 | 0 | |
| | 0 | 95 | 2013.01.03 03:00:00 | 2013.01.05 16:00:00 | 94 | 2013.01.09 18:00:00 |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 1 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 17:00:00 | 2013.01.02 07:00:00 | 1 | |
| 1. 273 | 1 | 68 | 2013.01.02 07:00:00 | 2013.01.02 22:00:00 | 0 | |
| | 1 | 69 | 2013.01.02 22:00:00 | 2013.01.04 01:00:00 | 68 | 2013.01.07 06:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 106 | 2013.01.04 01:00:00 | 2013.01.05 15:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 80 | 2013.01.02 15:00:00 | 2013.01.03 16:00:00 | 79 | |
| | 1 | 55 | 2013.01.03 16:00:00 | 2013.01.04 11:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 69 | 2013.01.04 11:00:00 | 2013.01.05 12:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 2 | 94 | 2013.01.01 15:00:00 | 2013.01.03 03:00:00 | 0 | |
| | 2 | 43 | 2013.01.03 03:00:00 | 2013.01.03 16:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 54 | 2013.01.03 16:00:00 | 2013.01.04 17:00:00 | 53 | |
| | 3 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 1 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 3 | 31 | 2013.01.01 16:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 3 | 32 | 2013.01.02 00:00:00 | 2013.01.02 13:00:00 | 31 | 2013.01.04 08:00:00 |

Iteration:1 Best Value:207.0
Iteration:9 Best Value:214.0
Iteration:17 Best Value:218.0
Iteration:25 Best Value:221.0
Iteration:29 Best Value:231.0
Iteration:100 Best Value:273.0
Total Iteration Count:201

TotalProcessCompleteTime: 426.0 | AllOperationTardiness: 462.0 | AvgOperationsTardiness: 1.0845070422535212 | FlowTime: 112.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 36.0 | SetupCost: 1944000.0 | OperationCost: 1.404E7 | TotalOperationCost: 1.5984E7 | DiffJobEndDelivery: 11.0 | Earliness: 552.0 | ScheduledPercentage: 1.0

Karınca kolonisi 3.veri seti 1.test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 178 | 0 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 0 | 79 | 2013.01.01 15:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 0 | 142 | 2013.01.02 07:00:00 | 2013.01.03 05:00:00 | 0 | |
| | 0 | 116 | 2013.01.03 05:00:00 | 2013.01.04 03:00:00 | 0 | |
| | 0 | 80 | 2013.01.04 03:00:00 | 2013.01.05 05:00:00 | 79 | |
| | 0 | 143 | 2013.01.05 05:00:00 | 2013.01.06 19:00:00 | 142 | |
| | 0 | 118 | 2013.01.06 19:00:00 | 2013.01.07 22:00:00 | 117 | 2013.01.11 10:00:00 |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 131 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 1 | 105 | 2013.01.02 06:00:00 | 2013.01.03 04:00:00 | 0 | |
| | 1 | 132 | 2013.01.03 04:00:00 | 2013.01.04 18:00:00 | 131 | 2013.01.12 06:00:00 |
| | 1 | 54 | 2013.01.04 18:00:00 | 2013.01.05 18:00:00 | 53 | |
| | 1 | 95 | 2013.01.05 18:00:00 | 2013.01.08 08:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 16 | |
| | 2 | 69 | 2013.01.01 21:00:00 | 2013.01.03 00:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 18 | 2013.01.03 00:00:00 | 2013.01.03 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 32 | 2013.01.03 09:00:00 | 2013.01.03 23:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 42 | 2013.01.03 23:00:00 | 2013.01.04 08:00:00 | 0 | |
| | 2 | 43 | 2013.01.04 08:00:00 | 2013.01.04 23:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 117 | 2013.01.04 23:00:00 | 2013.01.06 11:00:00 | 116 | |
| | 2 | 144 | 2013.01.06 11:00:00 | 2013.01.07 14:00:00 | 143 | 2013.01.13 02:00:00 |
| | 3 | 53 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 31 | 2013.01.01 15:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 23:00:00 | 2013.01.02 12:00:00 | 1 | |
| | 3 | 3 | 2013.01.02 12:00:00 | 2013.01.02 23:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 94 | 2013.01.02 23:00:00 | 2013.01.04 12:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 106 | 2013.01.04 12:00:00 | 2013.01.06 03:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 55 | 2013.01.06 03:00:00 | 2013.01.06 22:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 81 | 2013.01.06 22:00:00 | 2013.01.07 17:00:00 | 80 | 2013.01.08 12:00:00 |
| | | | | | | |
| | | | | | | |
| 2. 184 | 0 | 53 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 15:00:00 | 2013.01.02 05:00:00 | 1 | |
| | 0 | 131 | 2013.01.02 05:00:00 | 2013.01.03 03:00:00 | 0 | |
| | 0 | 94 | 2013.01.03 03:00:00 | 2013.01.04 15:00:00 | 0 | |
| | 0 | 54 | 2013.01.04 15:00:00 | 2013.01.05 16:00:00 | 53 | |
| | 0 | 143 | 2013.01.05 16:00:00 | 2013.01.07 04:00:00 | 142 | |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 79 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 1 | 16 | 2013.01.01 23:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 1 | 17 | 2013.01.02 07:00:00 | 2013.01.02 21:00:00 | 16 | |
| | 1 | 18 | 2013.01.02 21:00:00 | 2013.01.03 07:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 80 | 2013.01.03 07:00:00 | 2013.01.04 08:00:00 | 79 | |
| | 1 | 32 | 2013.01.04 08:00:00 | 2013.01.04 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 55 | 2013.01.04 21:00:00 | 2013.01.05 14:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 132 | 2013.01.05 14:00:00 | 2013.01.07 03:00:00 | 131 | 2013.01.12 06:00:00 |
| | 2 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 116 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 2 | 3 | 2013.01.02 06:00:00 | 2013.01.02 17:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 43 | 2013.01.02 17:00:00 | 2013.01.03 08:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 142 | 2013.01.03 08:00:00 | 2013.01.04 08:00:00 | 0 | |
| | 2 | 68 | 2013.01.04 08:00:00 | 2013.01.04 23:00:00 | 0 | |
| | 2 | 69 | 2013.01.04 23:00:00 | 2013.01.06 01:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 81 | 2013.01.06 01:00:00 | 2013.01.06 20:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 144 | 2013.01.06 20:00:00 | 2013.01.07 23:00:00 | 143 | 2013.01.13 02:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 105 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 3 | 106 | 2013.01.02 06:00:00 | 2013.01.03 19:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 117 | 2013.01.03 19:00:00 | 2013.01.05 08:00:00 | 116 | |
| | 3 | 118 | 2013.01.05 08:00:00 | 2013.01.06 10:00:00 | 117 | 2013.01.11 10:00:00 |
| | 3 | 95 | 2013.01.06 10:00:00 | 2013.01.08 23:00:00 | 94 | 2013.01.09 18:00:00 |
| | | | | | | |
| | | | | | | |
| 1. 188 | 0 | 116 | 2013.01.01 00:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 0 | 68 | 2013.01.01 23:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 0 | 2 | 2013.01.02 15:00:00 | 2013.01.03 05:00:00 | 1 | |
| | 0 | 43 | 2013.01.03 05:00:00 | 2013.01.03 20:00:00 | 42 | 2013.01.05 04:00:00 |
| | 0 | 69 | 2013.01.03 20:00:00 | 2013.01.04 21:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 144 | 2013.01.04 21:00:00 | 2013.01.06 00:00:00 | 143 | 2013.01.13 02:00:00 |
| | 0 | 95 | 2013.01.06 00:00:00 | 2013.01.08 15:00:00 | 94 | 2013.01.09 18:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 42 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 1 | 32 | 2013.01.01 16:00:00 | 2013.01.02 05:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 1 | 2013.01.02 05:00:00 | 2013.01.02 15:00:00 | 0 | |
| | 1 | 3 | 2013.01.02 15:00:00 | 2013.01.03 00:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 94 | 2013.01.03 00:00:00 | 2013.01.04 13:00:00 | 0 | |
| | 1 | 106 | 2013.01.04 13:00:00 | 2013.01.06 03:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 54 | 2013.01.06 03:00:00 | 2013.01.07 04:00:00 | 53 | |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 142 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 2 | 79 | 2013.01.02 06:00:00 | 2013.01.02 22:00:00 | 0 | |
| | 2 | 105 | 2013.01.02 22:00:00 | 2013.01.03 20:00:00 | 0 | |
| | 2 | 117 | 2013.01.03 20:00:00 | 2013.01.05 09:00:00 | 116 | |
| | 2 | 118 | 2013.01.05 09:00:00 | 2013.01.06 12:00:00 | 117 | 2013.01.11 10:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 2 | 81 | 2013.01.06 12:00:00 | 2013.01.07 05:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 53 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 17 | 2013.01.01 15:00:00 | 2013.01.02 05:00:00 | 16 | |
| | 3 | 18 | 2013.01.02 05:00:00 | 2013.01.02 14:00:00 | 17 | 2013.01.03 12:00:00 |
| | 3 | 143 | 2013.01.02 14:00:00 | 2013.01.04 04:00:00 | 142 | |
| | 3 | 80 | 2013.01.04 04:00:00 | 2013.01.05 05:00:00 | 79 | |
| | 3 | 131 | 2013.01.05 05:00:00 | 2013.01.06 03:00:00 | 0 | |
| | 3 | 55 | 2013.01.06 03:00:00 | 2013.01.06 22:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 132 | 2013.01.06 22:00:00 | 2013.01.08 11:00:00 | 131 | 2013.01.12 06:00:00 |

Iteration:1 Best Value:46.0
Iteration:15 Best Value:97.0
Iteration:34 Best Value:138.0
Iteration:53 Best Value:175.0
Iteration:79 Best Value:188.0
Total Iteration Count:180

TotalProcessCompleteTime: 658.0 | AllOperationTardiness: 1158.0 | AvgOperationsTardiness: 1.7598784194528876 | FlowTime: 183.0 | IdleTimeTotal: 1.0 | IdleCost: 3600.0 | TotalOutSource: 0.0 | SetupTime: 49.0 | SetupCost: 2646000.0 | OperationCost: 2.1924E7 | TotalOperationCost: 2.457E7 | DiffJobEndDelivery: 30.0 | Earliness: 706.0 | ScheduledPercentage: 1.0

Karınca kolonisi 3.veri seti 2.test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 123 | 0 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 0 | 142 | 2013.01.01 15:00:00 | 2013.01.02 13:00:00 | 0 | |
| | 0 | 131 | 2013.01.02 13:00:00 | 2013.01.03 12:00:00 | 0 | |
| | 0 | 116 | 2013.01.03 12:00:00 | 2013.01.04 12:00:00 | 0 | |
| | 0 | 55 | 2013.01.04 12:00:00 | 2013.01.05 06:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 117 | 2013.01.05 06:00:00 | 2013.01.06 19:00:00 | 116 | |
| | 0 | 80 | 2013.01.06 19:00:00 | 2013.01.07 19:00:00 | 79 | |
| | 1 | 105 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 1 | 31 | 2013.01.01 22:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 1 | 32 | 2013.01.02 06:00:00 | 2013.01.02 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 106 | 2013.01.02 21:00:00 | 2013.01.04 10:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 18 | 2013.01.04 10:00:00 | 2013.01.04 21:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 143 | 2013.01.04 21:00:00 | 2013.01.06 11:00:00 | 142 | |
| | 1 | 144 | 2013.01.06 11:00:00 | 2013.01.07 14:00:00 | 143 | 2013.01.13 02:00:00 |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 53 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 2 | 54 | 2013.01.01 23:00:00 | 2013.01.03 01:00:00 | 53 | |
| | 2 | 94 | 2013.01.03 01:00:00 | 2013.01.04 13:00:00 | 0 | |
| | 2 | 95 | 2013.01.04 13:00:00 | 2013.01.07 02:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 81 | 2013.01.07 02:00:00 | 2013.01.07 19:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 1 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 16:00:00 | 2013.01.02 06:00:00 | 1 | |
| | 3 | 3 | 2013.01.02 06:00:00 | 2013.01.02 16:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 79 | 2013.01.02 16:00:00 | 2013.01.03 08:00:00 | 0 | |
| | 3 | 17 | 2013.01.03 08:00:00 | 2013.01.03 22:00:00 | 16 | |
| | 3 | 43 | 2013.01.03 22:00:00 | 2013.01.04 12:00:00 | 42 | 2013.01.05 04:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 69 | 2013.01.04 12:00:00 | 2013.01.05 13:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 132 | 2013.01.05 13:00:00 | 2013.01.07 03:00:00 | 131 | 2013.01.12 06:00:00 |
| | 3 | 118 | 2013.01.07 03:00:00 | 2013.01.08 04:00:00 | 117 | 2013.01.11 10:00:00 |
| | | | | | | |
| | | | | | | |
| 2. 145 | 0 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 0 | 79 | 2013.01.01 15:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 0 | 17 | 2013.01.02 06:00:00 | 2013.01.02 20:00:00 | 16 | |
| | 0 | 18 | 2013.01.02 20:00:00 | 2013.01.03 07:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 69 | 2013.01.03 07:00:00 | 2013.01.04 10:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 80 | 2013.01.04 10:00:00 | 2013.01.05 11:00:00 | 79 | |
| | 0 | 81 | 2013.01.05 11:00:00 | 2013.01.06 04:00:00 | 80 | 2013.01.08 12:00:00 |
| | 0 | 54 | 2013.01.06 04:00:00 | 2013.01.07 05:00:00 | 53 | |
| | 0 | 55 | 2013.01.07 05:00:00 | 2013.01.08 00:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 32 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 42 | 2013.01.01 23:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 1 | 131 | 2013.01.02 07:00:00 | 2013.01.03 05:00:00 | 0 | |
| | 1 | 94 | 2013.01.03 05:00:00 | 2013.01.04 17:00:00 | 0 | |
| | 1 | 43 | 2013.01.04 17:00:00 | 2013.01.05 07:00:00 | 42 | 2013.01.05 04:00:00 |
| | 1 | 95 | 2013.01.05 07:00:00 | 2013.01.07 20:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 2 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 1 | |
| | 2 | 3 | 2013.01.01 23:00:00 | 2013.01.02 10:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 142 | 2013.01.02 10:00:00 | 2013.01.03 10:00:00 | 0 | |
| | 2 | 117 | 2013.01.03 10:00:00 | 2013.01.05 00:00:00 | 116 | |
| | 2 | 118 | 2013.01.05 00:00:00 | 2013.01.06 03:00:00 | 117 | 2013.01.11 10:00:00 |
| | 2 | 143 | 2013.01.06 03:00:00 | 2013.01.07 16:00:00 | 142 | |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 105 | 2013.01.01 09:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 3 | 116 | 2013.01.02 07:00:00 | 2013.01.03 06:00:00 | 0 | |
| | 3 | 53 | 2013.01.03 06:00:00 | 2013.01.03 22:00:00 | 0 | |
| | 3 | 106 | 2013.01.03 22:00:00 | 2013.01.05 12:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 132 | 2013.01.05 12:00:00 | 2013.01.07 03:00:00 | 131 | 2013.01.12 06:00:00 |
| | 3 | 144 | 2013.01.07 03:00:00 | 2013.01.08 06:00:00 | 143 | 2013.01.13 02:00:00 |
| | | | | | | |
| | | | | | | |
| 1. 220 | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 1 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 16:00:00 | 2013.01.02 05:00:00 | 1 | |
| | 0 | 3 | 2013.01.02 05:00:00 | 2013.01.02 16:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 54 | 2013.01.02 16:00:00 | 2013.01.03 18:00:00 | 53 | |
| | 0 | 32 | 2013.01.03 18:00:00 | 2013.01.04 07:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 94 | 2013.01.04 07:00:00 | 2013.01.05 21:00:00 | 0 | |
| | 0 | 117 | 2013.01.05 21:00:00 | 2013.01.07 09:00:00 | 116 | |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 53 | 2013.01.01 08:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 1 | 43 | 2013.01.02 00:00:00 | 2013.01.02 13:00:00 | 42 | 2013.01.05 04:00:00 |
| | 1 | 105 | 2013.01.02 13:00:00 | 2013.01.03 12:00:00 | 0 | |
| | 1 | 55 | 2013.01.03 12:00:00 | 2013.01.04 05:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 132 | 2013.01.04 05:00:00 | 2013.01.05 19:00:00 | 131 | 2013.01.12 06:00:00 |
| | 1 | 144 | 2013.01.05 19:00:00 | 2013.01.06 22:00:00 | 143 | 2013.01.13 02:00:00 |
| | 1 | 95 | 2013.01.06 22:00:00 | 2013.01.09 11:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 116 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 22:00:00 | 2013.01.02 11:00:00 | 16 | |
| | 2 | 18 | 2013.01.02 11:00:00 | 2013.01.02 22:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 79 | 2013.01.02 22:00:00 | 2013.01.03 13:00:00 | 0 | |
| | 2 | 143 | 2013.01.03 20:00:00 | 2013.01.05 09:00:00 | 142 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 2 | 106 | 2013.01.05 09:00:00 | 2013.01.07 00:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 131 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 3 | 68 | 2013.01.02 06:00:00 | 2013.01.02 21:00:00 | 0 | |
| | 3 | 142 | 2013.01.02 21:00:00 | 2013.01.03 20:00:00 | 0 | |
| | 3 | 80 | 2013.01.03 20:00:00 | 2013.01.04 21:00:00 | 79 | |
| | 3 | 81 | 2013.01.04 21:00:00 | 2013.01.05 16:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 69 | 2013.01.05 16:00:00 | 2013.01.06 19:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 118 | 2013.01.06 19:00:00 | 2013.01.07 22:00:00 | 117 | 2013.01.11 10:00:00 |

Iteration:1 Best Value:36.0
Iteration:4 Best Value:100.0
Iteration:46 Best Value:145.0
Iteration:62 Best Value:220.0
Total Iteration Count:163

TotalProcessCompleteTime: 659.0 | AllOperationTardiness: 1122.0 | AvgOperationsTardiness: 1.7025796661608499 | FlowTime: 203.0 | IdleTimeTotal: 7.0 | IdleCost: 25200.0 | TotalOutSource: 0.0 | SetupTime: 50.0 | SetupCost: 2700000.0 | OperationCost: 2.1924E7 | TotalOperationCost: 2.4624E7 | DiffJobEndDelivery: 10.0 | Earliness: 690.0 | ScheduledPercentage: 1.0

Karınca kolonisi 3.veri seti 3.test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 149 | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 105 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 0 | 68 | 2013.01.02 06:00:00 | 2013.01.02 21:00:00 | 0 | |
| | 0 | 17 | 2013.01.02 21:00:00 | 2013.01.03 09:00:00 | 16 | |
| | 0 | 18 | 2013.01.03 09:00:00 | 2013.01.03 20:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 69 | 2013.01.03 20:00:00 | 2013.01.04 21:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 81 | 2013.01.04 21:00:00 | 2013.01.05 16:00:00 | 80 | 2013.01.08 12:00:00 |
| | 0 | 55 | 2013.01.05 16:00:00 | 2013.01.06 09:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 95 | 2013.01.06 09:00:00 | 2013.01.08 23:00:00 | 94 | 2013.01.09 18:00:00 |
| | 1 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 1 | 53 | 2013.01.01 15:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 1 | 2 | 2013.01.02 06:00:00 | 2013.01.02 19:00:00 | 1 | |
| | 1 | 3 | 2013.01.02 19:00:00 | 2013.01.03 05:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 131 | 2013.01.03 05:00:00 | 2013.01.04 03:00:00 | 0 | |
| | 1 | 132 | 2013.01.04 03:00:00 | 2013.01.05 18:00:00 | 131 | 2013.01.12 06:00:00 |
| | 1 | 143 | 2013.01.05 18:00:00 | 2013.01.07 07:00:00 | 142 | |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 142 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 2 | 32 | 2013.01.02 06:00:00 | 2013.01.02 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 54 | 2013.01.02 21:00:00 | 2013.01.03 23:00:00 | 53 | |
| | 2 | 117 | 2013.01.03 23:00:00 | 2013.01.05 12:00:00 | 116 | |
| | 2 | 118 | 2013.01.05 12:00:00 | 2013.01.06 13:00:00 | 117 | 2013.01.11 10:00:00 |
| | 2 | 144 | 2013.01.06 13:00:00 | 2013.01.07 16:00:00 | 143 | 2013.01.13 02:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 116 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 3 | 42 | 2013.01.02 06:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 3 | 80 | 2013.01.02 14:00:00 | 2013.01.03 15:00:00 | 79 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 43 | 2013.01.03 15:00:00 | 2013.01.04 05:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 94 | 2013.01.04 05:00:00 | 2013.01.05 17:00:00 | 0 | |
| | 3 | 106 | 2013.01.05 17:00:00 | 2013.01.07 06:00:00 | 105 | 2013.01.10 14:00:00 |
| | | | | | | |
| | | | | | | |
| 2. 171 | 0 | 32 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 53 | 2013.01.01 23:00:00 | 2013.01.02 16:00:00 | 0 | |
| | 0 | 68 | 2013.01.02 16:00:00 | 2013.01.03 07:00:00 | 0 | |
| | 0 | 79 | 2013.01.03 07:00:00 | 2013.01.03 23:00:00 | 0 | |
| | 0 | 69 | 2013.01.03 23:00:00 | 2013.01.05 00:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 81 | 2013.01.05 00:00:00 | 2013.01.05 19:00:00 | 80 | 2013.01.08 12:00:00 |
| | 0 | 95 | 2013.01.05 19:00:00 | 2013.01.08 10:00:00 | 94 | 2013.01.09 18:00:00 |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 1 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 17:00:00 | 2013.01.02 07:00:00 | 1 | |
| | 1 | 3 | 2013.01.02 07:00:00 | 2013.01.02 18:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 94 | 2013.01.02 18:00:00 | 2013.01.04 06:00:00 | 0 | |
| | 1 | 142 | 2013.01.04 06:00:00 | 2013.01.05 05:00:00 | 0 | |
| | 1 | 143 | 2013.01.05 05:00:00 | 2013.01.06 18:00:00 | 142 | |
| | 1 | 144 | 2013.01.06 18:00:00 | 2013.01.07 20:00:00 | 143 | 2013.01.13 02:00:00 |
| | 2 | 105 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 2 | 16 | 2013.01.01 22:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 2 | 17 | 2013.01.02 07:00:00 | 2013.01.02 21:00:00 | 16 | |
| | 2 | 18 | 2013.01.02 21:00:00 | 2013.01.03 08:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 54 | 2013.01.03 08:00:00 | 2013.01.04 09:00:00 | 53 | |
| | 2 | 43 | 2013.01.04 09:00:00 | 2013.01.04 22:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 80 | 2013.01.04 22:00:00 | 2013.01.05 23:00:00 | 79 | |
| | 2 | 55 | 2013.01.05 23:00:00 | 2013.01.06 16:00:00 | 54 | 2013.01.06 10:00:00 |
| | 2 | 106 | 2013.01.06 16:00:00 | 2013.01.08 07:00:00 | 105 | 2013.01.10 14:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 116 | 2013.01.01 08:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 3 | 131 | 2013.01.02 07:00:00 | 2013.01.03 05:00:00 | 0 | |
| | 3 | 117 | 2013.01.03 05:00:00 | 2013.01.04 19:00:00 | 116 | |
| | 3 | 132 | 2013.01.04 19:00:00 | 2013.01.06 10:00:00 | 131 | 2013.01.12 06:00:00 |
| | 3 | 118 | 2013.01.06 10:00:00 | 2013.01.07 12:00:00 | 117 | 2013.01.11 10:00:00 |
| | | | | | | |
| | | | | | | |
| 1. 183 | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 32 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 53 | 2013.01.01 23:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 0 | 142 | 2013.01.02 14:00:00 | 2013.01.03 12:00:00 | 0 | |
| | 0 | 105 | 2013.01.03 12:00:00 | 2013.01.04 10:00:00 | 0 | |
| | 0 | 54 | 2013.01.04 10:00:00 | 2013.01.05 11:00:00 | 53 | |
| | 0 | 55 | 2013.01.05 11:00:00 | 2013.01.06 04:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 106 | 2013.01.06 04:00:00 | 2013.01.07 18:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 1 | 79 | 2013.01.01 09:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 1 | 68 | 2013.01.02 00:00:00 | 2013.01.02 15:00:00 | 0 | |
| | 1 | 80 | 2013.01.02 15:00:00 | 2013.01.03 17:00:00 | 79 | |
| | 1 | 143 | 2013.01.03 17:00:00 | 2013.01.05 06:00:00 | 142 | |
| | 1 | 144 | 2013.01.05 06:00:00 | 2013.01.06 09:00:00 | 143 | 2013.01.13 02:00:00 |
| | 1 | 132 | 2013.01.06 09:00:00 | 2013.01.07 22:00:00 | 131 | 2013.01.12 06:00:00 |
| | 2 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 43 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 94 | 2013.01.01 23:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 2 | 131 | 2013.01.03 11:00:00 | 2013.01.04 09:00:00 | 0 | |
| | 2 | 117 | 2013.01.04 09:00:00 | 2013.01.05 23:00:00 | 116 | |
| | 2 | 95 | 2013.01.05 23:00:00 | 2013.01.08 14:00:00 | 94 | 2013.01.09 18:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 131 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 3 | 68 | 2013.01.02 06:00:00 | 2013.01.02 21:00:00 | 0 | |
| | 3 | 142 | 2013.01.02 21:00:00 | 2013.01.03 20:00:00 | 0 | |
| | 3 | 80 | 2013.01.03 20:00:00 | 2013.01.04 21:00:00 | 79 | |
| | 3 | 81 | 2013.01.04 21:00:00 | 2013.01.05 16:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 69 | 2013.01.05 16:00:00 | 2013.01.06 19:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 118 | 2013.01.06 19:00:00 | 2013.01.07 22:00:00 | 117 | 2013.01.11 10:00:00 |

Iteration:1 Best Value:-133.0
Iteration:2 Best Value:-111.0
Iteration:5 Best Value:-59.0
Iteration:8 Best Value:-46.0
Iteration:10 Best Value:-36.0
Iteration:15 Best Value:71.0
Iteration:27 Best Value:77.0
Iteration:30 Best Value:80.0
Iteration:39 Best Value:90.0
Iteration:105 Best Value:110.0
Iteration:169 Best Value:183.0
Total Iteration Count:270

TotalProcessCompleteTime: 661.0 | AllOperationTardiness: 1117.0 | AvgOperationsTardiness: 1.6898638426626325 | FlowTime: 182.0 | IdleTimeTotal: 1.0 | IdleCost: 3600.0 | TotalOutSource: 0.0 | SetupTime: 52.0 | SetupCost: 2808000.0 | OperationCost: 2.1924E7 | TotalOperationCost: 2.4732E7 | DiffJobEndDelivery: 29.0 | Earliness: 694.0 | ScheduledPercentage: 1.0

Melez Algoritma 1. veri seti 1. test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 128 | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 1 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 17:00:00 | 2013.01.02 06:00:00 | 1 | |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 32 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 3 | 2013.01.01 21:00:00 | 2013.01.02 06:00:00 | 2 | |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 3 | 18 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 3 | 43 | 2013.01.01 09:00:00 | 2013.01.01 22:00:00 | 42 | 2013.01.05 04:00:00 |
| | | | | | | |
| 2. 129 | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 16 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 16:00:00 | 2013.01.02 06:00:00 | 16 | |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 43 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 3 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 18 | 2013.01.01 09:00:00 | 2013.01.01 18:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 32 | 2013.01.01 18:00:00 | 2013.01.02 07:00:00 | 31 | 2013.01.04 08:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | | | | | | |
| 1. 130 | | | | | | |
| | 0 | 3 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 32 | 2013.01.01 09:00:00 | 2013.01.02 00:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 16 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 17 | 2013.01.01 16:00:00 | 2013.01.02 06:00:00 | 16 | |
| | 2 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 18 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 17 | 2013.01.03 12:00:00 |

Iteration:1 Best Value:113.0
Iteration:5 Best Value:124.0
Iteration:7 Best Value:130.0
Total Iteration Count:108

TotalProcessCompleteTime: 106.0 | AllOperationTardiness: 26.0 | AvgOperationsTardiness: 0.24528301886792453 | FlowTime: 30.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 14.0 | SetupCost: 756000.0 | OperationCost: 3312000.0 | TotalOperationCost: 4068000.0 | DiffJobEndDelivery: 0.0 | Earliness: 190.0 | ScheduledPercentage: 1.0

Melez Algoritma 1. veri seti 2. test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 128 | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 1 | |
| | 1 | 18 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 3 | 2013.01.01 09:00:00 | 2013.01.01 18:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 32 | 2013.01.01 18:00:00 | 2013.01.02 07:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 43 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 3 | 17 | 2013.01.01 17:00:00 | 2013.01.02 07:00:00 | 16 | |
| | | | | | | |
| 2. 129 | | | | | | |
| | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 32 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 2 | 18 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 3 | 2013.01.01 09:00:00 | 2013.01.01 18:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 43 | 2013.01.01 18:00:00 | 2013.01.02 07:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 1 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 17:00:00 | 2013.01.02 07:00:00 | 1 | |
| 1. 129 | | | | | | |
| | | | | | | |
| | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 17:00:00 | 2013.01.02 07:00:00 | 16 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 1 | |
| | 2 | 3 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 18 | 2013.01.01 09:00:00 | 2013.01.01 18:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 32 | 2013.01.01 18:00:00 | 2013.01.02 07:00:00 | 31 | 2013.01.04 08:00:00 |

Iteration:1 Best Value:123.0
Iteration:7 Best Value:124.0
Iteration:69 Best Value:129.0
Total Iteration Count:170

TotalProcessCompleteTime: 106.0 | AllOperationTardiness: 27.0 | AvgOperationsTardiness: 0.25471698113207547 | FlowTime: 31.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 14.0 | SetupCost: 756000.0 | OperationCost: 3312000.0 | TotalOperationCost: 4068000.0 | DiffJobEndDelivery: 0.0 | Earliness: 191.0 | ScheduledPercentage: 1.0

Melez Algoritma 1. veri seti 3. test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 124 | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 1 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 17:00:00 | 2013.01.02 05:00:00 | 1 | |
| | 1 | 16 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 16 | |
| | 2 | 18 | 2013.01.01 00:00:00 | 2013.01.01 11:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 32 | 2013.01.01 11:00:00 | 2013.01.02 00:00:00 | 31 | 2013.01.04 08:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 43 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 3 | 2013.01.01 21:00:00 | 2013.01.02 06:00:00 | 2 | |
| | | | | | | |
| 2. 124 | | | | | | |
| | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 3 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 32 | 2013.01.01 17:00:00 | 2013.01.02 08:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 16 | |
| | 2 | 18 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 43 | 2013.01.01 09:00:00 | 2013.01.02 00:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 1 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 16:00:00 | 2013.01.02 05:00:00 | 1 | |
| 1. 128 | | | | | | |
| | | | | | | |
| | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 32 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 18 | 2013.01.01 21:00:00 | 2013.01.02 06:00:00 | 17 | 2013.01.03 12:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 16 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 1 | 17 | 2013.01.01 16:00:00 | 2013.01.02 06:00:00 | 16 | |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 2 | 2 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 1 | |

Iteration:1 Best Value:120.0
Iteration:2 Best Value:121.0
Iteration:53 Best Value:123.0
Iteration:101 Best Value:124.0
Iteration:188 Best Value:128.0
Total Iteration Count:289

TotalProcessCompleteTime: 105.0 | AllOperationTardiness: 30.0 | AvgOperationsTardiness: 0.2857142857142857 | FlowTime: 30.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 13.0 | SetupCost: 702000.0 | OperationCost: 3312000.0 | TotalOperationCost: 4014000.0 | DiffJobEndDelivery: 0.0 | Earliness: 188.0 | ScheduledPercentage: 1.0

Melez Algoritma 2. veri seti 1. test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 281 | 0 | 94 | 2013.01.01 00:00:00 | 2013.01.02 12:00:00 | 0 | |
| | 0 | 17 | 2013.01.02 12:00:00 | 2013.01.03 02:00:00 | 16 | |
| | 0 | 18 | 2013.01.03 02:00:00 | 2013.01.03 12:00:00 | 17 | |
| | 0 | 106 | 2013.01.03 12:00:00 | 2013.01.05 02:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 1 | 16 | 2013.01.01 15:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 1 | 53 | 2013.01.01 23:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 1 | 105 | 2013.01.02 14:00:00 | 2013.01.03 12:00:00 | 0 | |
| | 1 | 55 | 2013.01.03 12:00:00 | 2013.01.04 07:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 81 | 2013.01.04 07:00:00 | 2013.01.05 02:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 42 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 2 | 3 | 2013.01.01 16:00:00 | 2013.01.02 02:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 43 | 2013.01.02 02:00:00 | 2013.01.02 15:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 32 | 2013.01.02 15:00:00 | 2013.01.03 04:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 54 | 2013.01.03 04:00:00 | 2013.01.04 04:00:00 | 53 | |
| | 2 | 80 | 2013.01.04 04:00:00 | 2013.01.05 04:00:00 | 79 | |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 3 | 68 | 2013.01.01 22:00:00 | 2013.01.02 13:00:00 | 0 | |
| | 3 | 69 | 2013.01.02 13:00:00 | 2013.01.03 14:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 95 | 2013.01.03 14:00:00 | 2013.01.06 05:00:00 | 94 | 2013.01.09 18:00:00 |
| | | | | | | |
| 2. 289 | | | | | | |
| | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 94 | 2013.01.01 08:00:00 | 2013.01.02 20:00:00 | 0 | |
| | 0 | 18 | 2013.01.02 20:00:00 | 2013.01.03 05:00:00 | 17 | 2013.01.03 12:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 80 | 2013.01.03 05:00:00 | 2013.01.04 08:00:00 | 79 | |
| | 0 | 55 | 2013.01.04 08:00:00 | 2013.01.05 03:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 1 | 3 | 2013.01.01 22:00:00 | 2013.01.02 07:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 32 | 2013.01.02 07:00:00 | 2013.01.02 22:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 53 | 2013.01.02 22:00:00 | 2013.01.03 13:00:00 | 0 | |
| | 1 | 81 | 2013.01.03 13:00:00 | 2013.01.04 08:00:00 | 80 | 2013.01.08 12:00:00 |
| | 1 | 54 | 2013.01.04 08:00:00 | 2013.01.05 08:00:00 | 53 | |
| | 2 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 105 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 2 | 79 | 2013.01.02 06:00:00 | 2013.01.02 22:00:00 | 0 | |
| | 2 | 106 | 2013.01.02 22:00:00 | 2013.01.04 11:00:00 | 105 | 2013.01.10 14:00:00 |
| | 2 | 69 | 2013.01.04 11:00:00 | 2013.01.05 14:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 16 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 3 | 68 | 2013.01.01 09:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 3 | 43 | 2013.01.02 00:00:00 | 2013.01.02 15:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 17 | 2013.01.02 15:00:00 | 2013.01.03 03:00:00 | 16 | |
| | 3 | 95 | 2013.01.03 03:00:00 | 2013.01.05 16:00:00 | 94 | 2013.01.09 18:00:00 |
| | | | | | | |
| | | | | | | |
| | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 68 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 0 | 32 | 2013.01.01 23:00:00 | 2013.01.02 12:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 69 | 2013.01.02 12:00:00 | 2013.01.03 14:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 80 | 2013.01.03 14:00:00 | 2013.01.04 14:00:00 | 79 | |
| | 0 | 55 | 2013.01.04 14:00:00 | 2013.01.05 07:00:00 | 54 | 2013.01.06 10:00:00 |
| 1. 291 | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 94 | 2013.01.01 08:00:00 | 2013.01.02 20:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 43 | 2013.01.02 20:00:00 | 2013.01.03 11:00:00 | 42 | 2013.01.05 04:00:00 |
| | 1 | 54 | 2013.01.03 11:00:00 | 2013.01.04 12:00:00 | 53 | |
| | 1 | 81 | 2013.01.04 12:00:00 | 2013.01.05 07:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 2 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 1 | |
| | 2 | 3 | 2013.01.01 21:00:00 | 2013.01.02 06:00:00 | 2 | |
| | 2 | 17 | 2013.01.02 06:00:00 | 2013.01.02 18:00:00 | 16 | |
| | 2 | 18 | 2013.01.02 18:00:00 | 2013.01.03 03:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 95 | 2013.01.03 03:00:00 | 2013.01.05 16:00:00 | 94 | 2013.01.09 18:00:00 |
| | 3 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 16 | 2013.01.01 15:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 3 | 53 | 2013.01.01 23:00:00 | 2013.01.02 14:00:00 | 0 | |

Iteration:1 Best Value:159.0
Iteration:2 Best Value:258.0
Iteration:8 Best Value:265.0
Iteration:30 Best Value:289.0
Iteration:107 Best Value:291.0
Total Iteration Count:208

TotalProcessCompleteTime: 417.0 | AllOperationTardiness: 471.0 | AvgOperationsTardiness: 1.129496402877698 | FlowTime: 112.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 27.0 | SetupCost: 1458000.0 | OperationCost: 1.404E7 | TotalOperationCost: 1.5498E7 | DiffJobEndDelivery: 0.0 | Earliness: 515.0 | ScheduledPercentage: 1.0

Melez Algoritma 2. veri seti 2. test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 265 | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 0 | 17 | 2013.01.01 22:00:00 | 2013.01.02 11:00:00 | 16 | |
| | 0 | 18 | 2013.01.02 11:00:00 | 2013.01.02 20:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 32 | 2013.01.02 20:00:00 | 2013.01.03 09:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 106 | 2013.01.03 09:00:00 | 2013.01.05 00:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 53 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 1 | 42 | 2013.01.01 15:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 1 | 3 | 2013.01.01 23:00:00 | 2013.01.02 08:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 43 | 2013.01.02 08:00:00 | 2013.01.02 23:00:00 | 42 | 2013.01.05 04:00:00 |
| | 1 | 54 | 2013.01.02 23:00:00 | 2013.01.04 00:00:00 | 53 | |
| | 1 | 80 | 2013.01.04 00:00:00 | 2013.01.05 00:00:00 | 79 | |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 31 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 2 | 105 | 2013.01.01 16:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 2 | 68 | 2013.01.02 14:00:00 | 2013.01.03 05:00:00 | 0 | |
| | 2 | 69 | 2013.01.03 05:00:00 | 2013.01.04 06:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 81 | 2013.01.04 06:00:00 | 2013.01.04 23:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 55 | 2013.01.04 23:00:00 | 2013.01.05 16:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 94 | 2013.01.01 15:00:00 | 2013.01.03 03:00:00 | 0 | |
| | 3 | 95 | 2013.01.03 03:00:00 | 2013.01.05 16:00:00 | 94 | 2013.01.09 18:00:00 |
| | | | | | | |
| | | | | | | |
| | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| 2. 274 | 0 | 3 | 2013.01.01 09:00:00 | 2013.01.01 20:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 32 | 2013.01.01 20:00:00 | 2013.01.02 11:00:00 | 31 | 2013.01.04 08:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 17 | 2013.01.02 11:00:00 | 2013.01.03 00:00:00 | 16 | |
| | 0 | 18 | 2013.01.03 00:00:00 | 2013.01.03 11:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 55 | 2013.01.03 11:00:00 | 2013.01.04 06:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 81 | 2013.01.04 06:00:00 | 2013.01.05 01:00:00 | 80 | 2013.01.08 12:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 1 | 53 | 2013.01.01 22:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 1 | 54 | 2013.01.02 14:00:00 | 2013.01.03 16:00:00 | 53 | |
| | 1 | 95 | 2013.01.03 16:00:00 | 2013.01.06 07:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 79 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 2 | 94 | 2013.01.01 23:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 2 | 80 | 2013.01.03 11:00:00 | 2013.01.04 12:00:00 | 79 | |
| | 3 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 68 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 3 | 43 | 2013.01.01 23:00:00 | 2013.01.02 14:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 69 | 2013.01.02 14:00:00 | 2013.01.03 16:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 105 | 2013.01.03 16:00:00 | 2013.01.04 15:00:00 | 0 | |
| | 3 | 106 | 2013.01.04 15:00:00 | 2013.01.06 04:00:00 | 105 | 2013.01.10 14:00:00 |
| | | | | | | |
| | | | | | | |
| | 0 | 105 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 22:00:00 | 2013.01.02 12:00:00 | 16 | |
| | 0 | 18 | 2013.01.02 12:00:00 | 2013.01.02 21:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 53 | 2013.01.02 21:00:00 | 2013.01.03 12:00:00 | 0 | |
| | 0 | 54 | 2013.01.03 12:00:00 | 2013.01.04 14:00:00 | 53 | |
| | 0 | 81 | 2013.01.04 14:00:00 | 2013.01.05 09:00:00 | 80 | 2013.01.08 12:00:00 |
| 1. 278 | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 79 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 94 | 2013.01.01 23:00:00 | 2013.01.03 11:00:00 | 0 | |
| | 1 | 95 | 2013.01.03 11:00:00 | 2013.01.06 01:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 2 | 2 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 1 | |
| | 2 | 3 | 2013.01.01 23:00:00 | 2013.01.02 10:00:00 | 2 | 2013.01.02 06:00:00 |
| | 2 | 42 | 2013.01.02 10:00:00 | 2013.01.02 18:00:00 | 0 | |
| | 2 | 68 | 2013.01.02 18:00:00 | 2013.01.03 09:00:00 | 0 | |
| | 2 | 43 | 2013.01.03 09:00:00 | 2013.01.03 23:00:00 | 42 | 2013.01.05 04:00:00 |
| | 2 | 55 | 2013.01.03 23:00:00 | 2013.01.04 18:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 32 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 31 | 2013.01.04 08:00:00 |
| | 3 | 106 | 2013.01.01 23:00:00 | 2013.01.03 12:00:00 | 105 | 2013.01.10 14:00:00 |

Iteration:1 Best Value:252.0
Iteration:20 Best Value:255.0
Iteration:38 Best Value:256.0
Iteration:94 Best Value:262.0
Iteration:100 Best Value:278.0
Total Iteration Count:201

TotalProcessCompleteTime: 426.0 | AllOperationTardiness: 463.0 | AvgOperationsTardiness: 1.0868544600938967 | FlowTime: 121.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 36.0 | SetupCost: 1944000.0 | OperationCost: 1.404E7 | TotalOperationCost: 1.5984E7 | DiffJobEndDelivery: 4.0 | Earliness: 540.0 | ScheduledPercentage: 1.0

Melez Algoritma 2. veri seti 3. test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 242 | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 0 | 3 | 2013.01.01 22:00:00 | 2013.01.02 08:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 18 | 2013.01.02 08:00:00 | 2013.01.02 17:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 32 | 2013.01.02 17:00:00 | 2013.01.03 06:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 95 | 2013.01.03 12:00:00 | 2013.01.06 01:00:00 | 94 | 2013.01.09 18:00:00 |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 68 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 1 | 43 | 2013.01.01 23:00:00 | 2013.01.02 12:00:00 | 42 | 2013.01.05 04:00:00 |
| | 1 | 79 | 2013.01.02 12:00:00 | 2013.01.03 03:00:00 | 0 | |
| | 1 | 54 | 2013.01.03 03:00:00 | 2013.01.04 03:00:00 | 53 | |
| | 1 | 55 | 2013.01.04 03:00:00 | 2013.01.04 20:00:00 | 54 | 2013.01.06 10:00:00 |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 2 | 105 | 2013.01.01 22:00:00 | 2013.01.02 20:00:00 | 0 | |
| | 2 | 106 | 2013.01.02 20:00:00 | 2013.01.04 09:00:00 | 105 | 2013.01.10 14:00:00 |
| | 2 | 69 | 2013.01.04 09:00:00 | 2013.01.05 10:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 53 | 2013.01.01 08:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 3 | 94 | 2013.01.02 00:00:00 | 2013.01.03 12:00:00 | 0 | |
| | 3 | 80 | 2013.01.03 12:00:00 | 2013.01.04 14:00:00 | 79 | |
| | 3 | 81 | 2013.01.04 14:00:00 | 2013.01.05 07:00:00 | 80 | 2013.01.08 12:00:00 |
| | | | | | | |
| 2. 247 | | | | | | |
| | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 09:00:00 | 2013.01.01 23:00:00 | 1 | |
| | 0 | 3 | 2013.01.01 23:00:00 | 2013.01.02 09:00:00 | 2 | 2013.01.02 06:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 17 | 2013.01.02 09:00:00 | 2013.01.03 00:00:00 | 16 | |
| | 0 | 105 | 2013.01.03 00:00:00 | 2013.01.03 22:00:00 | 0 | |
| | 0 | 69 | 2013.01.03 22:00:00 | 2013.01.04 23:00:00 | 68 | 2013.01.07 06:00:00 |
| | 1 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 1 | 16 | 2013.01.01 15:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 1 | 31 | 2013.01.02 00:00:00 | 2013.01.02 08:00:00 | 0 | |
| | 1 | 79 | 2013.01.02 08:00:00 | 2013.01.03 00:00:00 | 0 | |
| | 1 | 18 | 2013.01.03 00:00:00 | 2013.01.03 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 1 | 95 | 2013.01.03 09:00:00 | 2013.01.06 00:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 94 | 2013.01.01 00:00:00 | 2013.01.02 12:00:00 | 0 | |
| | 2 | 55 | 2013.01.02 12:00:00 | 2013.01.03 07:00:00 | 54 | 2013.01.06 10:00:00 |
| | 2 | 32 | 2013.01.03 07:00:00 | 2013.01.03 22:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 106 | 2013.01.03 22:00:00 | 2013.01.05 12:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 09:00:00 | 0 | |
| | 3 | 43 | 2013.01.01 09:00:00 | 2013.01.01 22:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 53 | 2013.01.01 22:00:00 | 2013.01.02 15:00:00 | 0 | |
| | 3 | 54 | 2013.01.02 15:00:00 | 2013.01.03 17:00:00 | 53 | |
| | 3 | 80 | 2013.01.03 17:00:00 | 2013.01.04 18:00:00 | 79 | |
| | 3 | 81 | 2013.01.04 18:00:00 | 2013.01.05 13:00:00 | 80 | 2013.01.08 12:00:00 |
| | | | | | | |
| | | | | | | |
| | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 43 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 42 | 2013.01.05 04:00:00 |
| | 0 | 69 | 2013.01.01 23:00:00 | 2013.01.03 00:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 95 | 2013.01.03 03:00:00 | 2013.01.05 16:00:00 | 94 | 2013.01.09 18:00:00 |
| | 1 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| 1. 274 | 1 | 3 | 2013.01.01 22:00:00 | 2013.01.02 07:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 53 | 2013.01.02 07:00:00 | 2013.01.02 23:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 17 | 2013.01.02 23:00:00 | 2013.01.03 13:00:00 | 16 | |
| | 1 | 81 | 2013.01.03 13:00:00 | 2013.01.04 08:00:00 | 80 | 2013.01.08 12:00:00 |
| | 1 | 55 | 2013.01.04 08:00:00 | 2013.01.05 03:00:00 | 54 | 2013.01.06 10:00:00 |
| | 2 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 79 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 2 | 16 | 2013.01.01 23:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 2 | 105 | 2013.01.02 07:00:00 | 2013.01.03 05:00:00 | 0 | |
| | 2 | 80 | 2013.01.03 05:00:00 | 2013.01.04 07:00:00 | 79 | |
| | 2 | 106 | 2013.01.04 07:00:00 | 2013.01.05 20:00:00 | 105 | 2013.01.10 14:00:00 |
| | 3 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 94 | 2013.01.01 15:00:00 | 2013.01.03 03:00:00 | 0 | |
| | 3 | 18 | 2013.01.03 03:00:00 | 2013.01.03 14:00:00 | 17 | 2013.01.03 12:00:00 |

Iteration:1 Best Value:274.0
Total Iteration Count:102

TotalProcessCompleteTime: 423.0 | AllOperationTardiness: 484.0 | AvgOperationsTardiness: 1.1442080378250592 | FlowTime: 116.0 | IdleTimeTotal: 3.0 | IdleCost: 10800.0 | TotalOutSource: 0.0 | SetupTime: 33.0 | SetupCost: 1782000.0 | OperationCost: 1.404E7 | TotalOperationCost: 1.5822E7 | DiffJobEndDelivery: 3.0 | Earliness: 527.0 | ScheduledPercentage: 1.0

Melez Algoritma 3. veri seti 1. test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 306 | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 116 | 2013.01.01 08:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 0 | 16 | 2013.01.02 07:00:00 | 2013.01.02 15:00:00 | 0 | |
| | 0 | 68 | 2013.01.02 15:00:00 | 2013.01.03 06:00:00 | 0 | |
| | 0 | 18 | 2013.01.03 06:00:00 | 2013.01.03 17:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 69 | 2013.01.03 17:00:00 | 2013.01.04 18:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 118 | 2013.01.04 18:00:00 | 2013.01.05 21:00:00 | 117 | 2013.01.11 10:00:00 |
| | 0 | 131 | 2013.01.05 21:00:00 | 2013.01.06 20:00:00 | 0 | |
| | 0 | 132 | 2013.01.06 20:00:00 | 2013.01.08 10:00:00 | 131 | 2013.01.12 06:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 32 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 2 | 2013.01.02 00:00:00 | 2013.01.02 14:00:00 | 1 | |
| | 1 | 54 | 2013.01.02 14:00:00 | 2013.01.03 15:00:00 | 53 | |
| | 1 | 106 | 2013.01.03 15:00:00 | 2013.01.05 06:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 55 | 2013.01.05 06:00:00 | 2013.01.06 01:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 80 | 2013.01.06 01:00:00 | 2013.01.07 02:00:00 | 79 | |
| | 2 | 94 | 2013.01.01 00:00:00 | 2013.01.02 12:00:00 | 0 | |
| | 2 | 105 | 2013.01.02 12:00:00 | 2013.01.03 10:00:00 | 0 | |
| | 2 | 79 | 2013.01.03 10:00:00 | 2013.01.04 02:00:00 | 0 | |
| | 2 | 95 | 2013.01.04 02:00:00 | 2013.01.06 15:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 143 | 2013.01.06 15:00:00 | 2013.01.08 04:00:00 | 142 | |
| | 3 | 53 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 1 | 2013.01.01 15:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 3 | 43 | 2013.01.02 00:00:00 | 2013.01.02 13:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 3 | 2013.01.02 13:00:00 | 2013.01.02 23:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 17 | 2013.01.02 23:00:00 | 2013.01.03 12:00:00 | 16 | |
| | 3 | 142 | 2013.01.03 12:00:00 | 2013.01.04 14:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 117 | 2013.01.04 14:00:00 | 2013.01.06 04:00:00 | 116 | |
| | 3 | 81 | 2013.01.06 04:00:00 | 2013.01.06 23:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 144 | 2013.01.06 23:00:00 | 2013.01.08 02:00:00 | 143 | 2013.01.13 02:00:00 |
| | | | | | | |
| | | | | | | |
| 2. 311 | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 17:00:00 | 2013.01.02 06:00:00 | 1 | |
| | 0 | 43 | 2013.01.02 06:00:00 | 2013.01.02 21:00:00 | 42 | 2013.01.05 04:00:00 |
| | 0 | 17 | 2013.01.02 21:00:00 | 2013.01.03 11:00:00 | 16 | |
| | 0 | 18 | 2013.01.03 11:00:00 | 2013.01.03 22:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 105 | 2013.01.03 22:00:00 | 2013.01.04 21:00:00 | 0 | |
| | 0 | 106 | 2013.01.04 21:00:00 | 2013.01.06 10:00:00 | 105 | 2013.01.10 14:00:00 |
| | 0 | 143 | 2013.01.06 10:00:00 | 2013.01.07 23:00:00 | 142 | |
| | 1 | 53 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 1 | 68 | 2013.01.01 15:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 1 | 3 | 2013.01.02 06:00:00 | 2013.01.02 17:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 54 | 2013.01.02 17:00:00 | 2013.01.03 20:00:00 | 53 | |
| | 1 | 55 | 2013.01.03 20:00:00 | 2013.01.04 13:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 69 | 2013.01.04 13:00:00 | 2013.01.05 16:00:00 | 68 | 2013.01.07 06:00:00 |
| | 1 | 95 | 2013.01.05 16:00:00 | 2013.01.08 05:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 116 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 2 | 131 | 2013.01.02 06:00:00 | 2013.01.03 04:00:00 | 0 | |
| | 2 | 94 | 2013.01.03 04:00:00 | 2013.01.04 16:00:00 | 0 | |
| | 2 | 132 | 2013.01.04 16:00:00 | 2013.01.06 05:00:00 | 131 | 2013.01.12 06:00:00 |
| | 2 | 118 | 2013.01.06 05:00:00 | 2013.01.07 06:00:00 | 117 | 2013.01.11 10:00:00 |
| | 3 | 79 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 31 | 2013.01.01 15:00:00 | 2013.01.01 23:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 80 | 2013.01.01 23:00:00 | 2013.01.03 00:00:00 | 79 | |
| | 3 | 32 | 2013.01.03 00:00:00 | 2013.01.03 14:00:00 | 31 | 2013.01.04 08:00:00 |
| | 3 | 81 | 2013.01.03 14:00:00 | 2013.01.04 08:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 142 | 2013.01.04 08:00:00 | 2013.01.05 07:00:00 | 0 | |
| | 3 | 117 | 2013.01.05 07:00:00 | 2013.01.06 21:00:00 | 116 | |
| | 3 | 144 | 2013.01.06 21:00:00 | 2013.01.07 22:00:00 | 143 | 2013.01.13 02:00:00 |
| | | | | | | |
| | | | | | | |
| 1. 338 | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 53 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 0 | 31 | 2013.01.01 23:00:00 | 2013.01.02 09:00:00 | 0 | |
| | 0 | 18 | 2013.01.02 09:00:00 | 2013.01.02 18:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 105 | 2013.01.02 18:00:00 | 2013.01.03 17:00:00 | 0 | |
| | 0 | 94 | 2013.01.03 17:00:00 | 2013.01.05 05:00:00 | 0 | |
| | 0 | 95 | 2013.01.05 05:00:00 | 2013.01.07 20:00:00 | 94 | 2013.01.09 18:00:00 |
| | 1 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 1 | 3 | 2013.01.01 22:00:00 | 2013.01.02 08:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 68 | 2013.01.02 08:00:00 | 2013.01.02 23:00:00 | 0 | |
| | 1 | 117 | 2013.01.02 23:00:00 | 2013.01.04 12:00:00 | 116 | |
| | 1 | 118 | 2013.01.04 12:00:00 | 2013.01.05 13:00:00 | 117 | 2013.01.11 10:00:00 |
| | 1 | 144 | 2013.01.05 13:00:00 | 2013.01.06 14:00:00 | 143 | 2013.01.13 02:00:00 |
| | 1 | 80 | 2013.01.06 14:00:00 | 2013.01.07 14:00:00 | 79 | |
| | 2 | 116 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 22:00:00 | 2013.01.02 12:00:00 | 16 | |
| | 2 | 54 | 2013.01.02 12:00:00 | 2013.01.03 14:00:00 | 53 | |
| | 2 | 55 | 2013.01.03 14:00:00 | 2013.01.04 07:00:00 | 54 | 2013.01.06 10:00:00 |
| | 2 | 69 | 2013.01.04 07:00:00 | 2013.01.05 10:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 143 | 2013.01.05 10:00:00 | 2013.01.06 23:00:00 | 142 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 2 | 81 | 2013.01.06 23:00:00 | 2013.01.07 18:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 142 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 3 | 79 | 2013.01.01 22:00:00 | 2013.01.02 13:00:00 | 0 | |
| | 3 | 42 | 2013.01.02 13:00:00 | 2013.01.02 21:00:00 | 0 | |
| | 3 | 32 | 2013.01.02 21:00:00 | 2013.01.03 12:00:00 | 31 | 2013.01.04 08:00:00 |
| | 3 | 131 | 2013.01.03 12:00:00 | 2013.01.04 11:00:00 | 0 | |
| | 3 | 43 | 2013.01.04 11:00:00 | 2013.01.05 00:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 132 | 2013.01.05 00:00:00 | 2013.01.06 14:00:00 | 131 | 2013.01.12 06:00:00 |
| | 3 | 106 | 2013.01.06 14:00:00 | 2013.01.08 04:00:00 | 105 | 2013.01.10 14:00:00 |

Iteration:1 Best Value:33.0
Iteration:2 Best Value:147.0
Iteration:3 Best Value:180.0
Iteration:37 Best Value:213.0
Iteration:60 Best Value:293.0
Iteration:127 Best Value:311.0
Iteration:153 Best Value:338.0
Total Iteration Count:254

TotalProcessCompleteTime: 656.0 | AllOperationTardiness: 1139.0 | AvgOperationsTardiness: 1.736280487804878 | FlowTime: 172.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 47.0 | SetupCost: 2538000.0 | OperationCost: 2.1924E7 | TotalOperationCost: 2.4462E7 | DiffJobEndDelivery: 2.0 | Earliness: 692.0 | ScheduledPercentage: 1.0

Melez Algoritma 3. veri seti 2. test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 313 | 0 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 0 | 3 | 2013.01.01 22:00:00 | 2013.01.02 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 94 | 2013.01.02 09:00:00 | 2013.01.03 21:00:00 | 0 | |
| | 0 | 55 | 2013.01.03 21:00:00 | 2013.01.04 14:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 117 | 2013.01.04 14:00:00 | 2013.01.06 03:00:00 | 116 | |
| | 0 | 95 | 2013.01.06 03:00:00 | 2013.01.08 18:00:00 | 94 | 2013.01.09 18:00:00 |
| | 1 | 53 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 1 | 68 | 2013.01.01 15:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 1 | 69 | 2013.01.02 06:00:00 | 2013.01.03 07:00:00 | 68 | 2013.01.07 06:00:00 |
| | 1 | 43 | 2013.01.03 07:00:00 | 2013.01.03 21:00:00 | 42 | 2013.01.05 04:00:00 |
| | 1 | 54 | 2013.01.03 21:00:00 | 2013.01.04 23:00:00 | 53 | |
| | 1 | 144 | 2013.01.04 23:00:00 | 2013.01.06 00:00:00 | 143 | 2013.01.13 02:00:00 |
| | 1 | 106 | 2013.01.06 00:00:00 | 2013.01.07 14:00:00 | 105 | 2013.01.10 14:00:00 |
| | 2 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 2 | 79 | 2013.01.01 17:00:00 | 2013.01.02 08:00:00 | 0 | |
| | 2 | 17 | 2013.01.02 08:00:00 | 2013.01.02 22:00:00 | 16 | |
| | 2 | 42 | 2013.01.02 22:00:00 | 2013.01.03 06:00:00 | 0 | |
| | 2 | 18 | 2013.01.03 06:00:00 | 2013.01.03 15:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 105 | 2013.01.03 15:00:00 | 2013.01.04 14:00:00 | 0 | |
| | 2 | 131 | 2013.01.04 14:00:00 | 2013.01.05 14:00:00 | 0 | |
| | 2 | 81 | 2013.01.05 14:00:00 | 2013.01.06 09:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 132 | 2013.01.06 09:00:00 | 2013.01.08 00:00:00 | 131 | 2013.01.12 06:00:00 |
| | 3 | 116 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 3 | 142 | 2013.01.01 22:00:00 | 2013.01.02 20:00:00 | 0 | |
| | 3 | 32 | 2013.01.02 20:00:00 | 2013.01.03 11:00:00 | 31 | 2013.01.04 08:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 80 | 2013.01.03 11:00:00 | 2013.01.04 11:00:00 | 79 | |
| | 3 | 143 | 2013.01.04 11:00:00 | 2013.01.06 00:00:00 | 142 | |
| | 3 | 118 | 2013.01.06 00:00:00 | 2013.01.07 01:00:00 | 117 | 2013.01.11 10:00:00 |
| | | | | | | |
| | | | | | | |
| 2. 325 | 0 | 131 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 0 | 68 | 2013.01.01 22:00:00 | 2013.01.02 13:00:00 | 0 | |
| | 0 | 18 | 2013.01.02 13:00:00 | 2013.01.02 23:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 32 | 2013.01.02 23:00:00 | 2013.01.03 13:00:00 | 31 | 2013.01.04 08:00:00 |
| | 0 | 54 | 2013.01.03 13:00:00 | 2013.01.04 16:00:00 | 53 | |
| | 0 | 55 | 2013.01.04 16:00:00 | 2013.01.05 09:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 144 | 2013.01.05 09:00:00 | 2013.01.06 12:00:00 | 143 | 2013.01.13 02:00:00 |
| | 0 | 118 | 2013.01.06 12:00:00 | 2013.01.07 15:00:00 | 117 | 2013.01.11 10:00:00 |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 53 | 2013.01.01 08:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 1 | 142 | 2013.01.02 00:00:00 | 2013.01.02 22:00:00 | 0 | |
| | 1 | 94 | 2013.01.02 22:00:00 | 2013.01.04 10:00:00 | 0 | |
| | 1 | 80 | 2013.01.04 10:00:00 | 2013.01.05 11:00:00 | 79 | |
| | 1 | 95 | 2013.01.05 11:00:00 | 2013.01.08 00:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 79 | 2013.01.01 00:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 2 | 16 | 2013.01.01 16:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 2 | 42 | 2013.01.02 00:00:00 | 2013.01.02 08:00:00 | 0 | |
| | 2 | 132 | 2013.01.02 08:00:00 | 2013.01.03 23:00:00 | 131 | 2013.01.12 06:00:00 |
| | 2 | 105 | 2013.01.03 23:00:00 | 2013.01.04 21:00:00 | 0 | |
| | 2 | 143 | 2013.01.04 21:00:00 | 2013.01.06 10:00:00 | 142 | |
| | 2 | 69 | 2013.01.06 10:00:00 | 2013.01.07 12:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 3 | 3 | 2013.01.01 22:00:00 | 2013.01.02 09:00:00 | 2 | 2013.01.02 06:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 17 | 2013.01.02 09:00:00 | 2013.01.02 22:00:00 | 16 | |
| | 3 | 43 | 2013.01.02 22:00:00 | 2013.01.03 12:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 116 | 2013.01.03 12:00:00 | 2013.01.04 10:00:00 | 0 | |
| | 3 | 81 | 2013.01.04 10:00:00 | 2013.01.05 03:00:00 | 80 | 2013.01.08 12:00:00 |
| | 3 | 117 | 2013.01.05 03:00:00 | 2013.01.06 17:00:00 | 116 | |
| | 3 | 106 | 2013.01.06 17:00:00 | 2013.01.08 08:00:00 | 105 | 2013.01.10 14:00:00 |
| | | | | | | |
| | | | | | | |
| 1. 368 | 0 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 1 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 0 | 79 | 2013.01.01 16:00:00 | 2013.01.02 07:00:00 | 0 | |
| | 0 | 43 | 2013.01.02 07:00:00 | 2013.01.02 20:00:00 | 42 | 2013.01.05 04:00:00 |
| | 0 | 53 | 2013.01.02 20:00:00 | 2013.01.03 14:00:00 | 0 | |
| | 0 | 54 | 2013.01.03 14:00:00 | 2013.01.04 16:00:00 | 53 | |
| | 0 | 80 | 2013.01.04 16:00:00 | 2013.01.05 17:00:00 | 79 | |
| | 0 | 55 | 2013.01.05 17:00:00 | 2013.01.06 12:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 117 | 2013.01.06 12:00:00 | 2013.01.08 00:00:00 | 116 | |
| | 1 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 32 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 2 | 2013.01.01 21:00:00 | 2013.01.02 09:00:00 | 1 | |
| | 1 | 3 | 2013.01.02 09:00:00 | 2013.01.02 18:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 105 | 2013.01.02 18:00:00 | 2013.01.03 17:00:00 | 0 | |
| | 1 | 106 | 2013.01.03 17:00:00 | 2013.01.05 06:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 132 | 2013.01.05 06:00:00 | 2013.01.06 19:00:00 | 131 | 2013.01.12 06:00:00 |
| | 1 | 144 | 2013.01.06 19:00:00 | 2013.01.07 20:00:00 | 143 | 2013.01.13 02:00:00 |
| | 2 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 16 | |
| | 2 | 68 | 2013.01.01 22:00:00 | 2013.01.02 13:00:00 | 0 | |
| | 2 | 94 | 2013.01.02 13:00:00 | 2013.01.04 01:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 2 | 116 | 2013.01.04 01:00:00 | 2013.01.05 00:00:00 | 0 | |
| | 2 | 81 | 2013.01.05 00:00:00 | 2013.01.05 17:00:00 | 80 | 2013.01.08 12:00:00 |
| | 2 | 143 | 2013.01.05 17:00:00 | 2013.01.07 08:00:00 | 142 | |
| | 3 | 131 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 3 | 142 | 2013.01.01 22:00:00 | 2013.01.02 22:00:00 | 0 | |
| | 3 | 18 | 2013.01.02 22:00:00 | 2013.01.03 09:00:00 | 17 | 2013.01.03 12:00:00 |
| | 3 | 69 | 2013.01.03 09:00:00 | 2013.01.04 12:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 95 | 2013.01.04 12:00:00 | 2013.01.07 01:00:00 | 94 | 2013.01.09 18:00:00 |
| | 3 | 118 | 2013.01.07 01:00:00 | 2013.01.08 03:00:00 | 117 | 2013.01.11 10:00:00 |

Iteration:1 Best Value:63.0
Iteration:10 Best Value:69.0
Iteration:12 Best Value:93.0
Iteration:14 Best Value:123.0
Iteration:15 Best Value:128.0
Iteration:16 Best Value:163.0
Iteration:40 Best Value:174.0
Iteration:70 Best Value:220.0
Iteration:106 Best Value:250.0
Iteration:130 Best Value:297.0
Iteration:132 Best Value:368.0
Total Iteration Count:233

TotalProcessCompleteTime: 655.0 | AllOperationTardiness: 1135.0 | AvgOperationsTardiness: 1.7328244274809161 | FlowTime: 171.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 46.0 | SetupCost: 2484000.0 | OperationCost: 2.1924E7 | TotalOperationCost: 2.4408E7 | DiffJobEndDelivery: 14.0 | Earliness: 780.0 | ScheduledPercentage: 1.0

Melez Algoritma 3. veri seti 3. test

| | Makine | Operasyon Id | Başlangıç | Bitiş | Önceki | Sipariş Teslimi |
|---|---|---|---|---|---|---|
| 3. 204 | 0 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 16 | 2013.01.01 08:00:00 | 2013.01.01 17:00:00 | 0 | |
| | 0 | 17 | 2013.01.01 17:00:00 | 2013.01.02 07:00:00 | 16 | |
| | 0 | 18 | 2013.01.02 07:00:00 | 2013.01.02 17:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 3 | 2013.01.02 17:00:00 | 2013.01.03 04:00:00 | 2 | 2013.01.02 06:00:00 |
| | 0 | 79 | 2013.01.03 04:00:00 | 2013.01.03 20:00:00 | 0 | |
| | 0 | 54 | 2013.01.03 20:00:00 | 2013.01.04 21:00:00 | 53 | |
| | 0 | 94 | 2013.01.04 21:00:00 | 2013.01.06 10:00:00 | 0 | |
| | 0 | 132 | 2013.01.06 10:00:00 | 2013.01.07 23:00:00 | 131 | 2013.01.12 06:00:00 |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 105 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 1 | 1 | 2013.01.02 06:00:00 | 2013.01.02 15:00:00 | 0 | |
| | 1 | 2 | 2013.01.02 15:00:00 | 2013.01.03 05:00:00 | 1 | |
| | 1 | 32 | 2013.01.03 05:00:00 | 2013.01.03 18:00:00 | 31 | 2013.01.04 08:00:00 |
| | 1 | 131 | 2013.01.03 18:00:00 | 2013.01.04 17:00:00 | 0 | |
| | 1 | 144 | 2013.01.04 17:00:00 | 2013.01.05 20:00:00 | 143 | 2013.01.13 02:00:00 |
| | 1 | 55 | 2013.01.05 20:00:00 | 2013.01.06 15:00:00 | 54 | 2013.01.06 10:00:00 |
| | 1 | 95 | 2013.01.06 15:00:00 | 2013.01.09 06:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 142 | 2013.01.01 00:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 2 | 69 | 2013.01.01 23:00:00 | 2013.01.03 00:00:00 | 68 | 2013.01.07 06:00:00 |
| | 2 | 143 | 2013.01.03 00:00:00 | 2013.01.04 14:00:00 | 142 | |
| | 2 | 106 | 2013.01.04 14:00:00 | 2013.01.06 05:00:00 | 105 | 2013.01.10 14:00:00 |
| | 2 | 80 | 2013.01.06 05:00:00 | 2013.01.07 08:00:00 | 79 | |
| | 3 | 68 | 2013.01.01 00:00:00 | 2013.01.01 15:00:00 | 0 | |
| | 3 | 53 | 2013.01.01 15:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 3 | 43 | 2013.01.02 06:00:00 | 2013.01.02 21:00:00 | 42 | 2013.01.05 04:00:00 |
| | 3 | 116 | 2013.01.02 21:00:00 | 2013.01.03 21:00:00 | 0 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 117 | 2013.01.03 21:00:00 | 2013.01.05 10:00:00 | 116 | |
| | 3 | 118 | 2013.01.05 10:00:00 | 2013.01.06 11:00:00 | 117 | 2013.01.11 10:00:00 |
| | 3 | 81 | 2013.01.06 11:00:00 | 2013.01.07 04:00:00 | 80 | 2013.01.08 12:00:00 |
| | | | | | | |
| | | | | | | |
| 2. 237 | 0 | 16 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 0 | 79 | 2013.01.01 08:00:00 | 2013.01.01 23:00:00 | 0 | |
| | 0 | 116 | 2013.01.01 23:00:00 | 2013.01.03 01:00:00 | 0 | |
| | 0 | 94 | 2013.01.03 01:00:00 | 2013.01.04 13:00:00 | 0 | |
| | 0 | 43 | 2013.01.04 13:00:00 | 2013.01.05 04:00:00 | 42 | |
| | 0 | 81 | 2013.01.05 04:00:00 | 2013.01.05 23:00:00 | 80 | 2013.01.08 12:00:00 |
| | 0 | 117 | 2013.01.05 23:00:00 | 2013.01.07 12:00:00 | 116 | |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 1 | 3 | 2013.01.01 22:00:00 | 2013.01.02 07:00:00 | 2 | 2013.01.02 06:00:00 |
| | 1 | 143 | 2013.01.02 07:00:00 | 2013.01.03 22:00:00 | 142 | |
| | 1 | 144 | 2013.01.03 22:00:00 | 2013.01.05 00:00:00 | 143 | 2013.01.13 02:00:00 |
| | 1 | 95 | 2013.01.05 00:00:00 | 2013.01.07 15:00:00 | 94 | 2013.01.09 18:00:00 |
| | 2 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 2 | 17 | 2013.01.01 08:00:00 | 2013.01.01 21:00:00 | 16 | |
| | 2 | 105 | 2013.01.01 21:00:00 | 2013.01.02 19:00:00 | 0 | |
| | 2 | 18 | 2013.01.02 19:00:00 | 2013.01.03 05:00:00 | 17 | 2013.01.03 12:00:00 |
| | 2 | 68 | 2013.01.03 05:00:00 | 2013.01.03 21:00:00 | 0 | |
| | 2 | 131 | 2013.01.03 21:00:00 | 2013.01.04 19:00:00 | 0 | |
| | 2 | 106 | 2013.01.04 19:00:00 | 2013.01.06 08:00:00 | 105 | 2013.01.10 14:00:00 |
| | 2 | 132 | 2013.01.06 08:00:00 | 2013.01.07 22:00:00 | 131 | 2013.01.12 06:00:00 |
| | 3 | 31 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 142 | 2013.01.01 08:00:00 | 2013.01.02 06:00:00 | 0 | |
| | 3 | 32 | 2013.01.02 06:00:00 | 2013.01.02 21:00:00 | 31 | 2013.01.04 08:00:00 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3 | 53 | 2013.01.02 21:00:00 | 2013.01.03 14:00:00 | 0 | |
| | 3 | 80 | 2013.01.03 14:00:00 | 2013.01.04 15:00:00 | 79 | |
| | 3 | 54 | 2013.01.04 15:00:00 | 2013.01.05 16:00:00 | 53 | |
| | 3 | 55 | 2013.01.05 16:00:00 | 2013.01.06 09:00:00 | 54 | 2013.01.06 10:00:00 |
| | 3 | 69 | 2013.01.06 09:00:00 | 2013.01.07 10:00:00 | 68 | 2013.01.07 06:00:00 |
| | 3 | 118 | 2013.01.07 10:00:00 | 2013.01.08 13:00:00 | 117 | 2013.01.11 10:00:00 |
| | | | | | | |
| | | | | | | |
| 1. 350 | 0 | 131 | 2013.01.01 00:00:00 | 2013.01.01 22:00:00 | 0 | |
| | 0 | 68 | 2013.01.01 22:00:00 | 2013.01.02 13:00:00 | 0 | |
| | 0 | 17 | 2013.01.02 13:00:00 | 2013.01.03 02:00:00 | 16 | |
| | 0 | 18 | 2013.01.03 02:00:00 | 2013.01.03 13:00:00 | 17 | 2013.01.03 12:00:00 |
| | 0 | 69 | 2013.01.03 13:00:00 | 2013.01.04 14:00:00 | 68 | 2013.01.07 06:00:00 |
| | 0 | 55 | 2013.01.04 14:00:00 | 2013.01.05 07:00:00 | 54 | 2013.01.06 10:00:00 |
| | 0 | 81 | 2013.01.05 07:00:00 | 2013.01.06 01:00:00 | 80 | 2013.01.08 12:00:00 |
| | 0 | 106 | 2013.01.06 01:00:00 | 2013.01.07 16:00:00 | 105 | 2013.01.10 14:00:00 |
| | 1 | 42 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 1 | 31 | 2013.01.01 08:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 1 | 16 | 2013.01.01 16:00:00 | 2013.01.02 00:00:00 | 0 | |
| | 1 | 94 | 2013.01.02 00:00:00 | 2013.01.03 12:00:00 | 0 | |
| | 1 | 43 | 2013.01.03 12:00:00 | 2013.01.04 03:00:00 | 42 | 2013.01.05 04:00:00 |
| | 1 | 142 | 2013.01.04 03:00:00 | 2013.01.05 02:00:00 | 0 | |
| | 1 | 117 | 2013.01.05 02:00:00 | 2013.01.06 15:00:00 | 116 | |
| | 1 | 143 | 2013.01.06 15:00:00 | 2013.01.08 04:00:00 | 142 | |
| | 2 | 53 | 2013.01.01 00:00:00 | 2013.01.01 16:00:00 | 0 | |
| | 2 | 116 | 2013.01.01 16:00:00 | 2013.01.02 14:00:00 | 0 | |
| | 2 | 32 | 2013.01.02 14:00:00 | 2013.01.03 05:00:00 | 31 | 2013.01.04 08:00:00 |
| | 2 | 79 | 2013.01.03 05:00:00 | 2013.01.03 20:00:00 | 0 | |
| | 2 | 80 | 2013.01.03 20:00:00 | 2013.01.04 21:00:00 | 79 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 2 | 95 | 2013.01.04 21:00:00 | 2013.01.07 12:00:00 | 94 | 2013.01.09 18:00:00 |
| | 3 | 1 | 2013.01.01 00:00:00 | 2013.01.01 08:00:00 | 0 | |
| | 3 | 2 | 2013.01.01 08:00:00 | 2013.01.01 22:00:00 | 1 | |
| | 3 | 3 | 2013.01.01 22:00:00 | 2013.01.02 09:00:00 | 2 | 2013.01.02 06:00:00 |
| | 3 | 54 | 2013.01.02 09:00:00 | 2013.01.03 11:00:00 | 53 | |
| | 3 | 132 | 2013.01.03 11:00:00 | 2013.01.05 01:00:00 | 131 | 2013.01.12 06:00:00 |
| | 3 | 105 | 2013.01.05 01:00:00 | 2013.01.06 00:00:00 | 0 | |
| | 3 | 118 | 2013.01.06 00:00:00 | 2013.01.07 03:00:00 | 117 | 2013.01.11 10:00:00 |
| | 3 | 144 | 2013.01.07 03:00:00 | 2013.01.08 06:00:00 | 143 | 2013.01.13 02:00:00 |

Iteration:1 Best Value:72.0
Iteration:6 Best Value:161.0
Iteration:8 Best Value:350.0
Total Iteration Count:109

TotalProcessCompleteTime: 662.0 | AllOperationTardiness: 1135.0 | AvgOperationsTardiness: 1.7145015105740182 | FlowTime: 174.0 | IdleTimeTotal: 0.0 | IdleCost: 0.0 | TotalOutSource: 0.0 | SetupTime: 53.0 | SetupCost: 2862000.0 | OperationCost: 2.1924E7 | TotalOperationCost: 2.4786E7 | DiffJobEndDelivery: 4.0 | Earliness: 718.0 | ScheduledPercentage: 1.0

## EK : ALGORİTMA KODLAR

Paketler;

1- baseUtils : algoritma için data objeleri
2- Solver : algoritmayı uygulayan objeler

DNA.java

```
package baseUtils;

import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

public class DNA
{
	public List<Genome> populationlist;

	public DNA(int populationSize)
	{
		this.populationlist=new ArrayList<Genome>(populationSize);
	}

	public void SortGenomes()
	{
		Collections.sort(populationlist);
	}

	public String toString()
	{

		StringBuilder sb=new StringBuilder();

		for(Genome g:populationlist)
		{
			sb.append(g.toString());

		}
		return sb.toString();
	}

}
```

FeromonMatrix.java

```
package baseUtils;

public class FeromonMatrix extends Object  implements Cloneable{
	public Matrix matrix;

	public FeromonMatrix()
	{
		super();

	}

	public FeromonMatrix(int OprID[],double p_defaultValue)
	{
		super();

		this.matrix = new Matrix(OprID,p_defaultValue);
	}
```

```
	public double GetFeromon(int FromId,int ToId) throws Exception
	{
		return this.matrix.GetValueAt(FromId, ToId);
	}

	public void SetFeromon(int FromId,int ToId,double value) throws
Exception
	{
		this.matrix.SetValueAt(FromId, ToId, value);

	}
	public void SetFeromonPlus(int FromId,int ToId,double value) throws
Exception
	{
		double newVal=this.matrix.GetValueAt(FromId, ToId)+value;

		if(newVal<0.001)
			newVal=0.001;

		else if(newVal>((this.matrix.getMatrixSize()*1000)))
			newVal=(this.matrix.getMatrixSize()*1000);


		this.matrix.SetValueAt(FromId, ToId, newVal);

	}



	public  Object clone()  throws CloneNotSupportedException
	{
		FeromonMatrix newobj=new FeromonMatrix();
		newobj.matrix=(Matrix)this.matrix.clone();
		return newobj;

	}


	public String toString()
	{
		String retval=new String("     ");

		for(MatrixElementMain m:this.matrix.matrix )
		{
			retval+=m.ObjectId + " | ";

		}
		retval+="\n";

		for(MatrixElementMain m:this.matrix.matrix )
		{
			retval+=m.ObjectId + " | ";
			for(MatrixElementSub s:m.SubElements)
			{
				retval+=s.Value+ " | ";
			}
```

```
                retval+="\n";
            }
            return retval;
        }


}
```

GeneralTime.java

```
package baseUtils;

public class GeneralTime extends Object implements Cloneable {
        double time;
        int type;

        public static int _DAY=1;
        public static int _HOUR=2;
        public static int _MINUTE=3;
        public static int _SECOND=4;

        public GeneralTime(double Time, int Type) {
            super();
            this.time = Time;
            this.type = Type;
        }

        public GeneralTime()
        {
            time=0.0;
            type=0;
        }
        public GeneralTime(int Type,int Value)
        {
            type=Type;
            time=Value;
        }

        public double GetTimeInSecond()
        {
            double retval=0.0;

            switch(this.type)
            {
            case 1:
                retval=time*24*60*60;
                break;
            case 2:
                retval=time*60*60;
                break;
            case 3:
                retval=time*60;
                break;
            case 4:
```

```
                retval=time;
                break;
        }

        return retval;

    }

    public static double getTimeIn(double p_second,int p_type)
    {
        double retval=0;

        switch(p_type)
        {
        case 1:
            retval=p_second/86400;
            break;
        case 2:
            retval=p_second/3600;
            break;
        case 3:
            retval=p_second/60;
            break;
        case 4:
            retval=p_second;
            break;
        }

        return retval;
    }

    public double getTimeIn(int p_type)
    {
        double retval=0;

        retval=getTimeIn(GetTimeInSecond() , p_type);

        return retval;
    }

    public Object clone() throws CloneNotSupportedException
    {
        GeneralTime o = (GeneralTime)super.clone();

        return o;

    }


}
```

Genome.java

```
package baseUtils;

import java.util.ArrayList;
import java.util.Collections;
import java.util.Date;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import Solver.FitnessCalculator;

public class Genome  implements Comparable<Genome>, Cloneable{
	int GenomeId;
	public Machine[] MachineList;
	double FitnessValue;
	double lastFitnessValue;
	public FitnessCalculator FC;
	List<Source> OperationSources;
	Date ScheduleDate;
	double MutationRate;
	double MainMutationRate;
	public boolean PrintScheduled=false;

	private Map<Integer,ArrayList<Integer>> SourceMap;
	private Map<Integer,ArrayList<Integer>> ReleatedMap;



	public Genome(int Genome_ID , int MachineCount)
	{
		GenomeId=Genome_ID;
		MachineList=new Machine[MachineCount];
		FitnessValue=0.0;

	}

	@Override
	public int compareTo(Genome compareGenome) throws NullPointerException
{
		// TODO Auto-generated method stub


		return
((Double)this.FitnessValue).compareTo(compareGenome.FitnessValue);

	}


	public Schedulable GetFirstSchedulable(int MachineId)
	{


		Date AvailDate=(Date)this.ScheduleDate.clone();
```

```
		Date retVal=null;

		for(int i=0;i<this.MachineList[MachineId].operations.size();i++)
		{
			Operation
ops=this.MachineList[MachineId].operations.get(i);

			if(ops.IsScheduled)
			{
				AvailDate=(Date)ops.PlanEndDate.clone();// son
işlenen operasyonun bitiş tarihini al
				continue;
			}

			if(ops.AlternativeScheduled || !ops.Active)
				continue;


			retVal=this.FirstSchedulableDate(ops);

			if(retVal==null)
				return null;// operasyon planlanamaz

			if(retVal.after(AvailDate))
				AvailDate=retVal;

			return new Schedulable((Date)AvailDate.clone(), MachineId,
i);
		}

		return null ;// tüm operasyonlar yapıldı
	}


	public Date FirstSchedulableDate(Operation p_opr)
	{
		Date FirstAvailable=(Date)this.ScheduleDate.clone();

		int IsFound=0;



		for(Integer i:getOperationSource((p_opr.getMainOperationId())))
		{

			Source src=OperationSources.get(i);


			// operasyona tanımlı source bul

		/*	if(src.OperationId!=p_opr.MainOperationId)
				continue;
		 */

			IsFound=1;
			switch (src.SourceType)
```

```
		{
			case 0:

				if(src.IsAvailable)
				{

if(FirstAvailable.before(src.DeadLine))
					FirstAvailable=src.DeadLine;

				}
				else
					IsFound=-1;

				break;

			case 1:


				if(FirstAvailable.before(src.DeadLine))

FirstAvailable=(Date)src.DeadLine.clone();
				break;

		}



		if(IsFound==-1)
			return null;



	}


	p_opr.setFirstSchedulableDate((Date)FirstAvailable.clone());

	return FirstAvailable;
}


public Genome clone() throws CloneNotSupportedException
{
	Genome newgen=(Genome)super.clone();

	newgen.ScheduleDate=(Date) ScheduleDate.clone();
	newgen.MachineList=new Machine[MachineList.length];
	int i=0;

	for(Machine m : MachineList)
	{
		newgen.MachineList[i]=(Machine)m.clone();
		i++;
	}

	newgen.OperationSources=new ArrayList<Source>();
```

```
		for(Source s:OperationSources)
		{
			newgen.OperationSources.add((Source)s.clone());

		}

		newgen.ReleatedMap=new
HashMap<Integer,ArrayList<Integer>>(this.ReleatedMap);
		newgen.SourceMap=new
HashMap<Integer,ArrayList<Integer>>(this.SourceMap);


		return newgen;
	}



	public void AddOprSource(Source src)
	{
		if(this.OperationSources==null)
		{
			OperationSources=new ArrayList<Source>();
		}

		OperationSources.add(src);
		Collections.sort(OperationSources);
	}

	public List<Source> getOperationSources() {
		return OperationSources;
	}

	public void setOperationSources(List<Source> operationSources) {
		OperationSources = operationSources;
		Collections.sort(OperationSources);
	}

	public double getFitnessValue() {
		return FitnessValue;
	}

	public void setFitnessValue(Double fitnessValue) {
		lastFitnessValue=FitnessValue;
		FitnessValue = fitnessValue;

	}

	public double getMainMutationRate() {
		return MainMutationRate;
	}

	public void setMainMutationRate(double mainMutationRate) {
		MainMutationRate = mainMutationRate;
	}

	public Date getScheduleDate() {
```

```
		return ScheduleDate;
	}

	public void setScheduleDate(Date scheduleDate) {
		ScheduleDate = scheduleDate;
	}

	public double getMutationRate() {
		return MutationRate;
	}

	public void setMutationRate(double mutationRate) {
		MutationRate = mutationRate;
	}


	public String toString()
	{

		StringBuilder sb=new StringBuilder();

	sb.append("##############################################################
#################################################\n");
		sb.append("Fitness Value: "+ ((Double)FitnessValue).toString()
+"\n");
		for(Machine m : MachineList)
		{
			if(PrintScheduled)
				sb.append(m.toStringScheduled());
			else
				sb.append(m.toStringM());
		}

		if(FC!=null)
		{
			sb.append("\n\n Fitness Values \n");
			sb.append(FC.toString());
			sb.append("\n\n");
		}

	sb.append("##############################################################
#################################################\n");
		return sb.toString();
	}
	@SuppressWarnings("unchecked")
	public ArrayList<Integer> getOperationSource(Integer MainOpId)
	{
		if(SourceMap==null)
			return null;

		ArrayList<Integer> result=SourceMap.get(MainOpId);
		if(result==null)
			return null;

		return (ArrayList<Integer>) result.clone();
	}
```

```
	@SuppressWarnings("unchecked")
	public ArrayList<Integer> getReleatedOprSource(Integer ReleatedId)
	{
		if(ReleatedMap==null)
			return null;

		ArrayList<Integer> result=ReleatedMap.get(ReleatedId);

		if(result==null)
			return null;

		return (ArrayList<Integer>) result.clone();
	}


	public void buildSourceMap()
	{
		SourceMap=new HashMap<Integer,ArrayList<Integer>>();
		ReleatedMap=new HashMap<Integer,ArrayList<Integer>>();

		Collections.sort(OperationSources);

		int lastId=-1;
		int counter=0;
		ArrayList<Integer> indexList=new ArrayList<Integer>();;

		for(Source s:OperationSources)
		{
			if(lastId!=-1 && lastId!=s.OperationId)
			{
				SourceMap.put(new Integer(lastId), indexList);
				indexList=new ArrayList<Integer>();
			}

			indexList.add(counter);


			if(ReleatedMap.get(s.ReleatedId)==null)
				ReleatedMap.put(s.ReleatedId, new
ArrayList<Integer>());
			ReleatedMap.get(s.ReleatedId).add(counter);
			lastId=s.OperationId;
			counter++;
		}
		if(indexList!=null)
			SourceMap.put(lastId, indexList);
	}

}
```

Machine.java

```
package baseUtils;


import java.util.ArrayList;
import java.util.Calendar;
import java.util.List;


public class Machine  implements Cloneable {

	int MachineId,MachineType;
	String Name;

	public FeromonMatrix feromon;
	SetupMatrix setup;
	double IdleCost;
	public double MutationRate=0.50;
	public static final int Type_Normal=0,Type_Fason=1;
	public List<Operation> operations;
	public Machine(int machineID,int machineType)
	{
		new Machine(machineID,machineType," ");
	}


	public Machine(int machineID,int machineType,String Name)
	{

		MachineId=machineID;
		MachineType=machineType;
		Name=new String(Name);

	}


	public void CalculateEnd(int OperationOrder )
	{
		Calendar cal = Calendar.getInstance();
		double setuptime=0;

		cal.setTime(operations.get(OperationOrder).PlanStartDate);
		int willbeAdd = ((Double)
(operations.get(OperationOrder).MachineTime.GetTimeInSecond()*(1-
operations.get(OperationOrder).CompletedPercentage))).intValue();


		if(!operations.get(OperationOrder).IsGoingOn)
		{
			if(OperationOrder>0)
			{
				try {
	setuptime=setup.GetSetup(operations.get(OperationOrder-
1).getSetupKey() , operations.get(OperationOrder).getSetupKey());
				} catch (Exception e) {
```

```
                                    e.printStackTrace();
                              }
                        }
                        else
                        {

      setuptime=operations.get(OperationOrder).getSetupTime().GetTimeInSecon
d();
                        }
                        operations.get(OperationOrder).setSetupSec(setuptime);

                        willbeAdd+=setuptime;
                  }

                  cal.add(Calendar.SECOND,willbeAdd );

                  operations.get(OperationOrder).PlanEndDate=cal.getTime();

      }

      public void setOperations(List<Operation> operations)
      {
            this.operations = operations;
            this.CreateFeromon();

      }

      public void AddOperation(Operation pop)
      {
            if(this.operations==null)
            {
                  this.operations=new ArrayList<Operation>();
            }

            this.operations.add(pop);
            setOperations(operations);

            this.CreateFeromon();


      }

      public void CreateFeromon()
      {

            int i=0;

            int OprIdList[]=new int[operations.size()];

            for(Operation opr:operations)
            {
                  OprIdList[i]=opr.getOperationId();
                  i++;
            }

            this.feromon=new FeromonMatrix(OprIdList, 1.0);
```

```
	}

	public Machine clone() throws CloneNotSupportedException
	 {
		Machine newMach=(Machine)super.clone();
		newMach.Name= new String(Name);
		newMach.feromon=(FeromonMatrix)this.feromon.clone();
		newMach.setup=(SetupMatrix)newMach.setup.clone();
		newMach.operations=new ArrayList<Operation>();

		for(Operation op: operations)
		{
			newMach.operations.add((Operation)op.clone());
		}

		return newMach;
	 }

	public String toString()
	{
		return this.Name;
	}

	public String toStringM()
	{
		String retval=new String();

		retval+="-------------------------------------------------------
--------------------------------";
		retval+="\n Machine : "+MachineId+"\n";



		retval+="Operations: \n ";

		for(Operation op: this.operations)
		{
			retval+="OperationID: "+op.getOperationId()+" , MainOprId
: "+ op.getMainOperationId() +" \t \n "+  op.toStringO()+" \n ";

		}

		return retval;
	}
	public String toStringScheduled()
	{
		String retval=new String();

		for(Operation op: this.operations)
		{
			if(op.IsScheduled)
				retval+="  Machine | "+MachineId+" | "+
op.toStringO()+" \n ";

		}

		return retval;
```

```
}

public Double getIdleCost() {
	return IdleCost;
}


public void setIdleCost(Double idleCost) {
	IdleCost = idleCost;
}


public Double getMutationRate() {
	return MutationRate;
}


public void setMutationRate(Double mutationRate) {
	MutationRate = mutationRate;
}

public int getMachineId() {
	return MachineId;
}


public void setMachineId(int machineId) {
	MachineId = machineId;
}


public int getMachineType() {
	return MachineType;
}


public void setMachineType(int machineType) {
	MachineType = machineType;
}


public List<Operation> getOperations() {
	return operations;
}


public void setName(String name) {
	Name=name;

}


public SetupMatrix getSetup() {
	return setup;
}
```

```
	public void setSetup(SetupMatrix setup) {
		this.setup = setup;
	}
}
```

Matrix.java

```
package baseUtils;

public class Matrix extends Object  implements Cloneable{

	MatrixElementMain[] matrix;
	double DefVal;

	public Matrix(int ObjectArray[],double defaultValue)
	{
		int ElementCount=ObjectArray.length;
		DefVal=defaultValue;
		matrix=new MatrixElementMain[ElementCount];

		for(int i=0;i<ElementCount;i++)
		{

			matrix[i]=new
MatrixElementMain(ObjectArray[i],ElementCount);

			for(int j=0;j<ElementCount;j++)
				matrix[i].getSubElements()[j]=new
MatrixElementSub(ObjectArray[j], defaultValue);

		}

	}


	public double GetValueAt(int FromId,int ToId) throws Exception
	{
		int FromIdex=this.SearchMain(FromId);

		if(FromIdex<0)
			throw new Exception("from operation not found");

		int ToIndex=this.SearchSub(FromIdex,ToId);
		if(ToIndex<0)
			throw new Exception("to operation not found");

		return
this.matrix[FromIdex].getSubElements()[ToIndex].getValue();

	}

	public  void SetValueAt(int FromId,int ToId,double value) throws
Exception
	{
		int FromIdex=this.SearchMain(FromId);
```

```
        if(FromIdex<0)
                throw new Exception("from operation not found");

        int ToIndex=this.SearchSub(FromIdex,ToId);
        if(ToIndex<0)
                throw new Exception("to operation not found");

        this.matrix[FromIdex].getSubElements()[ToIndex].setValue(value);

}

private int SearchMain(int FromId)
{
        for(int i=0;i<matrix.length;i++)
        {
                if(matrix[i].getObjectId()==FromId)
                {
                        return i;

                }

        }

        return -1;

}

private int SearchSub(int Findex ,int ToId) {

        for(int i=0;i<matrix[Findex].getSubElements().length;i++)
        {
                if(matrix[Findex].getSubElements()[i].getObjectId()==ToId)
                        return i;
        }

        return -1;
}

public int getMatrixSize()
{

        return this.matrix.length;
}


public void UpdateAll(double Factor)
{
        for(MatrixElementMain MxM:this.matrix)
        {
                for(MatrixElementSub MxS:MxM.getSubElements())
                {
                        MxS.Value+=Factor;
                }

        }
```

```
	}

	public double getDefVal()
	{
		return DefVal;
	}

	public Object clone() throws CloneNotSupportedException
	{
		Matrix newo=(Matrix)super.clone();

		newo.DefVal=DefVal;
		newo.matrix=matrix.clone();

		for(int i=0;i<newo.matrix.length;i++)
		{
			newo.matrix[i]=(MatrixElementMain) matrix[i].clone();

		}

		return newo;
	}
}
```

MatrixElementMain.java

```
package baseUtils;

public class MatrixElementMain implements Cloneable {

	int ObjectId;
	MatrixElementSub[] SubElements;



	public MatrixElementMain(int objectId,int SubCount)
	{
		ObjectId=objectId;
		SubElements=new MatrixElementSub[SubCount];
	}

	public int getObjectId() {
		return ObjectId;
	}

	public void setObjectId(int objectId) {
		ObjectId = objectId;
	}

	public MatrixElementSub[] getSubElements() {
		return SubElements;
	}

	public void setSubElements(MatrixElementSub[] subElements) {
```

```
			SubElements = subElements;
		}
		public Object clone() throws CloneNotSupportedException
		{
			MatrixElementMain newobj=new MatrixElementMain(this.ObjectId
,SubElements.length );
			newobj.SubElements=this.SubElements.clone();

			for(int i=0;i<SubElements.length;i++)
			{
				newobj.SubElements[i]=(MatrixElementSub)
SubElements[i].clone();

			}

			return newobj;
		}

}
```

Source.java

```
package baseUtils;

import java.text.SimpleDateFormat;
import java.util.Date;


public class Source extends Object implements Cloneable,Comparable<Source> {

	int SourceType; // 0:operasyon,1:hammadde , 3:sipariş
	Date DeadLine;
	int OperationId;
	int MachineId;// paralel veya alternatifin makine sırası
	int ReleatedId;
	boolean IsAvailable;
	public static final int _Operation=0;
	public static final int _RawMaterial=1;
	public static final int _Alternative=2;
	public static final int _Order=3;
	public static final int _Paralell=4;

	public Source( int sourceType,int operationId, Date deadLine,  int
releatedId) {
		super();
		SourceType = sourceType;
		DeadLine = deadLine;
		OperationId = operationId;
		ReleatedId = releatedId;
	}

	public Source( int sourceType,int operationId, Date deadLine) {
		super();
```

```
		SourceType = sourceType;
		DeadLine = deadLine;
		OperationId = operationId;
	}


	public Source(int sourceType,int operationId,  Date deadLine,  int
releatedId,int machineId)
	{
		super();
		SourceType = sourceType;
		DeadLine = deadLine;
		OperationId = operationId;
		ReleatedId = releatedId;
		MachineId=machineId;
	}


	public Source()
	{
		this.SourceType=0;
		this.DeadLine=new Date();
		MachineId=0;
	}


	public int getSourceType() {
		return SourceType;
	}

	public void setSourceType(int sourceType) {
		if(sourceType==_Order)
			IsAvailable=true;
		else
			IsAvailable=false;

		SourceType = sourceType;
	}
	public Date getDeadLine() {
		return DeadLine;
	}

	public void setDeadLine(Date deadLine) {
		DeadLine = deadLine;
	}

	public int getMachineId() {
		return MachineId;
	}

	public void setMachineId(int machineId) {
		MachineId = machineId;
	}



	public Boolean getIsAvailable() {
```

```
		return IsAvailable;
	}

	public void setIsAvailable(Boolean isAvailable) {
		IsAvailable = isAvailable;
	}

	public int getOperationId() {
		return OperationId;
	}
	public Integer getOperationId2() {
		return Integer.valueOf(OperationId) ;
	}

	public void setOperationId(int operationId) {
		OperationId = operationId;
	}

	public int getReleatedId() {
		return ReleatedId;
	}

	public void setReleatedId(int releatedId) {
		ReleatedId = releatedId;
	}


	public Object clone() throws CloneNotSupportedException
	{
		Source s=(Source)super.clone();
		s.DeadLine=(Date)this.DeadLine.clone();
		return s;
	}

	public String toString()
	{
		String retval;
		retval="SourceType:";
		switch(SourceType)
		{
		case 0:
			retval+="Operasyon";
			break;
		case 1:
			retval+="hammadde";break;
		case 2:
			retval+="Alternatif";break;
		case 3:
			retval+="Order";break;
		case 4:
			retval+="Paralell";break;
		}


	retval+=",OperationId:"+OperationId+",MachineId:"+MachineId+",Releated
Id:"+ReleatedId+",IsAvailable:";
```

```
		if(IsAvailable)
			retval+="aktif";
		else
			retval+="pasif";

		SimpleDateFormat dFormatter = new SimpleDateFormat("yyyy.MM.dd
HH:mm:ss");
		retval+=dFormatter.format(DeadLine);

		return retval;

	}

	@Override
	public int compareTo(Source o) {
		return this.getOperationId2().compareTo(o.getOperationId2());
	}
}
```

MatrixElementSub.java

```
package baseUtils;

public class MatrixElementSub implements Cloneable{


	int ObjectId;
	double Value;


	public int getObjectId() {
		return ObjectId;
	}
	public void setObjectId(int objectId) {
		ObjectId = objectId;
	}
	public double getValue() {
		return Value;
	}
	public void setValue(double value) {
		Value = value;
	}
	public MatrixElementSub(int objectId, double value) {
		super();
		ObjectId = objectId;
		Value = value;
	}

	public MatrixElementSub(int objectId) {
		new MatrixElementSub(objectId,0.0);
	}

	public Object clone() throws CloneNotSupportedException
	{
```

```
            Object o=super.clone();

            return o;
      }
}
```

Operation.java

```
package baseUtils;

import java.text.SimpleDateFormat;
import java.util.Date;


public class Operation extends Object implements Cloneable{

      int OperationId;
      int MainOperationId;
      int OperationRelation;
      public boolean Active,IsScheduled,IsGoingOn,AlternativeScheduled;
      public double CompletedPercentage;
      double MachineCost,SetupCost,OutSourceQuantity,OutSourceCost;
      public GeneralTime MachineTime,SetupTime,Act1Time,Act2Time;
      int SetupKey;
      double MachineSec,SetupSec,Act1Sec,Act2Sec,CalcSetup;
      public Date PlanStartDate,PlanEndDate;
      Date FirstSchedulableDate;
      int Level,AfterOperation;
      String OptText;
      double Earliness=0,DiffJobEndDelivery=0;
      public double TotalOperationtime=0;
      public Date OrderDeliveryDate;



      public Operation(Integer operationId, Integer mainOperationId,
                  Integer operationRelation,Boolean isGoingOn,Double
completedPercentage,
                  Double machineCost,Double setupCost, Double
outSourceQuantity,
                  Double outSourceCost, GeneralTime machineTime, GeneralTime
setupTime,
                  GeneralTime act1Time, GeneralTime act2Time, int setupKey,
                  Double machineSec, Double setupSec, Double act1Sec, Double
act2Sec) {
            super();

            OperationId = operationId;
            MainOperationId = mainOperationId;
            OperationRelation = operationRelation;
            Active = false;
            MachineCost = machineCost;
            SetupCost = setupCost;
```

```
        OutSourceQuantity = outSourceQuantity;
        OutSourceCost = outSourceCost;
        MachineTime = machineTime;
        SetupTime = setupTime;
        Act1Time = act1Time;
        Act2Time = act2Time;
        SetupKey = setupKey;
        IsScheduled=false;
        IsGoingOn=isGoingOn;
        CompletedPercentage=completedPercentage;

        MachineSec = machineTime.GetTimeInSecond();
        SetupSec = setupTime.GetTimeInSecond();
        Act1Sec = act1Time.GetTimeInSecond();
        Act2Sec = act2Time.GetTimeInSecond();

}


public Operation ()
{
        CalcSetup=0.0;
        IsScheduled=false;
        Level=0;

}

public Integer getOperationId() {
        return OperationId;
}


public void setOperationId(Integer operationId) {
        OperationId = operationId;
}


public Integer getMainOperationId() {
        return MainOperationId;
}


public void setMainOperationId(Integer mainOperationId) {
        MainOperationId = mainOperationId;
}


public Integer getOperationRelation() {
        return OperationRelation;
}


public void setOperationRelation(Integer operationRelation) {
        OperationRelation = operationRelation;
}
```

```
public Boolean getActive() {
	return Active;
}


public void setActive(Boolean aktive) {
	Active = aktive;
}


public Double getMachineCost() {
	return MachineCost;
}


public void setMachineCost(Double machineCost) {
	MachineCost = machineCost;
}


public Double getSetupCost() {
	return SetupCost;
}


public void setSetupCost(Double setupCost) {
	SetupCost = setupCost;
}


public Double getOutSourceQuantity() {
	return OutSourceQuantity;
}


public void setOutSourceQuantity(Double outSourceQuantity) {
	OutSourceQuantity = outSourceQuantity;
}


public Double getOutSourceCost() {
	return OutSourceCost;
}


public void setOutSourceCost(Double outSourceCost) {
	OutSourceCost = outSourceCost;
}


public GeneralTime getMachineTime() {
	return MachineTime;
}


public void setMachineTime(GeneralTime machineTime) {
	MachineTime = machineTime;
```

```
}


public GeneralTime getSetupTime() {
	return SetupTime;
}


public void setSetupTime(GeneralTime setupTime) {
	SetupTime = setupTime;
}


public GeneralTime getAct1Time() {
	return Act1Time;
}


public void setAct1Time(GeneralTime act1Time) {
	Act1Time = act1Time;
}


public GeneralTime getAct2Time() {
	return Act2Time;
}


public void setAct2Time(GeneralTime act2Time) {
	Act2Time = act2Time;
}


public int getSetupKey() {
	return SetupKey;
}


public void setSetupKey(int setupKey) {
	SetupKey = setupKey;
}

public Boolean getIsScheduled() {
	return IsScheduled;
}


public void setIsScheduled(Boolean isScheduled) {
	IsScheduled = isScheduled;
}


public Boolean getIsGoingOn() {
	return IsGoingOn;
}
```

```
public void setIsGoingOn(Boolean isGoingOn) {
	IsGoingOn = isGoingOn;
}


public Date getFirstSchedulableDate() {
	return FirstSchedulableDate;
}


public void setFirstSchedulableDate(Date firstSchedulableDate) {
	FirstSchedulableDate = firstSchedulableDate;
}


public Boolean getAlternativeScheduled() {
	return AlternativeScheduled;
}


public void setAlternativeScheduled(Boolean alternativeScheduled) {
	AlternativeScheduled = alternativeScheduled;
}

public int getLevel() {
	return Level;
}


public void setLevel(int level) {
	Level = level;
}


public int getAfterOperation() {
	return AfterOperation;
}


public void setAfterOperation(int afterOperation) {
	AfterOperation = afterOperation;
}


public String getOptText() {
	return OptText;
}


public void setOptText(String optText) {
	OptText = optText;
}



public double getEarliness() {
	return Earliness;
```

```
	}


	public void setEarliness(double earliness) {
		Earliness = earliness;
	}


	public double getDiffJobEndDelivery() {
		return DiffJobEndDelivery;
	}


	public void setDiffJobEndDelivery(double diffJobEndDelivery) {
		DiffJobEndDelivery = diffJobEndDelivery;
	}


	public double getSetupSec() {
		return SetupSec;
	}


	public void setSetupSec(double setupSec) {
		SetupSec = setupSec;
	}
	public Object clone() throws CloneNotSupportedException
	{
		Operation newo=(Operation)super.clone();

		newo.PlanStartDate=(Date) this.PlanStartDate.clone();
		newo.PlanEndDate=(Date) this.PlanEndDate.clone();

	newo.setFirstSchedulableDate((Date)this.FirstSchedulableDate.clone());
		newo.OrderDeliveryDate=(Date) this.OrderDeliveryDate.clone();

		return newo;

	}

	public String toStringO()
	{
		String retval=new String();


		if(!Active)
			return "Passive";

		if(!IsScheduled)
			return "Cannot Scheduled!!!";

		SimpleDateFormat dFormatter = new SimpleDateFormat("yyyy.MM.dd
HH:mm:ss");

		retval+=String.format("OperationId| %5d  | ", OperationId ) ;
		retval+=String.format("MainOperationId| %5d | ", MainOperationId
) ;
```

```
            retval+=String.format("Start|%20s | ",
dFormatter.format(PlanStartDate) ) ;
            retval+=String.format("End|%20s | ",
dFormatter.format(PlanEndDate ) ) ;
            retval+=String.format("OperationTime|%10.1f |
",MachineTime.GetTimeInSecond()) ;
            retval+=String.format("SetupTime|%10.1f | ",SetupSec) ;

            retval+=String.format("TotalOperationtime| %10.1f | ",
TotalOperationtime ) ;
            retval+=String.format("AfterOperation| %5d | ", AfterOperation )
;


            if(Earliness>0)
            {
                  retval+=String.format("Earliness         | %10.1f | ",
Earliness ) ;
                  retval+=String.format("OrderDeliveryDate| %20s | ",
dFormatter.format(OrderDeliveryDate ) ) ;
            }
            else if(DiffJobEndDelivery>0)
            {
                  retval+=String.format("DiffJobEndDelivery| %10.1f | ",
DiffJobEndDelivery ) ;
                  retval+=String.format("OrderDeliveryDate| %20s | ",
dFormatter.format(OrderDeliveryDate ) ) ;
            }
            return retval;

      }

      public String toString()
      {
            return MainOperationId+":"+OptText;
      }



}
```

Schedulable.java

```
package baseUtils;

import java.text.SimpleDateFormat;
import java.util.Date;

public class Schedulable implements Comparable<Schedulable> ,  Cloneable {
      Date AvailDate;
      int MachineOrder, OperationOrder;
```

```
	public Schedulable(Date availDate, int machineOrder, int
operationOrder) {
		super();
		AvailDate = availDate;
		MachineOrder = machineOrder;
		OperationOrder = operationOrder;
	}


	@Override
	public int compareTo(Schedulable arg0) {
		// TODO Auto-generated method stub
		return this.AvailDate.compareTo(arg0.AvailDate);

	}

	public Object clone() throws CloneNotSupportedException
	{
		Schedulable o = (Schedulable)super.clone();
		o.AvailDate=(Date)o.AvailDate.clone();

		return o;
	}

	public String toString()
	{
		SimpleDateFormat dFormatter = new SimpleDateFormat("yyyy.MM.dd
HH:mm:ss");

		return "MachineOrder: "+MachineOrder +", OperationOrder:
"+OperationOrder + " , Availdate : "+dFormatter.format(AvailDate) ;

	}



	public Date getAvailDate() {
		return AvailDate;
	}
	public void setAvailDate(Date availDate) {
		AvailDate = availDate;
	}
	public int getMachineOrder() {
		return MachineOrder;
	}
	public void setMachineOrder(int machineOrder) {
		MachineOrder = machineOrder;
	}
	public int getOperationOrder() {
		return OperationOrder;
	}
	public void setOperationOrder(int operationOrder) {
		OperationOrder = operationOrder;
	}

}
```

SetupMatrix.java

```
package baseUtils;


public class SetupMatrix extends Object  implements Cloneable{

	Matrix setupMatx;

	public SetupMatrix(int p_element[],double p_default)
	{
		this.setupMatx=new Matrix(p_element, p_default);
	}


	public double GetSetup(int FromId,int ToId) throws Exception
	{
		return this.setupMatx.GetValueAt(FromId, ToId);
	}

	public void SetSetup(int FromId,int ToId,double value) throws
Exception
	{
		this.setupMatx.SetValueAt(FromId, ToId, value);
	}

	public Object clone() throws CloneNotSupportedException
	{
		SetupMatrix newm=(SetupMatrix)super.clone();

		newm.setupMatx=(Matrix)newm.setupMatx.clone();

		return newm;

	}
}
```

ScheduleOperations.java

```
package Solver;

import java.util.ArrayList;
import java.util.Collections;
import java.util.Date;
import java.util.List;
import java.util.concurrent.RecursiveTask;

import baseUtils.Genome;
import baseUtils.Machine;
import baseUtils.Operation;
import baseUtils.Schedulable;
import baseUtils.Source;

public class ScheduleOperations  extends RecursiveTask<Genome>
{
```

```
	/**
	 *
	 */
	private static final long serialVersionUID = 8907332140603856954L;

	Genome currentGen;
	Date ScheduleDate;


	public ScheduleOperations(Genome currentGen,Date scheduleDate) {
		super();
		this.currentGen = currentGen;
		this.ScheduleDate=scheduleDate;
	}

	@Override
	protected Genome compute() {


		return this.Schedule();
	}


	public Genome  Schedule()
	{
		this.RefreshOperations();

		this.GiveFirstOrderGoingOnJobs();

		while(true)
		{
			Schedulable willSch=this.AssignMinStartDate();

			if(willSch==null)
				break;

			this.ScheduleOpr(willSch);
		}



		return currentGen;
	}



	private void ScheduleOpr(Schedulable willSch) {

		Operation
opr=currentGen.MachineList[willSch.getMachineOrder()].operations.get(willSch.
getOperationOrder());

		opr.setIsScheduled(true);
		opr.setActive(true);
		opr.PlanStartDate=(Date)willSch.getAvailDate().clone();
	currentGen.MachineList[willSch.getMachineOrder()].CalculateEnd(willSch
.getOperationOrder());
```

```
		currentGen.MachineList[willSch.getMachineOrder()].operations.set(willS
ch.getOperationOrder(),opr);
			this.UpdateSourceNew(opr);
		}



		private void UpdateSource(Operation popr) {

			for(Source src:currentGen.getOperationSources() )
			{
				if(src.getSourceType()==Source._Operation &&
src.getReleatedId()==popr.getMainOperationId())
				{
					src.setDeadLine(popr.PlanEndDate);
					src.setIsAvailable(true);
				}


				if(src.getSourceType()==Source._Alternative &&
popr.getMainOperationId()==src.getOperationId() &&
popr.getOperationId()!=src.getReleatedId() )
				{

					Machine
m=currentGen.MachineList[src.getMachineId()];


					for(Operation o:m.operations)
					{
						if(o.getOperationId()!=src.getReleatedId())
							continue;

						o.setAlternativeScheduled(true);
						o.setActive(false);
					}
				}
			}
		}


		private void UpdateSourceNew(Operation popr) {

			ArrayList<Integer>
list=currentGen.getReleatedOprSource(popr.getMainOperationId());
			if(list!=null)
			{
				for(Integer i:list)
				{
					Source src=currentGen.getOperationSources().get(i);

					if(src.getSourceType()==Source._Operation &&
src.getReleatedId()==popr.getMainOperationId())
					{
						src.setDeadLine(popr.PlanEndDate);
						src.setIsAvailable(true);
```

```
                        }
                    }
                }
                list=currentGen.getOperationSource(popr.getMainOperationId());
                if(list!=null)
                {
                    for(Integer i:list)
                    {
                        Source src=currentGen.getOperationSources().get(i);
                        if(src.getSourceType()==Source._Alternative &&
popr.getMainOperationId()==src.getOperationId() &&
popr.getOperationId()!=src.getReleatedId() )
                        {
                            Machine m =
currentGen.MachineList[src.getMachineId()];

                            for(Operation o:m.operations)
                            {

        if(o.getOperationId()!=src.getReleatedId())
                                    continue;
                                o.setAlternativeScheduled(true);
                                o.setActive(false);

                            }


                        }


                    }
                }

        }


        private Schedulable AssignMinStartDate() {

            List<Schedulable> schList=new ArrayList<Schedulable>();

            for(int i=0;i<this.currentGen.MachineList.length;i++)
            {
                Schedulable r_sch=this.currentGen.GetFirstSchedulable(i);
                if(r_sch==null)
                    continue;

                schList.add(r_sch);

            }

            if(schList.size()==0)
                return null;
            Collections.shuffle(schList);
            Collections.sort(schList);

            return schList.get(0);
        }
```

```
	public void RefreshOperations()
	{
		for(Machine mach:currentGen.MachineList)
		{
			for(Operation opr:mach.operations)
			{
				opr.setIsScheduled(false);
				opr.setActive(true);
				opr.setAlternativeScheduled(false);
				opr.setEarliness(0);
				opr.setDiffJobEndDelivery(0);
			}
		}
		for(Source src:currentGen.getOperationSources())
		{
			if(src.getSourceType()==Source._Operation)
				src.setIsAvailable(false);
		}
	}
	public void GiveFirstOrderGoingOnJobs()
	{
		for(Machine wc:currentGen.MachineList)
		{

			for(int i=0;i<wc.operations.size();i++)
			{
				if(wc.operations.get(i).getIsGoingOn() && i>0)
				{
					Collections.swap(wc.operations, 0, i);
					wc.operations.get(i).setIsScheduled(true);

	wc.operations.get(i).PlanStartDate=ScheduleDate;
					wc.CalculateEnd(i);
					break;
				}
			}
		}

	}

}
```

Maintenance.java

```
package Solver;

import java.util.Collections;
import java.util.concurrent.RecursiveTask;

import baseUtils.Genome;
import baseUtils.Machine;

public class Maintenance extends RecursiveTask<Genome> {
```

```
	/**
	 *
	 */
	private static final long serialVersionUID = 6193234427083971819L;
	Genome genome;


	@Override
	protected Genome compute() {
		this.perform();

		return genome;
	}


	public Maintenance(Genome pgen)
	{
		this.genome=pgen;

	}


	public void perform()
	{


		for(Machine m:genome.MachineList)
		{

			for(int i=0;i<m.operations.size();i++)
			{
				this.CheckForSeq(m, i);
			}

		}

	}

	private void CheckForSeq(Machine m , int CurrOpSeq)
	{
		for(int x=CurrOpSeq+1;x<m.operations.size();x++)
		{

	if(m.operations.get(CurrOpSeq).getAfterOperation()==m.operations.get(x
).getMainOperationId()   )
			{
				Collections.swap(m.operations, CurrOpSeq, x);
				this.CheckForSeq(m, CurrOpSeq);
				break;
			}

		}

	}


}
```

Mutation.java

```
package Solver;

import java.util.Collections;
import java.util.Random;
import java.util.concurrent.RecursiveTask;

import baseUtils.Machine;

public class Mutation extends RecursiveTask<Machine> {
	private static final long serialVersionUID = -7942436616596160057L;

	Machine MutationMachine;
	double MutationRate;

	public Mutation(Machine p_pach,double mrate)
	{
		this.MutationMachine=p_pach;
		this.MutationRate=mrate;

	}

	@Override
	protected Machine compute() {

		return this.MutationCompute();
	}

	public Machine MutationCompute() {

		int
WillBeMutate=((Double)(MutationMachine.operations.size()*MutationRate)).intVa
lue();

		if(WillBeMutate<=0)
			WillBeMutate=1;


		if(WillBeMutate==1 && MutationMachine.operations.size()<2)
			return MutationMachine;

		Random rand=new Random();

		for(int i=0;i<WillBeMutate;i++)
		{
			int
ChangeOperationNo=rand.nextInt(MutationMachine.operations.size()-1)+1;
			int
NextOperation=this.GetNextOperation(ChangeOperationNo);
			Collections.swap(MutationMachine.operations,
ChangeOperationNo, NextOperation);
		}


		return MutationMachine;
	}
```

```
	private int GetNextOperation(int changeOperationNo) {
		Random rand = new Random();

		int nextopr=rand.nextInt(MutationMachine.operations.size());
		while(changeOperationNo==nextopr)
		{
			nextopr=rand.nextInt(MutationMachine.operations.size());
		}
		return nextopr;
	}

}
```

GenProcess.java

```
package Solver;

import java.util.ArrayList;
import java.util.Collections;
import java.util.Date;
import java.util.LinkedList;
import java.util.List;
import java.util.concurrent.RecursiveTask;

import baseUtils.Genome;
import baseUtils.Machine;

public class GenProcess extends RecursiveTask<Genome> {

	/**
	 *
	 */
	private static final long serialVersionUID = 6279559553031162373L;
	Genome currGen;
	List<Genome> GenList;
	int AntCount;
	public static int _AntMutation=0,_GAMutation=1,_BothMutation=2;
	public double mutationRate;
	public int MutationType=-1;
	public int Level=2;

	List<RecursiveTask<Genome>> forks ;

	public GenProcess(Genome currGen,int pAntCount) {
		super();
		this.currGen = currGen;
		this.AntCount=pAntCount;

	}


	@Override
	protected Genome compute() {
		return ProcessOnGen();

	}
```

```
	public Genome ProcessOnGen()
	{
		int size=AntCount;
		if(MutationType==2)
			size=size*2;

		GenList=new ArrayList<Genome>(size);
		forks =new LinkedList<>();

		try {

			for(int i=0;i<AntCount;i++)
			{

				if(Level==1)
				{
					GenProcess gp2=new GenProcess(currGen, 1);
					gp2.MutationType=MutationType;
					gp2.mutationRate=mutationRate;
					gp2.Level=2;
					forks.add(gp2);
					gp2.fork();
					continue;
				}

				Genome newGen=currGen.clone();

				if(MutationType==_BothMutation ||
MutationType==_AntMutation)
				{
					this.DoAll(newGen, _AntMutation);
					GenList.add(newGen);
				}

				if(MutationType==_BothMutation ||
MutationType==_GAMutation)
				{
					newGen=newGen.clone();
					this.DoAll(newGen, _GAMutation);
					GenList.add(newGen);
				}
			}


		} catch (CloneNotSupportedException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}catch(Exception e )
		{
			e.printStackTrace();
		}

		if(Level==1)
		{
			for(RecursiveTask<Genome> task : forks)
			{
				GenList.add(task.join());
```

```
                }

            }

            Collections.shuffle(GenList);
            Collections.sort(GenList);


            try {
                Genome bestGen=(Genome)GenList.get(GenList.size()-
1).clone();
                GenList.clear();
                System.gc();
                return bestGen;
            } catch (CloneNotSupportedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
                return null;
            }
        }

        private void DoAll(Genome p_gen, int e_type)
        {
            this.Evolution(p_gen,e_type);
            // Mutation
            this.Maintenance(p_gen);
            // maintain
            this.ScheduleOps(p_gen);
            // operation maintenance
            this.CalcFitness(p_gen);
        }


        private void CalcFitness(Genome currGen2) {
            FitnessCalculator FC=new FitnessCalculator(currGen2);
            currGen2.setFitnessValue(FC.CalcFitness());
        }


        private void ScheduleOps(Genome currGen2)
        {
            ScheduleOperations SO=new ScheduleOperations(currGen2,
currGen.getScheduleDate());
            SO.Schedule();
        }


        private void Evolution(Genome pgenome,int E_type)
        {
            for(Machine m : pgenome.MachineList)
            {
                if(E_type==_AntMutation)
                {
                    AntMutation mutate=new AntMutation(m);
                    mutate.SchStartDate= (Date)
currGen.getScheduleDate().clone();
                    m=mutate.MutationCompute();
```

```
                    }
                    else
                    {
                          Mutation mutate=new Mutation(m,mutationRate);
                          m=mutate.MutationCompute();
                    }

             }

      }

      private void Maintenance(Genome pgenome)
      {
             Maintenance mp=new Maintenance(pgenome);
             mp.perform();


      }

}
```

GaAntSolver.java

```
package Solver;


import java.util.Collections;
import java.util.Date;
import java.util.LinkedList;
import java.util.List;
import java.util.Random;
import java.util.concurrent.*;

import org.apache.commons.math3.stat.descriptive.DescriptiveStatistics;

import baseUtils.DNA;
import baseUtils.Genome;
import baseUtils.Machine;

public class GaAntSolver extends  RecursiveAction {

      /**
       *
       */
      private static final long serialVersionUID = 1L;

      boolean forkjoin=false;
      int iteration,iterationCount,MutationChange;
      int populationSize;
      DNA population;
      DNA bestPopulation;
      int BestPopulationSize;
      int AntCount=1;
      int IterationWithoutHeal=0;
      public int MutationType=GenProcess._BothMutation;
```

```
double Q=0.1,rho=1.00;
double GQ=0.0,Grho=1.50;

double CrossOverRate=0.25;
double MainMutationRate=0.50;
double MutationDecreaseFactor=0.005;
Date ScheduleDate;
Genome Problem;

List<RecursiveTask<Genome>> forks ;
DescriptiveStatistics stats;
DescriptiveStatistics BestStats;


public GaAntSolver()
{
	iterationCount=0;
	iteration=0;
	populationSize=0;
	ScheduleDate=new Date();
	forkjoin=false;
}



@Override
protected void compute() {

	forkjoin=true;
	this.run();

}


public void run() {

		Problem.buildSourceMap();

		// generate first population
		// operation maintenance
		// Calculate Fitness Value
		this.GenerateFirstPopulation();

		// System.out.println(population.toString());
		BestStats=new DescriptiveStatistics();
		IterationWithoutHeal=0;
		double lastBest;

		lastBest=-999999999;

		while(iteration<iterationCount)
		{
			iteration++;

			if(forkjoin)
```

```
                        forks =new LinkedList<>();

                    this.GenerateNewPopulation();


                    for(int i=0;i<populationSize;i++)
                    {
                        population.populationlist.set(i,
this.ProcessOnGen(population.populationlist.get(i)));

                    }


                    if(forkjoin)
                    {
                        population.populationlist.clear();

                         for (RecursiveTask<Genome> task : forks)
                         {

population.populationlist.add(task.join());
                         }
                    }

                    // Collect Best  Genomes
                    this.KeepBests();


                    // Global Feromon Upgrade
                    this.FeromonUpdate();

                    IterationWithoutHeal++;


      if(lastBest<bestPopulation.populationlist.get(BestPopulationSize-
1).getFitnessValue())
                    {

      lastBest=bestPopulation.populationlist.get(BestPopulationSize-
1).getFitnessValue();
                        BestStats.addValue(lastBest);
                        System.out.println("Iteration:"+iteration+"
Best Value:"+bestPopulation.populationlist.get(BestPopulationSize-
1).getFitnessValue());
                        IterationWithoutHeal=0;
                    }
                    stats = new DescriptiveStatistics();

                    for(Genome g:population.populationlist)
                    {
                        stats.addValue(g.getFitnessValue());
                    }
                    MainMutationRate-=MutationDecreaseFactor;

                    if(MainMutationRate<0.05)
                        MainMutationRate=0.05;
```

```
                        System.gc();

                        if(IterationWithoutHeal>100)
                                break;
                }

                System.out.println("Total Iteration Count:"+iteration);
                System.out.print(bestPopulation.toString());


        }


        private void GenerateFirstPopulation()
        {

                try {
                        Random rnd=new Random();

                        forks =new LinkedList<>();

                        for(int i=0;i<populationSize;i++)
                        {
                                Genome b=(Genome)Problem.clone();

                                for(Machine m:b.MachineList)
                                {
                                        Collections.shuffle(m.operations, rnd);
                                }

                                b=this.ProcessOnGen(b);

                                if(!forkjoin)
                                        population.populationlist.add(b);

                        }

                        if(forkjoin)
                        {
                                population.populationlist.clear();

                                 for (RecursiveTask<Genome> task : forks)
                                 {
                                        population.populationlist.add(task.join());
                                 }
                        }

                        population.SortGenomes();

                        bestPopulation=new DNA(BestPopulationSize);

                        for(int i=0;i<BestPopulationSize;i++)
                        {
                                bestPopulation.populationlist.add((Genome)
population.populationlist.get(populationSize-i-1).clone());
                        }
```

```
		} catch (CloneNotSupportedException e) {

			e.printStackTrace();
		}

	}

	private void KeepBests() {

		DNA tempPop=new DNA(BestPopulationSize);
		population.SortGenomes();
		bestPopulation.SortGenomes();

		for(int i=1;i<=BestPopulationSize;i++)
		{

	tempPop.populationlist.add(bestPopulation.populationlist.get(bestPopul
ation.populationlist.size()-i));

	tempPop.populationlist.add(population.populationlist.get(population.po
pulationlist.size()-i ));
		}

		tempPop.SortGenomes();

		bestPopulation=new DNA(BestPopulationSize);

		try {
			for(int i=0;i<BestPopulationSize;i++)
			{

	bestPopulation.populationlist.add((Genome)tempPop.populationlist.get(i
+BestPopulationSize).clone());
			}
		} catch (CloneNotSupportedException e) {

			e.printStackTrace();
		}
		return;

	}
	private Genome ProcessOnGen(Genome currGenome) {


		GenProcess gp;

		try {
			Double GenMutationRate=MainMutationRate;

			gp = new GenProcess((Genome)currGenome.clone(),AntCount);

			if(MutationType==GenProcess._BothMutation)
			{
				if(iteration>(MutationChange))
					gp.MutationType=GenProcess._GAMutation;
				else
					gp.MutationType=MutationType;
```

```
                    }
                    else
                          gp.MutationType=MutationType;

                    if(stats!=null && currGenome.getFitnessValue()!=0 )
                    {
                          double diff=currGenome.getFitnessValue() -
stats.getMean();
                          double temp1=0.0;

                          temp1=((diff/stats.getStandardDeviation()));

                          GenMutationRate=GenMutationRate-
(GenMutationRate*temp1);
                          if(GenMutationRate>0.75)
                                GenMutationRate=0.75;
                          else if(GenMutationRate<0.05)
                                GenMutationRate=0.05;

                    }


                    gp.mutationRate=GenMutationRate;
                    gp.Level=2;


                    if(forkjoin)
                    {
                          forks.add(gp);
                          gp.fork();
                    }
                    else
                          currGenome=gp.ProcessOnGen();

             } catch (CloneNotSupportedException e) {

                    e.printStackTrace();
                    return null;
             }


             return currGenome;
       }


       private void GenerateNewPopulation() {

             this.AllCrossOver();
       }
       private void FeromonUpdate()
       {

             for(Genome currGenome  : population.populationlist)
             {
                    for(int i=0;i<currGenome.MachineList.length;i++)
                    {
                          Machine m=currGenome.MachineList[i];
```

```
                        try {

                        // local güncellemeler

                              FeromonUpdater FrUpL=new FeromonUpdater(m,
Q,rho);
                              FrUpL.Calc();
                        // Global güncellemeler

                              FeromonUpdater FrUpG=new
FeromonUpdater(bestPopulation.populationlist.get(BestPopulationSize-
1).MachineList[i].operations,m.feromon , GQ,Grho);
                              FrUpG.Calc();

                        } catch (Exception e) {

                              e.printStackTrace();
                        }

                  }

            }

            return;
      }


      private void AllCrossOver()
      {
            Random rand = new Random();
            /*sort population on fitness value */
            population.SortGenomes();
            /*how many genomes will be created*/
            int
crossoverGenome=((Double)(populationSize*CrossOverRate)).intValue();
            /* others are stayed */
            int stayedGenome=populationSize-crossoverGenome;

            for(int i=0;i<crossoverGenome;i++)
            {
                  int first=rand.nextInt(stayedGenome);
                  int second=0;
                  while(true)
                  {
                        second=rand.nextInt(stayedGenome);
                        if(first!=second)
                              break;

                  }
                  CrossOver crossGenomes=new
CrossOver(population.populationlist.get(first),population.populationlist.get(
second));
                  Genome newGen=crossGenomes.compute();

                  population.populationlist.set((stayedGenome+i), newGen);
```

```
	}

}

public int getIterationCount() {
	return iterationCount;
}

public void setIterationCount(int iterationCount) {
	this.iterationCount = iterationCount;
	this.MutationChange = iterationCount/2;
}

public int getAntCount() {
	return AntCount;
}

public void setAntCount(int antCount) {
	AntCount = antCount;
}

public int getBestPopulationSize() {
	return BestPopulationSize;
}

public void setBestPopulationSize(int bestPopulationSize) {
	if(bestPopulationSize<populationSize)
		BestPopulationSize=bestPopulationSize;
	else
		BestPopulationSize = populationSize;
}
public int getPopulationSize() {
	return populationSize;
}

public void setPopulationSize(int populationSize) {
	this.populationSize = populationSize;
	this.population=new DNA(populationSize);

}

public int getIteration() {
	return iteration;
}

public void setIteration(int iteration) {
	this.iteration = iteration;
}


public double getCrossOverRate() {
	return CrossOverRate;
}

public void setCrossOverRate(double crossOverRate) {
	CrossOverRate = crossOverRate;
```

```
	}

	public double getMainMutationRate() {
		return MainMutationRate;
	}

	public void setMainMutationRate(double mainMutationRate) {
		MainMutationRate = mainMutationRate;
	}

	public Date getScheduleDate() {
		return ScheduleDate;
	}

	public void setScheduleDate(Date scheduleDate) {
		ScheduleDate = scheduleDate;
	}

	public Genome getProblem() {
		return Problem;
	}

	public void setProblem(Genome problem) {
		Problem = problem;
	}

	public int getMutationChange() {
		return MutationChange;
	}



	public void setMutationChange(int mutationChange) {
		MutationChange = mutationChange;
	}

}
```

FitnessCalculator.java

```
package Solver;

import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import java.util.concurrent.RecursiveTask;

import baseUtils.GeneralTime;
import baseUtils.Genome;
import baseUtils.Machine;
import baseUtils.Operation;
import baseUtils.Source;
```

```
public class FitnessCalculator extends RecursiveTask<Double> {

	/**
	 *
	 */
	Genome CurrGen;
	public int TimeType=GeneralTime._HOUR;

	double TotalProcessCompleteTime,	AllOperationTardiness ,
	AvgOperationsTardiness,
		FlowTime,
	IdleTimeTotal,					IdleCost,
				TotalOutSource,
	SetupTime,
		SetupCost,
	OperationCost,
		TotalOperationCost,			DiffJobEndDelivery,
			Earliness,
		ScheduledPercentage,			TotalMaxEarliness,

		TotalFitness;
	double FactorTotalProcessCompleteTime=0,
	FactorAllOperationTardiness=0 ,
	FactorAvgOperationsTardiness=0,			FactorFlowTime=0,
				FactorIdleTimeTotal=0,

		FactorIdleCost=0,
	FactorTotalOutSource=0,
	FactorSetupTime=0,
		FactorSetupCost=0,
	FactorOperationCost=0,
		FactorTotalOperationCost=0,
	FactorDiffJobEndDelivery=0,
	FactorEarliness=0;



	private static final long serialVersionUID = -5838820003052778O6L;

	public FitnessCalculator(Genome currGen) {
		super();
		CurrGen = currGen;
	}


	@Override
	protected Double compute() {

		return this.CalcFitness();

	}

	public Double CalcFitness()
	{
		TotalFitness=0;
```

```
            this.CalcTotalProcessCompleteTime();
            this.CalcAllOperationTardiness();
            this.CalcIdle();
            this.CalcTotalOutSource();
            this.CalcOprCost();
            this.CalcFlowTime();
            this.CalcDiffJobEndDelivery();
            this.CalcSchPerc();


            TotalFitness=(ScheduledPercentage)* (Earliness -
5*DiffJobEndDelivery - 2*FlowTime - 2*IdleTimeTotal ) ;

            this.CurrGen.FC=this;

            return TotalFitness;  // /1000000.0;

      }

      private void CalcSchPerc()
      {
            int total=0,scheduled=0;
            ScheduledPercentage=0;


            for(Machine m:CurrGen.MachineList)
            {
                  for(Operation o:m.operations)
                  {
                        total++;
                        if(o.getIsScheduled() ||
o.getAlternativeScheduled())
                              scheduled++;

                  }

            }

            ScheduledPercentage=scheduled/total;

      }


      private void CalcDiffJobEndDelivery()
      {
            DiffJobEndDelivery=0;
            Earliness=0;
            TotalMaxEarliness=0;



            for(Machine CM:CurrGen.MachineList)
            {
                  for(Operation co:CM.operations)
                  {
                        if(!co.getActive())
                              continue;
```

```
                        ArrayList<Integer>
sources=CurrGen.getOperationSource(co.getMainOperationId());

                        if(sources==null) continue;

                        for(int s_index:sources)
                        {
                              Source
sc=CurrGen.getOperationSources().get(s_index);


      if(sc.getOperationId()!=co.getMainOperationId() ||
sc.getSourceType()!=Source._Order)
                                    continue;

                              Calendar timeCal=Calendar.getInstance();
                              timeCal.setTime(CurrGen.getScheduleDate());
                              timeCal.add(Calendar.SECOND,(int)
co.TotalOperationtime );


                              TotalMaxEarliness+= ( (double) (
(sc.getDeadLine().compareTo(timeCal.getTime()) )/1000.0*3600.0    ));



                              if(sc.getDeadLine().before(co.PlanEndDate))
                              {

      co.setDiffJobEndDelivery(GeneralTime.getTimeIn(((double)(co.PlanEndDat
e.getTime()-  sc.getDeadLine().getTime())/1000),TimeType));

      DiffJobEndDelivery+=co.getDiffJobEndDelivery();

                              }
                              else
                              {

      co.setEarliness(GeneralTime.getTimeIn(((double)(sc.getDeadLine().getTi
me() - co.PlanEndDate.getTime()  )/1000),TimeType));

                                    Earliness+=co.getEarliness();
                              }

                        }
                  }



            }


      TotalMaxEarliness=GeneralTime.getTimeIn(TotalMaxEarliness,TimeType );
            //Earliness=GeneralTime.getTimeIn(Earliness,TimeType );
      }
```

```
	private void CalcOprCost()
	{
		SetupCost=0;
		OperationCost=0;
		TotalOperationCost=0;
		SetupTime=0;


		for(Machine CM:CurrGen.MachineList)
		{
			if(CM.getMachineType()!=Machine.Type_Normal)
				continue;


			for(Operation OP:CM.operations)
			{
				if(!OP.getActive())
					continue;


				SetupCost+= OP.getSetupSec()*OP.getSetupCost();

				double
OprCost=OP.getMachineCost()*OP.getMachineTime().GetTimeInSecond();

	//OprCost+=OP.getSetupTime().GetTimeInSecond()*OP.getSetupCost();

				OperationCost+=OprCost;
				SetupTime+=OP.getSetupSec();
			}

		}

		SetupTime=GeneralTime.getTimeIn(SetupTime, TimeType);
		TotalOperationCost=SetupCost+OperationCost;

	}


	private void CalcTotalOutSource()
	{
		TotalOutSource=0;

		for(Machine CM:CurrGen.MachineList)
		{
			if(CM.getMachineType()==Machine.Type_Normal)
				continue;


			for(Operation OP:CM.operations)
			{
				if(!OP.getActive())
					continue;

				TotalOutSource+=
OP.getOutSourceCost()*OP.getOutSourceQuantity();
			}
```

```
		}
	}


	private void CalcIdle()
	{

		IdleTimeTotal=0;
		IdleCost=0;


		for(Machine CM:CurrGen.MachineList)
		{

			Date LastEnd = CurrGen.getScheduleDate();

			for(Operation OP:CM.operations)
			{
				if(!OP.getActive())
					continue;



				long idletime=(OP.PlanStartDate.getTime() -
LastEnd.getTime() )/1000;
				IdleTimeTotal+=idletime;
				IdleCost+= idletime*CM.getIdleCost();
				LastEnd=OP.PlanEndDate;

			}


		}

		IdleTimeTotal=GeneralTime.getTimeIn(IdleTimeTotal,TimeType );
	}


	private void CalcAllOperationTardiness() {

		AllOperationTardiness=0;
		AvgOperationsTardiness=0;


		for(Machine cm:CurrGen.MachineList)
		{

			for(Operation co:cm.operations)
			{
				if(!co.getActive())
					continue;


	if(co.PlanStartDate.after(co.getFirstSchedulableDate()))
				{
```

```
                                    long Diff= (co.PlanStartDate.getTime() -
co.getFirstSchedulableDate().getTime())/1000;

                                    AllOperationTardiness+=Diff;
                              }

                        }


                  }


            AllOperationTardiness=GeneralTime.getTimeIn(AllOperationTardiness,Time
Type );


            AvgOperationsTardiness=AllOperationTardiness/TotalProcessCompleteTime;


            }


            private void CalcTotalProcessCompleteTime()
            {
                  TotalProcessCompleteTime=0;

                  for(Machine CM:CurrGen.MachineList)
                  {
                        for(Operation OP:CM.operations)
                        {
                              if(!OP.getActive())
                                    continue;


                              TotalProcessCompleteTime+=
(OP.PlanEndDate.getTime()-OP.PlanStartDate.getTime())/1000;
                        }


                  }


            TotalProcessCompleteTime=GeneralTime.getTimeIn(TotalProcessCompleteTim
e,TimeType );
            }


            private void CalcFlowTime()
            {
                  Date MinStart , MaxEnd;
                  MinStart=CurrGen.getScheduleDate();
                  MaxEnd=CurrGen.getScheduleDate();



                  FlowTime=0;
```

```
		for(Machine CM:CurrGen.MachineList)
		{
			for(Operation OP:CM.operations)
			{
				if(!OP.getActive())
					continue;

				if(MinStart.after(OP.PlanStartDate) )
					MinStart=(Date) OP.PlanStartDate.clone();

				if(MaxEnd.before(OP.PlanEndDate))
					MaxEnd=(Date)OP.PlanEndDate.clone();

			}


		}

		FlowTime= (MaxEnd.getTime()-MinStart.getTime())/1000;


		FlowTime=GeneralTime.getTimeIn(FlowTime,TimeType );
	}

	public String toString()
	{
		String retval="";

		retval+="\n TotalProcessCompleteTime:
"+TotalProcessCompleteTime;
		retval+=" | AllOperationTardiness: "+AllOperationTardiness;
		retval+=" | AvgOperationsTardiness: "+AvgOperationsTardiness;
		retval+=" | FlowTime: "+FlowTime;
		retval+=" | IdleTimeTotal: "+IdleTimeTotal;
		retval+=" | IdleCost: "+IdleCost;
		retval+=" | TotalOutSource: "+TotalOutSource;
		retval+=" | SetupTime: "+SetupTime;
		retval+=" | SetupCost: "+SetupCost;
		retval+=" | OperationCost: "+OperationCost;
		retval+=" | TotalOperationCost: "+TotalOperationCost;
		retval+=" | DiffJobEndDelivery: "+DiffJobEndDelivery;
		retval+=" | Earliness: "+Earliness;
		retval+=" | ScheduledPercentage: "+ScheduledPercentage;
		retval+=" | TotalMaxEarliness: "+TotalMaxEarliness;

		return retval;
	}


}
```

FeromonUpdater.java

```java
package Solver;

import java.util.List;
import java.util.concurrent.RecursiveTask;

import baseUtils.FeromonMatrix;
import baseUtils.Machine;
import baseUtils.Operation;


public class FeromonUpdater extends RecursiveTask<FeromonMatrix>{

	/**
	 *
	 */
	private static final long serialVersionUID = -3471313862237293303L;

	List<Operation> Operations;
	FeromonMatrix Feromon;
	double Factor,rho;
	boolean Updrho;

	public FeromonUpdater(List<Operation> operations,FeromonMatrix
feromon,double factor,double p_rho) {
		super();
		Operations=operations;
		Feromon=feromon;

		Factor=factor;
		rho=p_rho;
		Updrho=false;
	}
	public FeromonUpdater(Machine p_machine,double factor,double p_rho) {
		super();
		Operations=p_machine.operations;
		Feromon=p_machine.feromon;

		Factor=factor;
		rho=p_rho;
		Updrho=true;
	}

	@Override
	protected FeromonMatrix compute() {
		// TODO Auto-generated method stub
		try {
			return this.Calc();
		} catch (Exception e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
			return null;
		}
	}

	public FeromonMatrix Calc() throws Exception
	{
		int lastOprSeq=0;
```

```
		// update first operation

		boolean isFirst=true;


		for(int i=0;i<Operations.size();i++)
		{
			if(Operations.get(i).IsScheduled==false)
			{
				//lastOprSeq=i;
				continue;
			}

			if(isFirst)
			{

	Feromon.SetFeromonPlus(Operations.get(i).getOperationId(),Operations.g
et(i).getOperationId() , Factor);
				isFirst=false;
				lastOprSeq=i;
				continue;
			}


	Feromon.SetFeromonPlus(Operations.get(lastOprSeq).getOperationId(),Ope
rations.get(i).getOperationId() , Factor);
			lastOprSeq=i;
		}

		if(Updrho)
			this.Evaporation();


		return Feromon;
	}


	public void Evaporation()
	{
		Feromon.matrix.UpdateAll(rho);
	}

}
```

CrossOver.java

```
package Solver;

import java.util.Random;
import java.util.concurrent.RecursiveTask;

import baseUtils.Genome;

public class CrossOver extends RecursiveTask<Genome> {
```

```
/**
 *
 */
private static final long serialVersionUID = 6654597963076227009L;

Genome Genome1;
Genome Genome2;

public CrossOver(Genome p_Genome1,Genome p_Genome2)
{
	this.Genome1=p_Genome1;
	this.Genome2=p_Genome2;

}


protected Genome compute() {
	// TODO Auto-generated method stub
	return this.CrossOverCompute();

}

public Genome CrossOverCompute()
{
	Random rand = new Random();
	try {

		if(this.Genome1!=null && this.Genome2!=null)
		{
			Genome child=(Genome)Genome1.clone();
			child.setFitnessValue(0.0);

			for(int i=0;i<Genome1.MachineList.length;i++)
			{
				if(rand.nextBoolean())
				{


child.MachineList[i]=Genome1.MachineList[i].clone();

				}
				else
				{

child.MachineList[i]=Genome2.MachineList[i].clone();
				}
			}


			return child;
		}

	} catch (CloneNotSupportedException e) {
		// TODO Auto-generated catch block
		e.printStackTrace();
	}
	return null;
```

```
	}
```

AntMutation.java

```
package Solver;

import java.util.Calendar;
import java.util.Collections;
import java.util.Date;
import java.util.Random;
import java.util.concurrent.RecursiveTask;

import baseUtils.Machine;

public class AntMutation extends RecursiveTask<Machine> {

	/**
	 *
	 */

	private static final long serialVersionUID = -7942436616596160057L;

	Machine MutationMachine;
	double MutationRate;
	Random rand;
	double alpha,beta;
	public Date SchStartDate;

	public AntMutation(Machine p_pach)
	{
		this.MutationMachine=p_pach;
		this.MutationRate=p_pach.MutationRate;
		rand=new Random();
		alpha=3;
		beta=2;
	}

	@Override
	protected Machine compute() {

		return this.MutationCompute();
	}

	public Machine MutationCompute() {

		if(MutationMachine.operations.size()<2)
			return MutationMachine;

		// set first operation
		int FromSeq=this.GetFirstOperation();
		int CurrSeq=0;
		Collections.swap(MutationMachine.operations, CurrSeq,FromSeq);
```

```
		int WillBeMutate=MutationMachine.operations.size()-1;
		// set other operations

		while(WillBeMutate>0)
		{
			CurrSeq++;
			FromSeq=this.GetNextOperation(CurrSeq);
			Collections.swap(MutationMachine.operations,
CurrSeq,FromSeq);

			WillBeMutate-=1;

		}


		return MutationMachine;
	}

	private int GetNextOperation(int pCurrSeq)
	{
		int OprCount=MutationMachine.operations.size()-pCurrSeq;
		if(OprCount==1)
			return pCurrSeq;

		double[] probs=new double[OprCount];
		double[] taueta=new double[OprCount];
		double total=0.0;

		int ToOpr=0;

		for(int i=1;i<OprCount;i++)
		{
			ToOpr=i+pCurrSeq;

			try {
				double Visibility=1;

	taueta[i]=MutationMachine.feromon.GetFeromon(MutationMachine.operation
s.get(pCurrSeq).getOperationId(),
MutationMachine.operations.get(ToOpr).getOperationId());

	Visibility=1/Math.pow((MutationMachine.operations.get(ToOpr).getLevel(
)),2);

					Calendar cal = Calendar.getInstance();
					cal.setTime(SchStartDate);
					cal.add(Calendar.MINUTE, (int)
MutationMachine.operations.get(ToOpr).TotalOperationtime);
					double x=
((MutationMachine.operations.get(ToOpr).OrderDeliveryDate.getTime() -
cal.getTime().getTime())/(3600000));
					if(x==0)
						x=1;

					Visibility*= (1/x)  ;
```

```
                        taueta[i]=Math.pow(taueta[i],alpha ) *
(Math.pow(Visibility, beta));

                        total+=taueta[i];
                  } catch (Exception e) {
                        e.printStackTrace();
                        return -1;
                  }
            }

            for(int i=0;i<OprCount;i++)
            {
                  probs[i]=taueta[i]/total;

            }
            total=0.0;

            for(int i=0;i<OprCount;i++)
            {
                  total+=probs[i];
                  probs[i]=total;
            }


            double prb=rand.nextDouble();

            for(int i=0;i<OprCount;i++)
            {
                  if( prb<probs[i+1]) // prb>probs[i] &&
                        return i+pCurrSeq;
            }

            return -1;
      }


      private int GetFirstOperation()
      {
            int OprCount=MutationMachine.operations.size();
            double[] probs=new double[OprCount];
            double[] taueta=new double[OprCount];
            double total=0.0;

            for(int i=0;i<OprCount;i++)
            {
                  try {
                        double Visibility=1;


      taueta[i]=MutationMachine.feromon.GetFeromon(MutationMachine.operation
s.get(i).getOperationId(),
MutationMachine.operations.get(i).getOperationId());

      Visibility=1/Math.pow((MutationMachine.operations.get(i).getLevel()),2
);

                        Calendar cal = Calendar.getInstance();
```

```
                        cal.setTime(SchStartDate);
                        cal.add(Calendar.MINUTE, (int)
MutationMachine.operations.get(i).TotalOperationtime);
                        double x=
((MutationMachine.operations.get(i).OrderDeliveryDate.getTime() -
cal.getTime().getTime())/3600000);
                        if(x==0)
                              x=1;

                        Visibility*= (1/x)  ;

                        taueta[i]=Math.pow(taueta[i],alpha ) *
(Math.pow(Visibility, beta));
                        total+=taueta[i];

                  } catch (Exception e) {
                        e.printStackTrace();
                        return -1;
                  }
            }

            // cumulative

            for(int i=0;i<OprCount;i++)
            {
                  probs[i]=taueta[i]/total;

            }
            // Distribute
            total=0.0;

            for(int i=0;i<OprCount;i++)
            {
                  total+=probs[i];
                  probs[i]=total;
            }



            double prb=rand.nextDouble();

            for(int i=0;i<OprCount;i++)
            {
                  if( prb<probs[i])
                        return i;
            }



            return -1;
      }


}
```

# ÖZGEÇMİŞ

**Adı Soyadı** : Celal BİLGİN

**Doğum Yeri ve Yılı** : İstanbul - 1986

**Yabancı Dili** :İngilizce

**İlk Öğretim** : Adnan Ötüken İlkokulu 1996

**Orta Öğretim** : Bahçelievler Süper Lisesi 2004

**Lisans** : Kırıkkale Üniversitesi Endüstri Mühendisliği 2008

**Yüksek Lisans** : Bahçeşehir Üniversitesi Bilgi Teknolojileri

**Enstitü Adı** :Fen Bilimleri Enstitüsü

**Program Adı** : Bilgi Teknolojileri

**Çalışma Hayatı** :

2008- IAS yazılım ve danışmanlık , Kıdemli Proje Yöneticisi