T.C.
BALIKESİR ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

# KESME ve PAKETLEME PROBLEMLERİ ve ARAŞTIRMAYA YÖNELİK BİR METOT GELİŞTİRİLMESİ ve BU METODUN ETKİNLİĞİNİN SINANMASI

YÜKSEK LİSANS TEZİ

Kadriye ERGÜN
Endüstri Mühendisi

Balıkesir, Ağustos – 2004

T.C.
BALIKESİR ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

# KESME ve PAKETLEME PROBLEMLERİ ve ARAŞTIRMAYA YÖNELİK BİR METOT GELİŞTİRİLMESİ ve BU METODUN ETKİNLİĞİNİN SINANMASI

YÜKSEK LİSANS TEZİ

Kadriye ERGÜN

Tez Danışmanı: Doç.Dr. Ramazan YAMAN

Sınav Tarihi : 01.09.2004

Jüri Üyeleri : Doç.Dr. Ramazan YAMAN (Danışman, BAÜ-MMF)
Yrd.Doç.Dr. Ziya AKSOY (BAÜ-MMF)
Yrd.Doç.Dr. Necati ÖZDEMİR (BAÜ-FEF)

Balıkesir, Eylül - 2004

## ÖZET

## KESME ve PAKETLEME PROBLEMLERİ ve ARAŞTIRMAYA YÖNELİK BİR METOT GELİŞTİRİLMESİ ve BU METODUN ETKİNLİĞİNİN SINANMASI

**Kadriye ERGÜN**
**Balıkesir Üniversitesi, Fen Bilimleri Enstitüsü,**
**Endüstri Mühendisliği Anabilim Dalı**

**(Yüksek Lisans Tezi / Tez Danışmanı : Doç. Dr. Ramazan YAMAN)**

**Balıkesir, 2004**

Yöneylem Araştırması alanında, en önemli problemlerden biri de Kesme ve Paketleme problemleridir. Kesme ve Paketleme problemleri, tasarımdan, üretimin çeşitli sınıfından, dağıtım ve satışa kadar tüm iş alanlarında görülmektedir. Kombinatoryal optimizasyon alanında yer alan bu problemler, matematiksel çözümün çok zor olduğu, NP-Hard sınıfı problemler olarak bilinirler. Makul bir zamanda bu problemleri çözebilecek, herhangi bir çözüm metodu bilinmemektedir. Bu nedenle, son zamanlarda sezgisel (heuristic) tekniklerin kullanımı ön plana çıkmıştır.

Bu çalışmanın ilk bölümünde, Kesme ve Paketleme problemlerinin, optimizasyon problemleri arasında, hangi sınıfa dahil olduğunu göstermek için, optimizasyon hakkında bilgi verilmiştir. Optimizasyon problemleri ve bunlara ilişkin çözüm metotları sınıflandırılmıştır.

İkinci bölümde, Kesme ve Paketleme problemlerinin yerleştirme problemleri ile ilişkisi açıklanmış ve bu problemlerin çözüm metotlarının, Kombinatoryal Optimizasyon Problemleri çözüm metotları arasında yer aldığı gösterilmiştir.

Son bölümde ise, düzgün dikdörtgensel parçalardan oluşan iki boyutlu Kesme ve Paketleme problemleri için sezgisel bir teknik geliştirilmiştir. Bu yaklaşımın adımları ve uygulanması kısmında oluşacak alternatifler açıklanmıştır. Çözümün geçerliliğini test etmek için, çeşitli örnekler kullanılmış ve ortaya çıkan sonuçlar karşılaştırılmıştır.



**ANAHTAR SÖZCÜKLER:** Kombinatoryal Optimizasyon / Kesme ve Paketleme Problemleri / Sezgisel

## ABSTRACT

**CUTTING and PACKING PROBLEMS and, IMPROVEMENT of A HEURISTIC METHOD and TESTING THE EFFICIENCY of THIS METHOD**

**Kadriye ERGÜN**
**Balıkesir University, Institute of Science, Department of Industrial Engineering**

**(M.Sc. Thesis / Supervisor : Assoc. Prof. Dr. Ramazan YAMAN)**

**Balıkesir-Turkey, 2004**

One of the most important problems in Operational Research are Cutting and Packing problems. Cutting and Packing problems are seen in every business areas as; in design, in some types of production, in delivery and sale. These problems, belonging to Combinatorial Optimization area, are known as NP-Hard class problems of which the mathematical solution is very hard. No solution method is known to solve these problems in a reasonable time. So using the heuristic techniques have gained importance and become popular recently.

In the first part of this study, information about optimization is given to point out the class of Cutting and Packing problems between optimization problems. Optimization problems and the solution methods relating to these are classified.

In the second part, the relation of Cutting and Packing problems with placement problems is explained and it is pointed out that, the solution methods of these problems are taking part between the solution methods of combinatorial optimization problems.

In the last part, a heuristic technique is improved for two dimensional Cutting and Packing problems forming of regular rectangular parts. The future possible alternatives are explained in the steps and application part of this approachment. Various samples are used to test the validity of the solution and the conclusions are compared.



**KEY WORDS :** Combinatorial optimization / Cutting and Packing Problems / heuristic

## İÇİNDEKİLER

## KISALTMALAR LİSTESİ

| | |
|---|---|
| TP | Tamsayılı programlama |
| TSP | Gezgin Satıcı Problemi |
| VRP | Araç Rotalama Problemi |
| P | Polinomial |
| NP | Non-deterministic polinomial |
| TB | Tavlama Benzetimi |
| TS | Tabu Arama |
| GA | Genetik Algoritma |
| BL-Algoritması | Aşağı-Sol Algoritması |
| 1D/2D/3D/4D | 1/2/3/4 Boyutlu |

* Algoritmanın geliştirilmesi ve kodlanması sırasında kullanılan kısaltmalara yer verilmemiştir.

## ŞEKİL LİSTESİ

## TABLO LİSTESİ

## ÖNSÖZ

Kesme ve Paketleme problemleri ekonomik değerlerinin fazla olması nedeniyle, bir çok araştırmacının ilgisini çekmektedir. Araştırmalar sonucu, bulunan yöntemlerin veya tekniklerin uygulanabilirliği bu ilgiyi daha da arttırmaktadır.

Çalışmaya başlamadan önce bu kadar ilgi gören, bu konu hakkında fikrim bulunmamaktaydı. Beni bu konu ile tanıştıran ve her aşamada sürekli destek veren, kendisinden çok şey öğrendiğim ve öğreneceğim, danışman hocam Doç. Dr. Ramazan YAMAN'a teşekkür ederim.

Bu çalışmada ve diğer zamanlarda karşılaştığım her problemi sabırla dinleyip, çözümleri aramamda her zaman yardımcı olan değerli hocam, Öğr. Gör. Emine UÇMUŞ'a teşekkür ederim.

Aynı sıkıntıları paylaştığım, arkadaşlarıma yardımlarından dolayı teşekkür ederim.

Son olarak, iyi dileklerini hiçbir zaman eksik etmeyen ve her konuda beni destekleyen sevgili aileme teşekkürlerimi sunarım.

**Balıkesir, 2004** **Kadriye ERGÜN**

# 1. GİRİŞ

İnsanların, "daha iyi nasıl olabilir ya da nasıl elde edilebilir?", sorusuna cevap aramaları, teknolojinin gelişmesini sağlayan en önemli etken olmuştur. Gerçek hayatı daha kolay hale getirebilen teknolojinin gelişimi, aynı zamanda yeni problemler de türetmektedir. Problemlerin sayısının ve çeşitlerinin artması, karar vericilerin işini zorlaştırmıştır. Çünkü bilinen çözüm teknikleri, problemlerin yeni durumları için yeterli olamamaktadır. Bundan dolayı, yöneticilerin, karşılaştıkları problemlerde karar vermelerine yardımcı olmak amacıyla kullanabilecekleri bilimsel problem çözme yaklaşımı olarak Yöneylem Araştırması adlı bilim ortaya çıkmıştır. Yöneylem Araştırma'nın konusu, karşılaşılan sorunları belirlemek ve karar problemlerinin en iyi çözümünü aramaktır. İlk başta da belirtildiği gibi, gelişimin sebebi en iyiyi aramak olmuştur. Yöneylem Araştırması içindeki en iyileme kavramı, optimizasyon olarak ifade edilmektedir.

Optimizasyon kavramı ve problemleri ile, günlük hayatımız içinde bile farkında olmadan karşı karşıya kalırız. Örneğin seyahat edecek bir kişinin, valizini hazırlarken, hacim, ağırlık ve eşyaların şekilleri gibi kısıtları düşünerek bir yerleşim düzeni belirlemesi aslında bir optimizasyon problemidir. Burada amaç, kişinin taşıyabileceği ağırlıkta ve tüm eşyalarını alabileceği bir valiz yerleşim düzeni seçmektir.

Optimizasyon problemleri genellikle ifade etmesi kolay fakat çözümü zor olan problemlerdir. Matematiksel anlamı ise, amaç fonksiyonunu belirli kısıtlar altında minimum veya maksimum yapmaktır.

Optimizasyon problemleri, temel olarak, karar değişkenlerinin yapısına göre, iki sınıfa ayrılmıştır. Bunlar Kesikli Optimizasyon ve Sürekli Optimizasyondur. Kesikli optimizasyon problemleri, Kombinatoryal Optimizasyon Problemleri olarak da isimlendirilebilirler.

Kesme ve paketleme problemleri, Kombinatoryal Optimizasyon Problemleri sınıfında yer almaktadır. Bu problemler, ağaç, cam, metal, tekstil sanayi, kargo ve lojistik gibi birçok alanda görülmektedir. Bu konudaki ilk yayınlar, Gilmore ve

Gomory tarafından 1961 ve 1965'te yapılmıştır. Daha sonra, literatürde yer alan çalışmalar, problemlerin ait oldukları sınıflar ile birlikte Tablo 1.1'de belirtilmiştir. (2001'de yapılan çalışmaya göre). Kesme ve paketleme problemlerinin sınıflandırılması konusunda en çok kabul gören çalışma 1990'da Dyckhoff tarafından yapılmış, **Typology of Cutting and Packing Problems**, adlı makalede yer almıştır.

Tablo 1.1 Kesme ve Paketleme problemlerine ait literatürde yer alan çalışmalar [1]

| Yazarlar | Konular | Problem sınıfı |
|---|---|---|
| Dyckhoff ve Finke (1992) | Çeşitli büyüklükteki problemlerin analizi | Dyckhoff sınıflandırması |
| Sweeney ve Paternoster (1992) | 400'den daha fazla problem içeren kitap, tezler ve makale çalışmaları | Boyutlar, çözüm metotları, özel konular |
| Golden (1976) | 2D kesme stok problemleri | Çözüm metodolojisi |
| Hinxman (1980) | 2D kesme kaybı ve çeşitli problemler | Boyutlar, çözüm metodolojisi, |
| Rayward-Smith ve Shing (1983) | 1D ve 2D kutu paketleme | Boyutlar |
| Sarin (1983) | 2D kesme stok problemleri | Çözüm metodolojisi |
| Coffman ve diğerleri (1984) | Kutu paketleme | Kutu paketleme tipi, boyutlar |
| Dowsland (1985) | 2D ve 3D dikdörtgensel problemler | Problem tipi, boyutlar |
| Coffman ve Shor (1990) | 2D düzgün paketleme problemleri | Açık, kapalı; olasılıklı analiz |
| Haessler ve Sweeney (1991) | 1D ve 2D kesme stok problemleri | Boyutlar, çözüm metodolojisi |
| Dowsland (1991) | 3D problemler | Çözüm metodolojisi |
| Dowsland ve Dowsland (1992) | 2D ve 3D paketleme problemleri, başlıca düzgün şekiller | Problem tipi, boyutlar |
| Whelan ve Batchelor (1993) | 2D düzgün olmayan paketleme problemleri için otomatik paketleme sistemlerinin endüstriyel uygulamaları | Uygulama, deri endüstrisinde odaklanma |
| Dowsland ve Dowsland (1995) | 2D, düzgün olmayan paketleme problemleri | Gruplar için metotlar, paketleme, hesaplama geometrisi |
| Hopper ve Turton (1997) | 2D ve 3D, düzgün ve düzgün olmayan paketleme problemleri ve genetik algoritmalar | Parçaların geometrik karakteristikleri, boyutlar |

Endüstride birçok alanda görülen Kesme ve Paketleme Problemleri, genellikle NP-Hard sınıfı problemler içinde yer almaktadırlar. Çözüm zamanı fonksiyonu, üstel olarak artış göstermektedir ve kabul edilebilir bir zamanda bu problemleri birebir çözebilecek algoritmalar bulunmamaktadır. Bu durum tüm alternatifleri deneyen kesin yöntemler yerine, sezgisel tekniklerin kullanımını daha geçerli hale getirmiştir.

Parçaların geometrik şekilleri ve boyut özellikleri bu problemlerin çözümünde etkili kriterler arasında yer almaktadır. En basit haliyle, bir büyük parçanın üzerine, küçük parçaların en iyi yerleşimi olarak tanımlanabilen Kesme ve Paketleme Problemleri, küçük parçaların düzgün olmadığı durumda Nesting problemler olarak tanımlanmaktadırlar.

Bu çalışmada iki boyutlu düzgün dikdörtgensel parçalardan oluşan, Kesme ve Paketleme Problemleri için sezgisel bir algoritma geliştirilmiştir. Algoritmanın temel aşamaları Matlab 6.5 programıyla kodlanmıştır. Algoritmanın performansını değerlendirmek üzere çeşitli örnekler üzerinde denemeler yapılarak, ortaya çıkan sonuçlar karşılaştırılmıştır.

## 2. OPTİMİZASYON

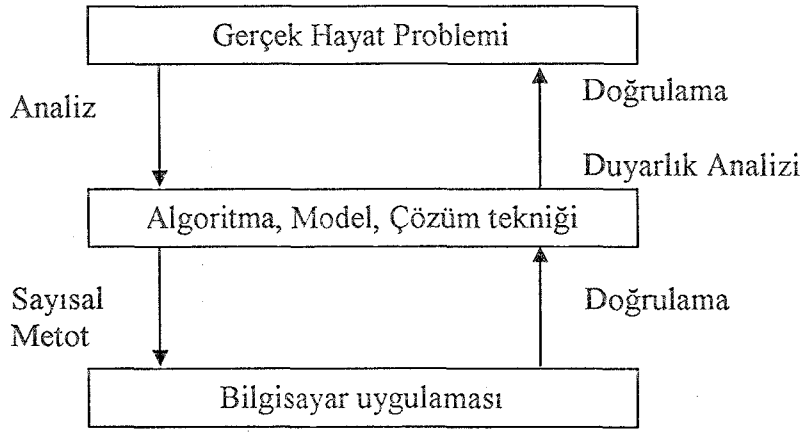
Günlük hayat içinde alternatifler içinden seçim yapmak, yani karar almak insanların olduğu gibi işletmelerin de devamlı olarak karşı karşıya geldikleri bir durumdur. İşletmelerin çözüm getirmek zorunda oldukları karar problemleri, esas itibariyle, kıt kaynakların rakip faaliyetler arasında optimum (en iyi) bir şekilde dağıtılması problemidir [2]. Karar verilmesi gereken alternatif veya çözülmesi gereken problem ne olursa olsun, amaç, optimum, yani en iyi sonuca ulaşmaktır. Buradan hareketle, en genel anlamda optimum sonucun elde edilmesi işlemine "Optimizasyon" denilmektedir. Bir bilgisayar terimi olarak optimizasyon şu şekilde ifade edilebilir.

**Algoritma:** Belirli bir problemin çözümünde izlenen yöntemdir.

**Kod:** Algoritmanın bir bilgisayar dili ile ifadesidir.

**Optimizasyon:** Bir algoritma veya kodun daha hızlı veya daha az yer kaplayacak şekilde düzenlenmesidir [3].

Günümüzde iş hayatında, endüstride, yönetimde, mühendislikte ve bilgisayar biliminde kullanılan optimizasyon işlemi Şekil 1.1'deki gibi özetlenebilir.




Şekil 1.1 Optimizasyon İşlemi [4]

Gerçek problemde algoritma, model veya çözüm tekniğine geçiş aşamasına *analiz* denir. Analiz aşamasında, problemle ilgisi olmayan detaylar çıkartılır ve önemli olan elemanlar üzerinde yoğunlaşılır. Birçok durumda bu aşama kendi başına önemli bir yer tutar. Kendisini takip eden optimizasyon işlemi yapılmasa bile, bu aşama bir anlam ifade edebilir [4].

Analiz sonucunda, bilgi edinme, kişilerle temasa geçme, uzman bulma, çözüm araçlarını belirleme, problemin anlaşılması ve ifade edilmesi gerçekleşmiş olur. Algoritma, model veya çözüm tekniğinden, bilgisayar uygulamasına geçiş, sayısal metotla olur. Bu aşama hesaplamaların doğruluğu, verimli uygulamaların yapılıp yapılmadığı gibi kısımların tartışıldığı ya da değerlendirildiği aşamadır. Ancak bu aşama doğru optimizasyon paketlerini seçmiş olan problemlerde, yalnızca sonuçlar dikkate alınıyorsa, önemli olmayabilir [4].

Doğrulama ve duyarlık analizi gerçek probleme dönüşte en önemli aşamayı göstermektedir.

Sonuçların karşılaştırılması ve hassasiyetlerin değerlendirilmesi için gerçekte uygulama sonucunda ne gibi durumların ortaya çıkabileceği düşünülmelidir. Bunların doğruluğu ya da hassasiyeti problem çözümü için varsa başka yollarla kıyaslanmalıdır.

Duyarlık analizinde ise sonucun veriler karşısında nasıl değiştiği durumunun, değerlendirilmesi yapılır. Örneğin çalışan kişi sayısı ile yapılan üretim arasında çok az bir değişim varsa bu faktör dikkate alınmamalı veya bir hatanın oluşup oluşmadığı kontrol edilmelidir [4].

Optimizasyon teknikleri, karar verme, planlama, kaynak kullanımı, çizelgeleme gibi durumlar için günlük hayatta önemli rol oynarlar. Örneğin uluslararası bir petrol şirketi ham petrolün değişik kaynaklardan alınması konusunda, taşıma, fiyat, ürün kalitesi gibi parametrelere bakarak karar oluşturur. Burada amaç; maksimum kazanç sağlayacak karar modelinin oluşturulmasıdır. Bir başka örnek ise bir havayolu

şirketinin uçuş planlama, personel sağlama, bakım yapma gibi konuları dikkate alarak minimum maliyetli optimizasyon modelinin kullanmasıdır [4].

Büyük ölçekli optimizasyon teknikleri daha çok II. Dünya Savaşı sırasında orduların yönlendirilmesi ve bunların gerek duyduğu lojistik desteğin sağlanması konusunda geliştirilmiş ve kullanılmıştır. Bu süreç esnasında etkinliği arttırıcı her türlü teknik ve yöntem değerlendirilmiş, kısıtlı insan kaynakları ve diğer kaynaklar etkin şekilde kullanılmaya başlanmıştır.

Sınırlı kaynakları kullanarak, amaçlanan fonksiyon ve/veya fonksiyonlar için en iyi sonucun elde edilmesini hedefleyen optimizasyon kavramının matematiksel anlamı aşağıdaki gibi ifade edilmektedir [4].

Belli bir amaç fonksiyonunu (objective function), eldeki değişkenleri kullanarak minimize ya da maximize etmektir. Tüm bunlar gerçekleşirken, çeşitli kısıtlamalar (constraint) söz konusu olabilir. Bu kısıtlamalar iki türlüdür:

- Eşitlik kısıtlamaları (equality constraints), ki bunlara her zaman uyulmak zorundadır.
- Eşitsizlik kısıtlamaları (inequality constraints), bunlar genelde aşılmaması gereken limitleri belirtir.

Genel olarak optimizasyon problemlerinin matematiksel gösterimi şu şekildedir [5];

$$\text{(max./min.)} \quad f(x)$$

$$\text{Kısıtlar} \qquad c_i(x) = 0\,,\ i=1, 2, \ldots, m' \tag{2.1}$$

$$c_i(x) \geq 0\,,\ i=m'+1, \ldots, \text{m.} \tag{2.2}$$

$f(x)$ amaç fonksiyonunu, $c_i(x)$ değerleri kısıtları göstermektedir.

Amaç, mevcut bir takım algoritmaları kullanarak yukarıda tanımlanan problemi çözmektir.

## 2.1 Optimizasyon Problemlerinin Sınıflandırılması

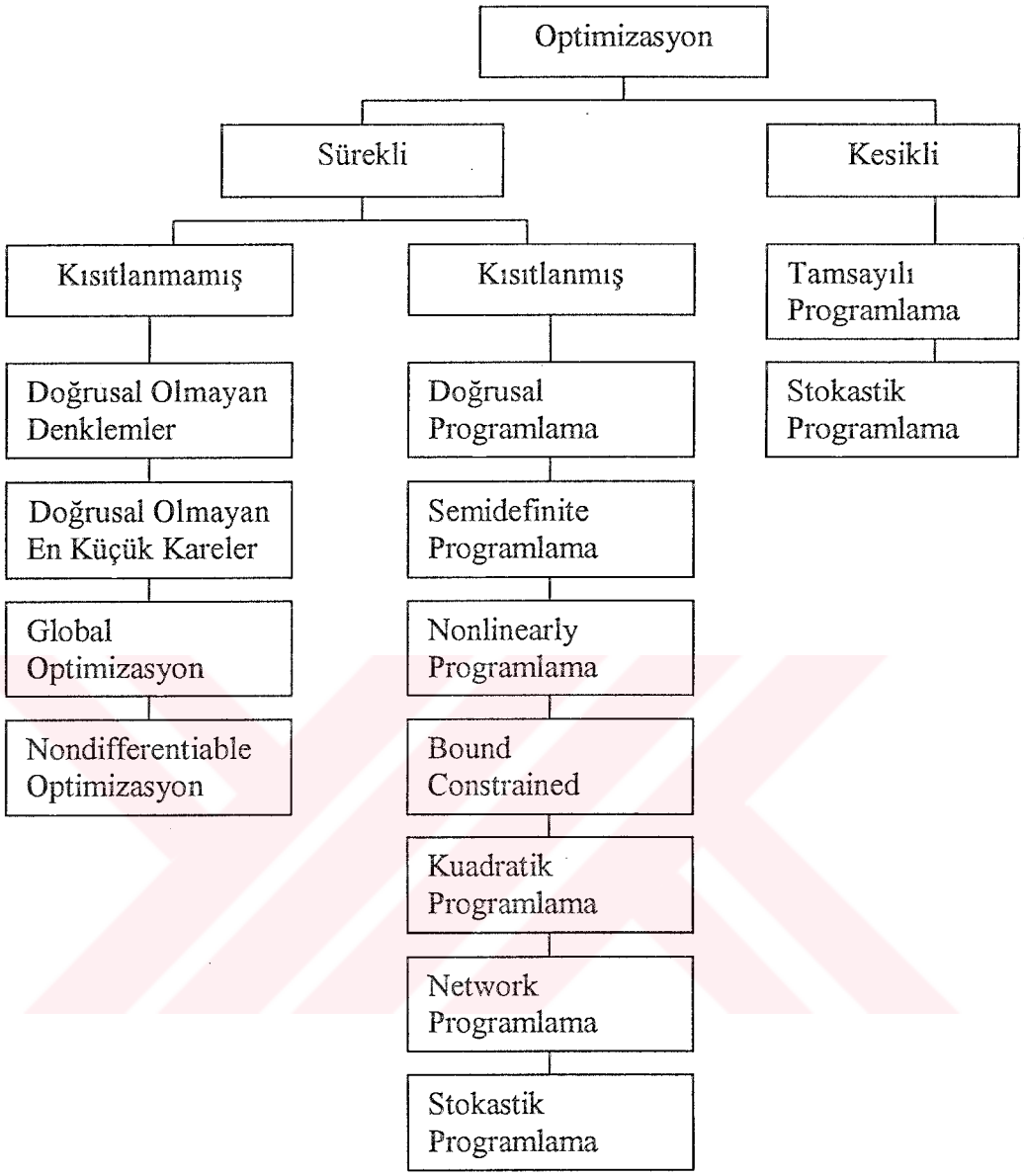
Optimizasyon problemlerinin pek çok çeşidi vardır. Temel olarak optimizasyon problemleri, karar değişkenlerinin sürekli veya kesikli olmalarına göre ikiye ayrılır. Karar değişkenleri kesikli veya süreksiz olanlara **K**ombinatoryal (birleşi) **O**ptimizasyon **P**roblemleri denir.

Bu sınıflandırma çeşitleri arttırılabilir. Amaç fonksiyonu ve kısıtların özelliklerine göre sınıflandırılması konusunda en son durum, her bir problemin ayrıldığı kategoriye yerleştirilmesiyle oluşmuştur [5].

Tablo 2.1 Optimizasyon problemlerinin amaç fonksiyonu ve kısıtlara göre sınıflandırılması [5]

| **Amaç $\{f(x)\}$ fonksiyonun özellikleri** | **Kısıtların $\{c_i(x)\}$ özellikleri** |
|---|---|
| Tek değişkenli bir fonksiyon | Kısıt yok |
| Lineer fonksiyon | Basit sınırlar |
| Lineer fonksiyonun kareleri toplamı | Lineer fonksiyon |
| Quadratic fonksiyon | Dağınık (Sparse) Lineer fonksiyon |
| Lineer Olmayan Fonksiyonların karelerinin toplamı | Düzgün Lineer Olmayan Fonksiyon |
| Düzgün (smooth) Lineer Olmayan Fonksiyon | Dağınık Lineer Olmayan Fonksiyon |
| Dağınık Lineer Olmayan Fonksiyon | Düzgün Olmayan Nonlinear Fonksiyon |
| Düzgün Olmayan Nonlinear Fonksiyon | |

Optimizasyon problemlerinin amaç fonksiyonu ve kısıtların oluşturduğu fonksiyonların gruplandırılmasının dışında en genel sınıflandırılması Şekil 2.1'de gösterilmiştir.




Şekil 2.1 Optimizasyonun en temel sınıflandırılması [6]

Bu sınıflandırma içinde en yaygın olarak kullanılanlardan biri Doğrusal Programlamadır.

**D**oğrusal **P**rogramlama modelinin temel varsayımlarından birisi tüm değişkenlerin sürekli olması ve karar değişkenlerinin değerlerinin tam sayı ve kesirli olmasıdır. Bazı gerçek hayat problemlerinde tamsayı değerli olmayan çözüm değişkenlerinin anlamı yoktur. Girdi ve çıktıların bölünmezlik sorunu karar

değişkenlerinin tamsayı değerli olmasını gerektirir. Sermaye bütçelemesi, elektrik jeneratör birimleri, araç ve gereçler, makinalar, kişiler bunlara birer örnektir. Herhangi bir problemin doğrusal programlama ile elde edilen optimal çözüm değerleri tamsayı değilse ve de çözüm değerlerinin tamsayı olması istendiğinde, yönetici veya kullanıcı diğer bir çözüm tekniği olan tamsayılı programlamaya başvurmalıdır. Doğrusal programlama problemlerinin tamsayılı çözümünün elde edilmesinde kullanılan algoritma tamsayılı programlamadır [7]. Bu nedenle Tamsayılı Programlama, optimizasyonun genel sınıflandırılması içinde Kesikli Optimizasyon içinde yer almaktadır. Tamsayılı doğrusal programlama modelinin değişkenleri kesikli olduğu gibi bu değerlerden bazıları veya tümü tamsayı değerlidir. Bu hali ile tamsayılı programlama yani değişkenlerin bazıları veya tümü tamsayı olması gereken bir doğrusal programlamadır. Genel olarak tamsayılı programlama modeli aşağıdaki şekilde ifade edilebilir.

$$\text{Max.} \quad Z = \sum_{j=1}^{n} c_j x_j \tag{2.3}$$

$$\text{Kısıtlayıcılar} \sum_{j=1}^{n} a_{ij} x_j \leq b \quad \text{i=1, 2, ...,m} \tag{2.4}$$

ve

$$x_j = 0, 1, 2, \ldots \text{ tamsayı (j=1, 2, ..., n)} \tag{2.5}$$

Doğrusal programlama modeli ile tamsayılı programlama modeli arasındaki tek ayrıcalık, doğrusal programlamadaki pozitif olma koşulunun ($x_j \geq 0$) tamsayı olma koşuluna yani $x_j$=0, 1, 2, 3, 4, 5, ..., n şekline dönüşmesidir. Tamsayılı programlamada tüm $x_j$ değişkenlerinin değerleri sıfıra eşit ve sıfırdan büyük tamsayıdır [7].

Doğrusal programlama problemlerinin tamsayılı çözümlerini sağlayacak hesaplama yöntemi ilk kez 1958 yılında Gomory tarafından geliştirilmiştir. Gomory'nin geliştirdiği hesaplama yöntemine Tamsayılı Algorithm veya Kesme Düzlemi Yöntemi adı verilmiştir. Bu yöntem sonlu sayıda işlemden sonra optimal bir tamsayılı çözümü sağlar [7].

Optimizasyon problemlerinin çeşitliliği nedeniyle bu tür problemlerin çözümünde etkili olabilecek yazılımlar geliştirilmiştir. Bunlardan bazıları, uygulandıkları problem çeşidi ile birlikte aşağıda verilmiştir [8].

| | |
|---|---|
| Ampl | Modelleme |
| BPMPD | Lineer programlama |
| BQPD | Kuadratik Programlama |
| BT | Minimizasyon |
| CO | Kısıtlanmış optimizasyon |
| CONOPT | Non-lineer Programlama |
| FortMP | Lineer ve Tam Sayılı Programlama |
| LINDO | Lineer, Tam Sayılı ve KuadratikProgramlama |
| M1QQ3 | Kısıtlanmamış optimizasyon |
| MATLAB | Optimizasyon |

Kesikli optimizasyon grubu içinde yer alan kombinatoryal problemlerin, doğrusal programlama ile yakın ilişkisi vardır ve bu tür problemlerin çözümü için ilk girişimler doğrusal programlama metotlarının kullanılması ile başlamıştır. Örneğin, tamsayılı programlamayı oluşturmak amacıyla genellikle 0-1 değerlerini alan tamsayılı değişkenler tanımlanmıştır. Örneğin, 0-1 sırt çantası probleminde değişken tanımlaması aşağıdaki gibi yapılmaktadır [9].

$$x = \begin{cases} 1, & \text{i parçası paketlenirse} \\ 0, & \text{i parçası paketlenmezse} \end{cases}$$

Bu problem daha sonra aşağıdaki tamsayılı programa dönüştürülmektedir:

$$\text{Max} \sum_{i=1}^{n} x_i v_i \text{ , buradan} \tag{2.6}$$

$$\sum_{i=1}^{n} x_i c_i \leq C \tag{2.7}$$

İzleyen bölümde belirtildiği gibi, n parça sayısını, C sırt çantasının kapasitesini, göstermekte ve $i$. parça $v_i$ değerine sahip olup kapasitenin $c_i$ birimini kullanmaktadır [9].

Ancak çoğu Kombinatoryal Optimizasyon probleminin bir Tamsayılı Programlama (TP) modelinin kurulması zordur. Bunun yanı sıra, bu tür problemlerin TP modelinin kurulması için oldukça fazla sayıda değişken ve kısıt tanımlamalarının yapılması gerekebilir. Bundan dolayı genel amaçlı TP bilgisayar programları, böyle büyük boyutlu problemleri çözememektedir. TP problemleri, Doğrusal Programlama problemlerinden daha zor olmaktadır [10].

Kombinatoryal problemlerin optimum çözümlerinin bulunmasında TP istenen başarıya ulaşmasa da, çeşitli faydaları olmaktadır. Öncelikle, problem yapısının tam olarak tanımlanmasında ve anlaşılmasında TP modelinin önemi büyüktür. Ayrıca problemin optimum çözümünün bir alt sınırının elde edilmesi amacıyla Lagrangean relaxation tekniğinin kullanılması için TP'den yararlanılmaktadır. Kombinatoryal problemlerin yer aldığı Kombinatoryal Optimizasyon izleyen bölümde açıklanmaktadır [10].

## 2.2 Kombinatoryal Optimizasyon

Çoğu problemler, karar değişkenlerinin bir fonksiyonu olan ve bazı kısıtlardan oluşan problemlerdir. Bu problemler genel olarak aşağıdaki gibi formüle edilirler.

$$\min f(x) \tag{2.8}$$

$$\text{Kısıtlar } g_i(x) \geq b_i;\ i=1, \ldots, m \tag{2.9}$$

Burada $x$, karar değişkenlerinin bir vektörünü, $f(x)$ ve $g_i(x)$ genel bir fonksiyonu göstermektedir. Bu formulasyondaki minimizasyon problemi, gerekli değişikliklerin yapılması sonucu maksimizasyon formuna da dönüştürülebilmektedir. Bu problemlerin pek çok spesifik sınıfı bulunmaktadır. Söz konusu problem sınıfları, kısıtların değiştirilmesine göre ve/veya karar değişkenlerinin alabileceği değerlere göre elde edilmektedir. Bu sınıflar içerisinde en yaygın olarak bileneni $f(x)$ ve $g_i(x)$'lerin karar değişkenlerinin doğrusal bir fonksiyonu olarak dikkate alınması ve

karar değişkenlerinin de sürekli (kesirli, reel) değerler almasıyla ortaya çıkan doğrusal programlama problemleridir [9,10].

Kombinatoryal terimi, karar değişkenlerinin kesikli olmasını yani problem çözümünün tamsayıların ya da diğer kesikli nesnelerin bir kümesi veya bir sırası olmasını ifade etmektedir. Bu sınıftaki problemler için optimum çözümlerin bulunması kombinatoryal optimizasyon olarak bilinmektedir [10].

Son yıllarda kombinatoryal optimizasyon problemlerine giderek artan bir ilgi vardır. Bir çok problemin optimum çözümünü verecek yöntemin bilinmemesi araştırmacıları bu konuda daha da ilgili hale getirmektedir. Bu durum Gezgin Satıcı Problemi (**T**raveling Salesman **P**roblem-TSP) ile açıklanabilir [10].

TSP, ifade edilmesi çok kolay fakat çözülmesi zor olduğundan dolayı kombinatoryal optimizasyonda araştırmacılar tarafından en çok üzerinde durulan problemlerden birisidir. Bu problemde bir seyyar satıcı, N tane şehri her birini sadece bir kez ziyaret etmek koşuluyla toplam seyahat edilen mesafeyi minimize edecek bir rota bulmak zorundadır. Herhangi bir başlangıç noktasına göre (N-1)! kadar mümkün çözüm bulunmaktadır (eğer seyahat etme yönüne bakılmaksızın her şehir çifti arasındaki uzaklık aynı ise (N-1)!/2 kadar mümkün çözüm olmaktadır). 20 şehirli bir TSP için mümkün tüm çözümleri 1 saatte listeleyen bir bilgisayar olduğunu düşünün. Bu durumda (N-1)! formülüne göre 21 şehirli bir problem 20 saat, 22 şehirli problem 17,5 gün ve 25 şehirli bir problem yaklaşık 6 yüzyıl alacaktır. Problem boyutuna göre hesaplama zamanının üstel olarak artması diğer yaklaşımların kullanılmasını imkansız kılmaktadır [10].

Aynı problem bilgisayar açısından tekrar incelendiğinde, dünyanın yaklaşık 20 milyar yaşında olduğu kabul edildiğinde bu rakam $6.3072 \times 10^{17}$ saniye veya $6.3072 \times 10^{23}$ mikro saniye olacaktır. Eğer her mikro saniyede bir diziliş üreten bir bilgisayar mevcut ise, dünyanın oluşumundan bu yana sadece 24! ya da $6.20448 \times 10^{23}$ diziliş elde edilebilecekti. Bu duruma Gezgin Satıcı Problemi açısından bakarsak, sadece 24 şehri gezmek mümkün olacaktır.

Buradan, bir problemin çözümünün belirlenen bir algoritma ile ne kadar zaman aldığının önemi ortaya çıkmaktadır [11].

Hesaplama zorluğu (computational complexity), eniyi çözümün araştırmak yerine, sezgisel tekniklerin kullanılmasını daha mantıklı bir hale getirmiştir. Bu da modern sezgisel yaklaşımların artarak gelişmesini sağlamış ve bu metotların gücünde ve etkinliğinde önemli bir yükseliş ortaya çıkmıştır.

Kombinatoryal optimizasyon problemleri için diğer bazı örnekler aşağıda verilmektedir.

**1) Atama Problemleri (The Assignment Problem) [9,10]**

*n* elemanın, *n* farklı göreve atanması problemidir. *i.* kişinin, *j* işi yapma maliyeti $c_{ij}$ dir. Bu durumda problem, $\sum_{i=1}^{n} c_i \pi_i$ amaç fonksiyonunu minimize edecek $\{\Pi_1,...,\Pi_n\}$ atama kümesinin bulunması şeklinde tanımlanabilir. Burada problem çözümü, {1, ..., n} sayılarının $\{\Pi_1, ..., \Pi_n\}$ permütasyonu olarak gösterilmektedir.

**2) 0-1 Sırt Çantası Problemi (The 0-1 Knapsack Problem) [9,10]**

*n* parçanın, kapasitesi *C* birim olan bir sırt çantasına yerleştirilmesi problemidir. *i.* parça $v_i$ değerine sahip olup kapasitenin $c_i$ birimini kullanmaktadır. Buna göre *I* altkümesinin belirlenmesi istenmektedir:

$$\sum_{i \in I} v_i \text{ , buradan} \qquad (2.10)$$

$$\sum_{i \in I} c_i \leq C \qquad (2.11)$$

Bu problemde çözüm $I \subseteq \{1, ..., n\}$ altkümesi tarafından gösterilmektedir.

### 3) Küme Kapsama (Set Covering) Problemi [12]

Tam sayılı programlama model tiplerinden biri "küme kapsama" problemleridir. Küme kapsama, tüm kısıtların $\geq$ eşitsizliğine sahip olan, tüm sağ taraf değerlerinin 1 olduğu ve kat sayı matrisinin 0-1 matrisi olan saf 0-1 tam sayılı programdır.

Genel olarak küme kapsama problemlerinin formu aşağıdaki biçimde ifade edilebilir:

$$Min \quad z = \sum_{j=1}^{N} c_j x_j \tag{2.12}$$

$$\text{Kısıtlar} \quad Ax \geq b \tag{2.13}$$

$$x_j = 0 \text{ veya } 1 \qquad (j=1,\ldots, N) \tag{2.14}$$

Burada A, (m x n) şeklinde kısıt kat sayılarından oluşan bir 0-1 matris; b ise değeri 1 olan bir sağ taraf vektörüdür. Amaç fonksiyonu her zaman $c_1x_1 + c_2x_2 + c_3x_3 + \ldots + c_nx_n$ 'yi minimum yapacak şekildedir. Burada tüm $j = 1,2,..,n$'ler için $c_j > 0$ dır. Çoğunlukla $c_j = 1$'dir. Bununla birlikte, eğer $c_j$, $j$ yerindeki tesis maliyetini gösteriyorsa, bu durumda bu kat sayıların 1'den başka değerler alabileceği de varsayılır. Küme kapsamanın matematiksel tanımı daha kısa olarak şöyle de ifade edilebilir [12]:

$$\text{Minimum } z = \{cx \mid Ax \geq b,\ x = (0,1)\} \tag{2.15}$$

Yukarıdaki matematiksel modelde kısıtların işaretine ve sağ taraf değerine bağlı olarak özel problem tipleri ortaya çıkmaktadır. Eğer kısıtlardaki eşitsizlik, eşitlik hâline getirilirse ve sağ taraf değeri (b), 1 olarak kalırsa, bu tip problemlere *küme bölünme problemi* denir. Eğer kısıtların hepsi ($\leq$) hâline getirilirse ve sağ taraf değeri 1 olarak kalırsa, *küme paketleme problemi* denir [12].

Gerçek hayatta birçok uygulamayı paketleme, bölünme ve kapsama yapısında görmek mümkündür. Dağıtım ve rotalama problemleri, plânlama problemleri ve yerleştirme problemleri sık sık küme kapsama yapısında karşımıza çıkmaktadır. Bu

yapı sayesinde yerleşim, araç ya da insanlar tarafından her bir müşterinin ihtiyaçları karşılanabilir. Diğer uygulamalar ise elektrik devresi teorisi, VLSI (Very-Large-Scale Integration "Çok Geniş Ölçekli Entegre Devreler") devrelerinin test edilmesi ve (montaj) hat dengelemedir. Benzer bir yapı olan küme paketleme de özellikle plânlama problemlerinde herhangi bir çatışma yaratmadan olası talebi karşılamada yardımcı olur. Küme bölünmede ise bir müşterinin talebini bir servisçi karşılar. Bu üç yapı genellikle şu üç alanda görülebilir. Bir hava yolu şirketinin her uçuş seferine kesinlikle bir kokpit mürettebatı planlanması, bölgelerin oy verme bölgelerine bölünmesi ve her vatandaşın bir seçim bölgesine atanması [12].

**4) Araç Rotalama (Vehicle Routing ) Problemi [13]**

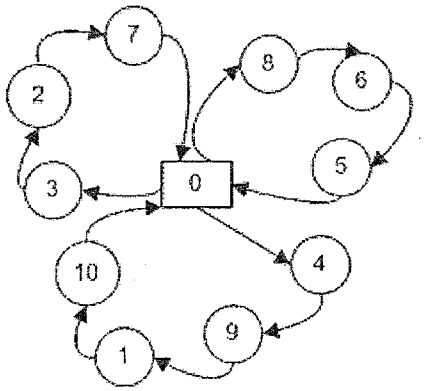
Araç Rotalama Problemi, ilk defa Dantzig ve Ramser tarafından 1950 yılında formülasyonu yapılan, müşterilerin taleplerini karşılamak amacıyla gezen araçların güzergâhlarının belirlenmesi şeklindeki problemlere verilen isimdir [13].

Bu problemde, verilen amaç fonksiyonuna göre araçlara rotalar belirlenir. Güzergâh belli bir depodan başlar ve coğrafik olarak dağılmış müşterilere tek bir araçla bir kez uğranarak talepleri karşılanıp tekrar depoya dönülür. Gezen aracın müşterilerin ihtiyaçlarını karşılayabilecek kapasitede olması gerekir. Yani toplam müşteri talebi araç kapasitesine eşit ve az olmalıdır. Araç kapasitesinin sınırsız olduğu durumlarda, depo veya müşterilerin arz ve talepleri yüzünden, Araç Rotalama Problemi (Vehicle Routing Problem -VRP), Gezgin Satıcı Problemine (TSP) dönüşür [13].

Bu tip problemlerde amaç fonksiyonu; toplam kat edilen mesafeyi minimize etmek veya toplam seyahat zamanını minimize etmek şeklinde olabilir.

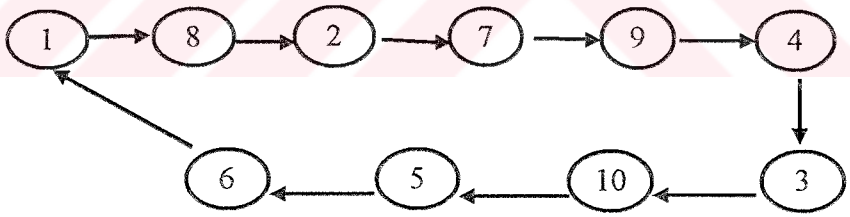
Toplam müşteri talepleri, araç kapasitesinden fazla ise bir araç, tek bir seferde müşteri taleplerini karşılayamayacağından dolayı; ya aynı araç belirlenen güzergâhları, her seferde depoya uğrayarak yükleme yapmak koşulu ile Şekil

2.2'deki gibi ayrı ayrı dolaşacak ya da her bir güzergâh için ayrı araç tahsisi yapılması gerekecektir.




Şekil 2.2 Araç Rotalama Problemi [13]

Toplam müşteri talepleri araç kapasitesine eşit veya daha az ise bu durumda tek bir araç, bir seferde her bir müşteriye uğrayarak taleplerini Şekil 2.3'te gibi karşılayabilir.




Şekil 2.3 Araç Rotalama Problemi [13]

Araç Rotalama Problemleri için değişik çözüm teknikleri geliştirilmiştir [13].

Tam Yaklaşım

- Dal-Sınır Tekniği (Fisher 1994)

Heuristik Teknikler

- Clark ve Wright (1964)
- Hiyerarşik Yaklaşım (Ayırma + Gezgin Satıcı Problemi)
- Fisher ve Jaikumur (1981)
- Taillard (1993)

- Birden Çok Güzergâhlı Sezgisel Teknikler
- Kinderwater ve Savelsbergh (1997)

Meta-sezgisel teknikler

- Tabu Araştırması, Rochat ve Taillard (1995)
- Tahdit Programlaması, Shown (1998)
- Tabu Araştırması, Kelly ve Xu (1999)
- Granular Tabu, Toth ve Vigo (1998)
- Ant Sistemi, Gambardella (1999)

Son otuz yıllık periyotta yönetim bilimi, bilgisayar, mühendislik gibi bir çok farklı alanda ortaya çıkan kombinatoryal optimizasyon problemlerinin büyük çoğunluğu, "NP-Zor (NP-hard)" problemler sınıfında yer almaktadır. İzleyen bölümde bu tür problemlerde çok sık karşılaşılan P ve NP terimlerinin tanımları yer almaktadır.

### 2.2.1 P ve NP Sınıfı Problemler

Atama problemi gibi bilinen bir polinomial algoritmaya sahip olan problemler P sınıfı problemler olarak adlandırılmaktadır. Fakat TSP ve diğer zor problemlerin böyle bir polinomial algoritmaya sahip olup olmadıkları henüz bilinmemektedir. Bu nedenle bu tür problemler NP (non-deterministic polinomial) sınıfı problemler olarak bilinmektedirler. NP sınıfı problemler, deterministik olmayan (rastgele seçilen bir çözüm olasılığını test eden) turing makinesiyle polinom zamanda çözülebilen karar problemlerini (evet-hayır cevaplı) içerirler. Bunun en basit anlamı ise NP sınıfının içerdiği problemler mümkün bütün çözümlerin denenmesiyle çözülebilir ki bu da problemin çözülmesi için gereken zamanın problemin büyüklüğünün üstel, yani çok büyük bir fonksiyonu olması demektir [10].

Bu konunun açıklanabilmesi için dönüştürülebilirlik (transformability, reducibility) kavramının açıklanmasını gerekmektedir. A algoritmasıyla çözülebilen bir $P_1$ problemi olsun. Eğer başka bir $P_2$ probleminin tüm örnekleri polinomial

zamanda $P_1$ probleminin herhangi bir örneğine dönüştürülebiliyorsa, A algoritması $P_2$ problemini çözmek için kullanılabilir. $P_1$ örneklerinin sınıfı en azından dönüştürülen $P_2$ örneklerinin sınıfı kadar geniştir. Böylece $P_2$ probleminin de en azından $P_1$ problemi kadar zor olduğunu düşünmek gerekir [10].

Süper bilgisayarlarla yapılan şifreleme tekniğini tarihe gömecek olan "P'ye karşı NP" problemi, bilgisayarcıların hangi algoritmanın hangi hesap işlemini hangi etkinlikte yapacağını araştırmalarıyla ortaya çıkmıştır. Genel olarak bir bilgisayar programına ne kadar çok veri yüklenirse, programın bu verileri işleme süresi o ölçüde uzamaktadır. Örneğin, bir dosyalar listesini alfabetik sıraya koyacak bir algoritma dosyaların sayısını ikiye katlandığında, programın bunları sıraya sokması için gereken süre dörde katlanacaktır. Bilgisayar bilimi dilinde bu, bir $N^2$ algoritması olmaktadır. Pekçok işlem için programcılar bu gibi "polinomial süre" ya da P algoritmaları kullanmaktadırlar. Çünkü işlemlerin çözüm zamanı çok uzun zaman gerektirebilmektedir. Çok haneli sayıların çarpanlarına ayrılması gibi polinomial sürede çözülemeyecek problemler bile, polinomial süre içinde sağlanabilir. Örneğin büyük bir sayıyı çarpanlarına ayırdığını söyleyen birinin doğru yapıp yapmadığını kontrol etmek için çarpanları birbirleriyle çarpmanız yeterlidir. Böyle polinomial süre içinde kontrolü yapılabilecek bir probleme NP denilmektedir [14].

Eğer NP sınıfındaki her problem polinomial olarak bir P problemine dönüştürülebiliyorsa bu durumda P problemi **NP-hard** olarak adlandırılmaktadır. Ayrıca eğer P probleminin kendisi de NP sınıfına ait ise, P problemi **NP-complete** olarak bilinmektedir. NP-complete problemler NP sınıfındaki problemlerin en zorudur. Eğer NP-complete bir problem için polinomial zamanda hesaplanan bir algoritma bulunabilirse NP sınıfındaki bütün diğer problemler için de bir polinomial algoritma var demektir. Böylece P = NP olduğu gösterilmiş olacaktır [10]. Fakat bu şimdiye kadar mümkün olmamıştır.

Tüm P algoritmaları birer NP'dir. Çünkü bir şeyi polinomial süre içinde çözebiliyorsa, başkasının bulduğu bir çözüm de polinomial süre içinde kontrol edilebilir. Fakat 1971'de bilgisayar bilimcisi Stephen Cook, bir NP algoritmasının aynı zamanda bir P algoritması olup olmadığını sormuştur. Büyük sayıları

çarpanlarına bölmek gibisinden NP problemlerinin polinomial süre içinde çözümünün bilinen örneği yoktur. Ancak bunu kanıtlamak göründüğü gibi kolay değildir. Matematikçiler NP-Complete denen ve NP problemlerinin en zor türü olan problemlerin birbirlerine eşit olduğunu kanıtlamışlardır. Böyle olunca da bir NP-Complete problemin polinomial süre algoritması, bu çeşit tüm problemlerin çözümü için uyarlanabilir. Cook, "böyle bir algoritmayla her türlü şifreyi kırabilirsiniz" demiştir [14].

Bu problem, 2000 yılında, ABD Massachusetts Eyaleti Cambridge kentinde bulunan Clay Matematik Enstitüsü tarafından, matematik dünyasının en zor yedi probleminden biri olarak belirlenmiş ve çözümü için bir milyon dolar ödül konulmuştur.

Bu çözüm bulunana kadar sezgisel (araştırmaya yönelik) metotlar kullanmak zorunlu olacaktır.

### 2.2.2 Kombinatoryal Optimizasyon Problemlerinin Çözüm Metotları

Yöneylem Araştırmasında yapılan ilk çalışmalar bir problem için optimal çözümü bulmak üzerinde odaklanmıştır. Bu nedenle tam birerlemeden daha etkin çeşitli algoritmalar geliştirilmiştir. Bunun en ünlü örneği, doğrusal programlamadaki Simplex Algoritmasıdır. Bu tür algoritmalar, küçük boyutlu problemleri çözmede oldukça başarılıdırlar. Ancak büyük boyutlu problemler için ihtiyaç duydukları hesaplama zamanı normal şartların dışına çıkmıştır. Fakat hesaplama gücünün yükselmesiyle beraber, daha büyük boyutlu problemleri çözmek mümkün olmuş ve araştırmacılar problem boyutuna göre çözüm zamanının nasıl değiştiği konusu üzerinde çalışmalar yapmışlardır. Atama probleminin çözülmesi için Hungarian metot, iki makinalı sıralama problemi için Johnson's metodunda olduğu gibi bazı durumlar için hesaplama güçlüğünün, problem boyutuyla düşük mertebeli bir polinomial olarak arttığı gösterilmiştir. Fakat TSP gibi bir çok problem için hesaplama güçlüğü problem boyutuyla üstel olarak artmaktadır [10].

Bu durumda, dal/sınır tekniği veya dinamik programlama gibi kesin metotlar, tam birerlemeden daha etkin çözümler oluşturamamaktadırlar [10].

Literatürde kombinatoryal optimizasyon konusundaki çalışmaların çoğunluğu, sezgisel metotlardan çok kesin (exact) metotlar üzerinde odaklanmaktadır. Yani, verilen bir problem için optimum çözümü garanti eden teknikler üzerinde durulmaktadır. Bu metotlar genellikle doğrusal programlama veya graf teorilerine ya da dal/sınır veya dinamik programlama gibi kesin birerleme yaklaşımına dayanmaktadır. Fakat modelin en iyi çözümünün aynı zamanda ele alınan gerçek problemin en iyi çözümü olduğunun garantisi yoktur. Bu durumda, yaklaşık bir model için kesin (exact) bir çözüm veya kesin bir model için yaklaşık bir çözüm arasında tercih yapılmalıdır. Fakat gerçekte tam olarak gerçeği yansıtan kesin bir modelin olması beklenemez. Bu durumda sezgisel yaklaşımlar karmaşık ve gerçekçi amaç fonksiyonları ve/veya kısıtlardan oluşmuş kombinatoryal optimizasyon problemlerinin çözümünde, kesin algoritmalara göre genellikle daha etkindirler [10].

### 2.2.3 Sezgisel (Heuristic) Teknikler

Sezgisel kelimesi Yunanca kökenlidir ve bulmak, keşfetmek anlamına gelmektedir. Türkçe heuristic kelimesinin karşılığı buluşsal ya da araştırmaya yönelik olarak da kullanılmaktadır. Bir çözüm tekniği olarak sezgisel, herhangi bir şeyin bulunmasını garanti etmeyen bir "arama" (seeking) metodu olarak tanımlanmaktadır [10].

En genel anlamda, sezgisel, iyi yani eniyi çözüme yakın çözümleri araştıran bir tekniktir ve bu işlemi fizibiliteyi ya da eniyi sonucu bulmayı garanti etmeksizin makul bir hesaplama maliyeti ile yapar [10].

Kombinatoryal optimizasyonda sezgisel terimi, bir tümel optimumun bulunmasını garanti eden metotların aksini ifade etmektedir [10].

**Yerel ve Tümel Optimumlar**

Çoğu kombinatoryal problemlerin bir özelliği, birkaç tane doğru yani "global" optimum çözümlerin ve çok fazla sayıda lokal (yerel) optimum çözümlerin bulunmasıdır. Aşağıda "komşu yöresi" (neighborhood) kavramı açıklanarak bu konu üzerinde durulmaktadır [10].

Genel olarak bakıldığında, bir x çözümünün komşu yöresi olan N(x, σ ), x üzerinde basit bir işlem olan σ 'yı uygulayarak ulaşılabilecek çözümlerin kümesi olmaktadır. Böyle bir σ işlemi, x çözümüne bir elemanın nesnenin veya x'den bir nesnenin çıkarılması olabilir. Bir çözümdeki iki nesnenin yerinin değiştirilmesi de genellikle sıralama (squencing) problemlerinde uygulanan başka bir işlem örneğidir. Genellikle ve özellikle tabu aramada bu tür işleme "hareket" (move) adı verilmektedir. Eğer x çözümü, N(x, σ ) komşu yöresindeki çözümlerden daha iyi ise, x çözümü bu yöreye göre yerel bir optimum olmaktadır [10].

Bazı durumlarda, aynı zamanda tümel bir optimum olan bir yerel optimum çözümün elde edilmesini sağlayacak bir σ hareketinin bulunması mümkün olmaktadır. Örneğin bazı tek-makineli sıralama problemlerinde bir ikili iç-değişim (pairwise interchange) hareketi bu özelliğe sahiptir. Bununla beraber, çoğu problem için bu mümkün değildir ve kesin birerleme (implicit enumeration) yönteminin bazı şekillerinin kullanılması tümel optimum çözümün bulunması için zorunlu olmaktadır [10].

Sezgisellerin çoğu probleme özel bir yapıya sahiptirler. Bir problem için kullanılan bir sezgisel, başka bir problem için kullanılamamaktadır. Bununla beraber, problemlere uygulanmaları açısından daha genel özelliğe sahip olan tekniklere ilgi son yıllarda giderek artmıştır. Bu teknikler çok sayıda problemlere uygulanmışlar ve oldukça güçlü bulunmuşlardır [10].

### 2.2.3.1 Kombinatoryal Problemlerde Sezgisel Teknikler

Kombinatoryal optimizasyon problemlerinin uygun çözümlerini bulmak için çeşitli sezgiseller geliştirilmiştir. Sezgiseller, çözümünün çok zor olduğu bazı büyük ölçekli problemlerde uygun çözümleri veren güçlü tekniklerdir. Sezgiseller uzmanların deneyimlerini ve yararlı problem karakteristiklerini kapsar. Bu özellikler, *problemin büyüklüğü, doğrusallık ve altbölünebilirlik* olarak gösterilebilir. Ancak sezgisel bir teknikle uygun çözümü açıklamak, optimal çözümü göstermez ve yalnızca sayılmayan objektif kriterlerle oluşturulmuş çözüm yaklaşımıdır. Sezgisel teknikler için, gerçek hayat sistemlerinde uzman sistemler denilebilir [15].

Bu tür problemlerde optimal çözüm, max. veya min amaç fonksiyonların uygun bir çözümü olarak tanımlanabilir. Fakat burada unutulmaması gereken, optimal bir çözümden daha iyi, başka uygun çözüm olmadığıdır. Bir anlamda yaklaşık bir çözüm optimize edilmiş çözüm olmaktadır [15].

### 2.2.4 Kombinatoryal Optimizasyon Problemlerinde Meta-sezgisel Teknikler

Son zamanlarda araştırmacılar tarafından, kombinatoryal optimizasyon problemlerinin bir çözüm metodu olarak meta sezgisel yöntemlere, giderek artan bir ilgi gösterilmektedir. Meta-sezgisel metodlar arasında, sinir ağları, tavlama benzetimi ve genetik algoritmalar örnek olarak verilebilir. Meta sezgiseller, sezgisel tekniklerle optimize edilmiş fakat yeterli olmayan durumlarda, etkili olan, standart sezgisel kavramının yüksek bir mertebesidir [15]. Uygun çözümler için diğer sezgisel arama tekniklerine kılavuzluk eden yüksek mertebeli stratejiler şeklinde söylenebilirler. Meta-sezgiseller yerel arama için standart sezgisellerin avantajlarına sahiptir [16].

Meta-sezgisel yöntemler, yerel optimallikten öte çözümler üretmek için, normal sezgileri değiştiren ve onlara kılavuzluk eden amir bir taktiği işaret eder [13].

Tavlama benzetimi (simulated annealing), tabu arama (tabu search) ve genetik algoritmalar gibi meta-sezgisel metotlar amaç fonksiyonlarının doğrusallık varsayımı

gibi bir basitleştirmeye ihtiyaç duymazlar. Böylece eğer kesin bir algoritma kullanılacaksa, gerçek-dünya probleminin modelinin mümkün olduğundan daha kusursuz bir şekilde kurulması gerekebilir [10].

Tavlama benzetimi ve tabu arama lokal komşuluk arama sezgiselinden hareketle geliştirilmişler ve oldukça güçlü oldukları gösterilmiştir. Lokal komşuluk arama sezgiseli esas olarak şu şekilde çalışmaktadır [10].

Proses herhangi bir başlangıç çözümü ile başlar ve bu çözümün tanımlanan komşuluğunu arayarak kendisinden daha iyi bir komşu çözümü olup olmadığını tespit eder. Eğer böyle bir çözüm varsa aynı işlemleri bu çözümün komşuluğu için tekrar eder, aksi halde proses durur. Bu durumda en son bulunan çözüm lokal olarak optimal bir sonuç olmaktadır [10].

Lokal optimaliteden kurtulmak için çeşitli yöntemler kullanılmaktadır. Bunlardan birisi komşuluğu genişletip içerisindeki komşu çözümleri çeşitlendirmektir. Bir diğeri ise, prosesi farklı başlangıç çözümleri kullanarak çok kez çalıştırmak ve elde edilen sonuçlar içerisinden en iyisini seçmektir. Son zamanlarda üzerinde durulan üçüncü bir yaklaşım ise "uphill" (yokuş-yukarı) hareketlere izin vererek lokal bir optimumda durmadan mevcut çözümden daha kötü olsa bile onun komşu çözümü üzerinden aramaya devam etmektir. Ancak "uphill" hareketlerin kabul edilmesi üzerinde bazı kısıtların olması zorunludur, aksi halde prosedür tüm çözüm uzayını saymaya çalışacaktır. Fakat bu prensibi kullanarak lokal bir optimumdan sıçrama ve daha iyi bir çözümü bulma şansı ortaya çıkmaktadır. Tabu arama ve tavlama benzetimi lokal optimumlardan sakınmak için bu üçüncü yöntemi farklı yaklaşımlarla kullanmaktadırlar [10].

**Modern metotlar**

Sezgiseller genellikle aşağıdaki kategorilere ayrılmaktadır [10]:

- Açgözlü yapılanma (greedy construction metotlar)
- Komşu yöresi arama (neighborhood search)
- Gevşetme (relaxation) teknikleri
- Kısmi birerleme (partial enumeration)

- Ayrıştırma (decomposition)
- Bölüştürme (partition yaklaşımları)

Modern sezgisel teknikler bazı doğal oluşum proseslerini taklit etmeye çalışmaktadırlar. Tavlama benzetimi, termodinamik prosesleri taklit etmekte, tabu arama ise bir çeşit hafıza (memory) uygulamasını dikkate alarak zeki (intelligent) prosesleri taklit etmektedir. Genetik algoritmalar ise, genetik yapılara benzer bir şekilde problemleri formüle ederek doğanın en iyi olan hayatta kalır prensibini taklit etmektedirler.

Kombinatoryal optimizasyon problemlerinin çözümünde kullanılan meta-sezgisel yöntemlerden Tavlama Benzetimi, Tabu Arama ve Genetik Algoritmalar izleyen bölümde açıklanmaktadır.

#### 2.2.4.1 Tavlama Benzetimi (Simulated Annealing)

Tavlama Benzetimi (TB), kombinatoryal optimizasyon problemleri için iyi çözümler veren olasılıklı bir arama yöntemidir. "Tavlama Benzetimi" ismi, katıların fiziksel tavlanma süreci ile olan benzerlikten gelmektedir [10]. Fiziksel tavlama, bir katının düşük enerjili durumlarının elde edilmesi prosesidir. Eritilen katının çok yavaş düşürülmesi ile katının düşük enerjili durumuna ulaşması sağlanır [17]. TB algoritması, birbirlerinden bağımsız olarak, Kirkpatrick, Gelatt ve Vecchi (1983) ile Cerny (1985) tarafından hemen hemen aynı yıllarda önerilmiş ve literatüre girmiştir. Bilgisayar tasarımı, görüntü işleme (*image processing*), moleküler fizik ve kimya, çizelgeleme problemleri, haberleşme şebekeleri tasarımı gibi farklı alanlardaki bir çok optimizasyon problemine uygulanmıştır [10].

Fiziksel tavlama ile kombinatoryal optimizasyon arasındaki ilişki aşağıdaki gibi ifade edilebilir [17].

| Termodinamik Simüslayon | | Kombinatoryal Optimizasyon Problemi |
|---|---|---|
| Sistem Durumları | ⟷ | Uygun Çözümler |
| Enerji | ⟷ | Amaç fonksiyonu |
| Durumun Değişimi | ⟷ | Komşu çözüm |
| Sıcaklık | ⟷ | Kontrol parametresi |
| Donma Durumu | ⟷ | Sezgisel Çözüm |

TB, yerel arama yöntemlerinin yerel minimuma ulaştıktan sonra global minimum için daha fazla arama yapmamasından kaynaklanan eksikliğini gidermeye çalışan bir yöntemdir. TB, bir istisnası dışında yerel arama yöntemindeki aynı temel adımları kullanır. Bazen yeni bir çözümün (j) amaç fonksiyon değeri (f(j)) mevcut çözümün (i) amaç fonksiyon değerinden büyük olmasına rağmen (yani $\Delta>0$, $\Delta=f(j)-f(i)$) yeni çözüm mevcut çözüm olarak kabul edilir ve arama işlemine devam edilir. Daha açık bir ifade ile, $\Delta\leq 0$ olduğunda, j mevcut çözüm olarak kabul edilir. $\Delta>0$ ise j, $e^{-\Delta/T}$ olasılığı ile mevcut çözüm olarak kabul edilir. T, fiziksel tavlamadaki sıcaklığı gösterir. Genellikle arama işlemine yüksek sıcaklık ile başlanır ve arama işlemi sırasında yavaş yavaş düşürülür. Bu strateji ile aramanın başlangıç aşamalarında amaç fonksiyonunda büyük artışların olduğu yeni çözümler kabul edilirken, aramanın sonuna doğru (sıcaklık sıfıra yaklaşırken) sadece amaç fonksiyonunda iyileşme sağlayan çözümler kabul edilir. Böylece TB, aramanın başlangıç aşamalarında kötü çözümleri de kabul ederek arama uzayının çeşitli bölgelerinde aramayı gerçekleştirmekte ve yerel minimuma takılmayı önlemektedir. Basit bir TB algoritmasının adımları aşağıda görülmektedir [17]

**Adım 1.** Uygun (feasible) bir başlangıç durumu seç: $i \in S$;
Bir başlangıç sıcaklığı seç: $T > 0$;
Sıcaklık değişim sayacını sıfırla t: = 0

**Adım 2.** Durdurma koşulu sağlanmışsa DUR, değilse tekrar sayacını sıfırla: n=0 ve devam et.

**Adım 3.** i'nin bir komşusu olan j durumunu rassal olarak üret.
$\Delta=f(j)-f(i)$

Eğer $\Delta < 0$ ise i = j, değilse ve U(0, 1) < exp(-$\Delta$ /T) ise i = j.

**Adım 4.** n = n + 1. Eğer n < M ise adım 3'e git, değilse t = t + 1, T= T(t) ve adım 2'ye git.

#### 2.2.4.1.1 Tavlama Benzetimi Algoritması

TB algoritmasında görülen, M her sıcaklık değerinde denenecek hareket sayısını ve T(t), t. sıcaklık değerini göstermektedir. TB algoritmasının global optimum çözümlere yakınsama hızı, M ve T(t), t = 0, 1, 2, ... parametreleri tarafından belirlenmektedir. Ancak pratikte, algoritmanın parametre değerlerinin uygulamaya yönelik seçimi "*tavlama*" veya "*soğutma planı*" ile belirlenmektedir. TB algoritmasında, başlangıç sıcaklığının, sıcaklık azaltma oranının, her sıcaklıktaki tekrar sayısının (komşu çözüm sayısı) ve durdurma koşulunun belirlenmesi tavlama veya soğutma planı olarak tanımlanmaktadır. Soğutma planının seçimi, algoritmanın performansı üzerinde çok önemli bir etkiye sahiptir [10].

#### 2.2.4.1.2 Tavlama Benzetimi Algoritmasının Uygulanması

TB algoritmasının uygulanması için verilmesi gereken bazı önemli kararlar; probleme özgü seçimler ve tavlama (veya soğutma) planına ait seçimler olmak üzere iki gruba ayrılabilir [10].

**Probleme Özgü Seçimler**

Problem, mümkün çözümlerin kümesi tanımlanabilecek şekilde formüle edilmelidir. Ayrıca, herhangi bir çözümün gösterimi (kodlama yapısı), herhangi bir komşu çözümün nasıl elde edileceği (hareket mekanizması) ve çözümlerin amaç fonksiyonu değerlerinin hesaplanma yöntemi tanımlanmak zorundadır. Bunun yanı sıra, bir başlangıç çözümünün üretilmesi gerekmektedir. Bazı araştırmacılar (Johnson ve diğerleri, 1991; Woodruff, 1994; Shapiro ve Alfa, 1995), kullanılan komşu

yapısının TB algoritmasının performansını etkilediğini göstermişlerdir. Genelde, yerel optimumların sığ olduğu "düzgün" bir arama uzayını gösteren komşu yapısı, yerel optimumların derin olduğu "engebeli" bir arama uzayını gösteren komşu yapısına tercih edilmektedir [10].

Eğer kısıtlı bir problem söz konusuysa, çözüm uzayı sadece kısıtları sağlayan çözümler ile sınırlandırılmalıdır veya kısıtları bozan çözümler uygun bir ceza fonksiyonu dikkate alınarak çözüm uzayına dahil edilmelidir.

**Tavlama Planına Ait Seçimler**

Sıcaklık parametresinin azalan değerlerinin sonlu bir sırası için sonlu uzunluklu homojen Markov zincirleri üretilerek oluşturulan bir TB algoritmasının, uygulaması da sonlu bir zaman alacaktır. Yakınsamanın sağlanabilmesi için bu algoritmada kullanılan parametreler kümesi uygun bir şekilde belirlenmek zorundadır. TB algoritmasında kullanılan parametrelerin ve değerlerinin belirlenmesi tavlama veya soğutma planı olarak tanımlanmaktadır. Tavlama planı ile aşağıdaki parametreler belirlenmektedir [10].

1. T sıcaklık parametresinin başlangıç değeri,
2. Sıcaklığın hangi yöntemle azaltılacağını belirlemek için kullanılan T(t) sıcaklık fonksiyonu,
3. Her sıcaklıkta gerçekleştirilmesi gereken M tekrar sayısı,
4. Algoritmayı durdurmak için T sıcaklık parametresinin son değeri.

Günümüze kadar çeşitli tavlama planları önerilmiştir. Önerilen en eski plan Kirkpatrick ve diğerlerinin (1983) fiziksel tavlama ile olan benzerliğe dayanarak ileri sürdükleri plandır. Bu tavlama planına göre, maddenin sıvı safhaya ulaştığında tüm parçacıklarının rassal olarak düzenlenmesini taklit etmek için, T sıcaklık parametresinin başlangıç değeri, denenen tüm hareketler kabul edilecek kadar yüksek seçilmiştir. Sıcaklık parametresinin değerini azaltmak için ise oransal bir sıcaklık fonksiyonu kullanılarak sabit bir $r$ için $T(t + 1) = r.T(t)$ dikkate alınmıştır. Burada $r$, 1'den küçük fakat 1'e yakın bir sabittir ve pratikte genellikle 0.80 ile 0.99 arasında

bir değer almaktadır. Bu sıcaklık fonksiyonu ile sıcaklık parametresinin değeri, sıfıra yaklaştıkça daha da yavaş azalmaktadır. Sıcaklık parametresinin her değerinde gerçekleştirilecek M tekrar sayısı, sabit bir üst sınıra göre kabul edilen yeterli sayıda geçişler (hareketler) tarafından belirlenmiştir. Böylece problemin, fiziksel tavlamadaki ısıl dengeye karşılık gelen bir denge durumuna ulaşması amaçlanmaktadır. M tekrar sayısı, sabit veya problemin yada komşu yapısının boyutuyla orantısal alınabilmektedir. Bu tavlama planı ile, sıcaklık parametresinin her değerinde elde edilen çözüm, belli sayıda ardıl sıcaklık değişimleri boyunca aynı kalırsa TB algoritması durdurulmaktadır. Buna göre elde edilen son durum, fiziksel tavlamadaki "*donma durumuna (frozen state)*" karşılık gelmektedir [10].

### 2.2.4.2 Tabu Arama (Tabu Search)

Tabu Arama tekniğine bağlı ilk çalışmalar Glover (1986, 1989, 1990) tarafından yapılmıştır. Teknik genel zeki problem çözme eğilimlerinden kaynaklanmaktadır ve zekice problem çözme prensiplerini ortaya çıkarmaya çalışır. Tabu arama tekniğinin yapay zeka ve optimizasyon alanlarını birleştiren kavramlara dayandığı söylenebilir [18].

Tabu Arama yönteminin özü uygun yada yerel optimallik sınırlarını aşmak için tasarlanan prosedürlere dayanır. Tabu Arama yerel optimalliğin ötesindeki çözüm uzayını keşfetmek için yerel bir sezgisel arama prosedürüne kılavuzluk eden bir meta-sezgidir. Tabu Arama bu kılavuzluğu, hafızasında aramanın hikayesini tutarak, yani bazı adaylara "yasak" koyarak, aramayı sınırlandırması ve yerel optimallikten kurtarmasıyla yerine getirmektedir. Bununla beraber, Brucker (1995)'e göre, genel Tabu Arama çerçevesi, belli bir algoritmayı zorlamaktan ziyade, çözüm prosedürlerinin tasarımı için üst düzey bir serbestliğe izin vererek yeni taktiklerin keşfedilmesini sağlar. Laguna ve Glover (1996)'ya göre de, Tabu Arama çok çeşitli iş problemlerinin çözümü için bir kapı açan yenilikçi bir yaklaşımdır. Bu yaklaşımın amacı hafızanın zekice kullanımını uyarlamaktır [18].

Tabu Arama'nın işleyişi basitçe şöyledir.

Bir başlangıç seçimi ele alınır. Bu seçimin komşuları bir komşuluk yapısıyla belirlenir. Bir amaç fonksiyonuna göre komşular değerlendirilir. Değerlendirilen komşu tabu listesinde yoksa ya da aspirasyon kriterini tatmin ediyorsa bu çözüm şimdiye kadar bulunan en iyi çözümle karşılaştırılır ve tabu listesine eklenir, eğer çözüm en iyi çözümden daha iyi ise sonraki arama için yeni başlangıç çözümü olarak alınır. Bu işlem bir durdurma kriteri karşılanıncaya kadar tekrarlanır [18].

Burada Tabu Arama ile ilgili bazı kriterler ön plana çıkmaktadır. Bunlar başlangıç çözümü, komşuluk yapıları, tabu listesinin düzenlenmesi, aspirasyon kriteri, durdurma kriteri, yoğunlaşma ve çeşitlendirme taktikleridir [18].

**Başlangıç Çözümü**

TS herhangi bir olurlu çözümle işe başlamaktadır. Bu problemin matematik ifadesinde geçen kısıtları tatmin eden yada kör döngüye yol açmayacak şekilde sıralayan bir çözümdür. Doğal olarak TS'ın performansı başlangıç çözümüne sıkıca bağladır. Bu yüzden TS'ya mümkün olduğunca iyi bir çözümle başlanmalıdır. Sevketme kuralları genellikle kullanılmaktadır. Bunun yanı sıra sözdizimsel (lexicographic) sıra ve etkin araya girme algoritmaları yöntemleri kullanılmaktadır [18].

**Komşuluk Yapıları**

Yerel arama yöntemlerinin kalitesi kullanılan komşuluk yapısına bağlıdır [18].

**Tabu Listesinin Düzenlenmesi**

Tabu listesi belli bir süre boyunca tekrar göz önüne alınmaması gereken çözümleri karakterize eden özellikleri saklar. Genellikle listeye ilk giren ilk çıkar stratejisi uygulanır. Listedeki varlık sayısı liste uzunluğuna ulaştıktan sonra listeye yeni varlıklar tepeden girdikçe eski varlıklar bir aşağı kayar ve en dipteki silinir [18].

### 2.2.4.3 Genetik Algoritmalar

Genetik algoritma, doğadaki evrim mekanizmasını örnek alan bir arama metodudur ve bir veri grubundan özel bir veriyi bulmak için kullanılır. Genetik algoritmalar 1970'lerin başında John Holland tarafından ortaya atılmıştır [19].

Genetik Algoritmaların teorik kısmı ve uygulamaları hakkında birçok uluslar arası konferans düzenlenmektedir. Genetik algoritmaların, fonksiyon optimizasyonu, çizelgeleme, mekanik öğrenme, tasarım, hücresel imalat gibi alanlarda başarılı uygulamaları bulunmaktadır. Geleneksel optimizasyon yöntemlerine göre farklılıkları olan genetik algoritmalar, parametre kümesini değil kodlanmış biçimlerini kullanırlar. Olasılık kurallarına göre çalışan genetik algoritmalar, yalnızca amaç fonksiyonuna gereksinim duyar. Çözüm uzayının tamamını değil belirli bir kısmını tararlar. Böylece, etkin arama yaparak çok daha kısa bir sürede çözüme ulaşırlar. Diğer önemli üstünlüleri ise çözümlerden oluşan populasyonu eş zamanlı incelemeleri ve böylelikle yerel en iyi çözümlere takılmamalarıdır [19].

"Reeves; ulusal hükümetler ve organizasyonlar tarafından genetik algoritma tabanlı projelere, tavlama benzetimi (simulated annealing) ve tabu arama tabanlı projelere göre daha fazla kaynak ayrılmakta olduğunu belirtmektedir." Bu ifadeden anlaşılacağı üzere, genetik algoritmaların geleneksel optimizasyon yöntemlerine olduğu gibi adı geçen yapay zeka yöntemlerine göre de çeşitli alanlarda üstünlükleri bulunmaktadır. Bu üstünlükler genetik algoritmaların arama yapısı ile ilgilidir. Genetik algoritmaların arama yapısı ise, alt diziler teoremi ve yapı blokları hipoteziyle açıklanmaktadır [19].

#### 2.2.4.3.1 Genetik Algoritmaların Temel Teoremi

Genetik algoritmaların nasıl arama yaptığı alt dizi kavramıyla açıklanmaktadır. Alt diziler, genetik algoritmaların davranışlarını açıklamak için kullanılan teorik yapılardır. Bir alt dizi, belirli dizi kümeleri arasındaki benzerliği tanımlayan bir dizidir. Alt diziler, {0, 1, *} alfabesi kullanılarak tanımlanır. Örneğin bir H alt dizisi,

ilk konumunda 0, ikinci ve dördüncü konumunda 1 değeri olan kromozomlar kümesi içindir [19].

H=0 1 * 1 *

* sembolü dizinin o konumunun hangi değeri alıp almadığının önemli olmadığı anlamındadır. Dizi o konumda 0 veya 1 değeri alabilir. Eğer bir x dizisi, alt dizinin kalıbına uyarsa x dizisine "H'nin bir örneğidir", denir. Alt dizilerin iki özelliği mevcuttur. Bu özellikler aşağıda verilmiştir [19].

**1. Alt dizi derecesi:** Bir H alt dizisinin derecesi o(H) ile gösterilir ve mevcut alt dizi kalıbında bulunan sabit konumların sayısıdır. Bu sayı ikili alfabede 0 ve 1 değerlerinin sayısının toplamına eşittir [19].

**2. Alt dizi uzunluğu:** Bir H alt dizisinin uzunluğu δ(H) ile gösterilir ve mevcut alt dizi kalıbında bulunan belirli ilk ve son konumlar arasındaki uzaklıktır [19].

Alt dizi derecesi ve alt dizi uzunluğu kavramlarının genetik algoritmaların temel teoreminde son derece önemli yeri vardır. Alt dizi derecesi düşük, alt dizi uzunluğu kısa olan diziler "yapı blokları olarak adlandırılırlar. John Holland, genetik algoritmaların işleyişinde uygun yapı blokları elde etmek amacıyla birleştirilmesini önermektedir. Bu fikir yapı blokları hipotezi olarak bilinmektedir. Genetik algoritmanın temel teoremi ise şöyle açıklanmaktadır:

Populasyon ortalamasının üstünde uyum gücü gösteren, kısa uzunluğa ve düşük dereceye sahip alt diziler zamının ilerlemesiyle üstel çoğalırlar [19].

Bu çoğalma, genetik işlemler aracılığıyla gerçekleşmektedir ve sonucunda ana-babadan daha üstün özellikler taşıyan bireyler ortay çıkmaktadır. Bu çözüm kalitesinin kuşaktan kuşağa artması iki nedene bağlanmaktadır. Bu nedenler şöyle açıklanabilir [19]:

- Başarısız olma bireylerin üreme şansları azaltıldığı için kötüye gidiş zorlaşmaktadır.
- Genetik algoritmaların yapısı kötüye gidişi engellemekle kalmamakta, genetik algoritmaların temel teoremi uyarınca, zaman içinde hızlı bir iyiye gidişi de sağlayabilmektedir.

Genetik algoritmaların işleme adımları incelendiğinde bu nendeler daha iyi anlaşılmaktadır. Genetik algoritmalar yapısı gereği, kötü bireyleri yani uygun olmayan çözümleri, operatörleri sayesinde elemektedir. Bu işlemler bir döngü içerisinde durdurma kriteri sağlanana kadar devam etmektedir [19].

#### 2.2.4.3.2 Basit Genetik Algoritma

Bir çok alanda uygulamaları bulunan genetik algoritmaların işleme adımları aşağıdaki gibi açıklanabilir [19].

- Arama uzayındaki tüm mümkün çözümler dizi olarak kodlanır.
- Genellikle rastsal bir çözüm kümesi seçilir ve başlangıç populasyonu olarak kabul edilir.
- Her bir dizi için uygunluk değeri hesaplanır, bulunan uygunluk değerleri dizilerin çözüm kalitesini gösterir.
- Bir grup dizi belirli bir olasılık değerine göre rastsal olarak seçilip çoğalma işlemi gerçekleştirilir.
- Yeni bireylerin uygunluk değerleri hesaplanarak, çaprazlama ve mutasyon işlemlerine tabi tutulur.
- Önceden belirlenen kuşak sayısı boyunca yukarıdaki işlemler devam ettirilir.
- İterasyon, belirlenen kuşak sayısına ulaşınca işlem sona erdirilir. Amaç fonksiyonuna göre uygun olan dizi seçilir.

Genetik algoritmalar bir çözüm uzayındaki her noktayı, kromozom adı verilen ikili bit dizisi ile kodlar. Her noktanın uygunluk değeri vardır. Tek bir nokta yerine, genetik algoritmalar bir populasyon olarak noktalar kümesini muhafaza eder. Her kuşakta, genetik algoritma çaprazlama ve mutasyon gibi genetik operatörleri kullanarak yeni bir populasyon oluşturur. Birkaç kuşak sonunda, populasyon daha iyi uygunluk değerine sahip olan üyeleri içerir. Bu, Darwin'in rastsal mutasyon ve doğal seçime dayanan evrim modellerine benzemektedir. Genetik algoritmalar, çözümlerin kodlanmasını, uygunlukların hesaplanmasını, çoğalma, çaprazlama ve mutasyon operatörlerinin uygulanmasını içerir [19].

#### 2.2.4.3.3 Çözümlerin Kodlanması

Bir problemin çözümü için genetik algoritma geliştirmenin ilk adımı tüm çözümlerin aynı boyutlara sahip bitler dizisi biçiminde gösterilmesidir. Dizilerden her biri, problemin olası çözümler uzayındaki rastsal bir noktayı simgeler. Parametrelerin kodlanması, probleme özgü bilgilerin genetik algoritmanın kullanacağı şekle çevrilmesine imkan tanır [19].

#### 2.2.4.3.4 İlk Populasyonun Oluşturulması

Mümkün çözümlerin kodlandığı bir çözüm grubu oluşturulur. Çözüm grubu populasyon, çözümlerin kodları da kromozom olarak adlandırılır. İkili alfabenin kullanıldığı kromozomların gösteriminde, ilk populasyonun oluşturulması için rastsal sayı üreticileri kullanılabilir. Rastsal sayı üreticisi çağrılır ve değer 0.5'ten küçükse konum 0'a değilse 1 değerine ayarlanır. Birey sayısının ve kromozom uzunluğunun az olduğu problemlerde yazı-tura ile de konum değerleri belirlenebilmektedir. Genetik algoritmalarda ikili kodlama yöntemi dışında, çözümü aranan probleme bağlı olarak farklı kodlama yöntemleri de kullanılmaktadır [19].

#### 2.2.4.3.5 Uygunluk Değerinin Hesaplanması

Bir kuşak oluşturulduktan sonraki ilk adım, populasyondaki her üyenin uygunluk değerini hesaplama adımıdır. Örneğin bir maksimizasyon problemi için i. üyenin uygunluk değeri f(i), genellikle o noktadaki amaç fonksiyonunun değeridir. Çözümü aranan her problem için bir uygunluk fonksiyonu mevcuttur. Verilen belirli bir kromozom için uygunluk fonksiyonu, o kromozomun temsil ettiği çözümün kullanımıyla veya yeteneğiyle orantılı olan sayısal bir uygunluk değeri verir. Bu bilgi, her kuşakta daha uygun çözümlerin seçiminde yol göstermektedir. Bir çözümün uygunluk değeri ne kadar yüksekse, yaşama ve çoğalma şansı o kadar fazladır ve bir sonraki kuşakta temsil edilme oranı da o kadar yüksektir [19].

#### 2.2.4.3.6 Çoğalma İşleminin Uygulanması

Çoğalma operatöründe diziler, amaç fonksiyonuna göre kopyalanır ve iyi kalıtsal özellikleri gelecek kuşağa daha iyi aktaracak bireyler seçilir. Üreme operatörü yapay bir seçimdir. Dizileri uygunluk değerlerine göre kopyalama, daha yüksek uygunluk değerine sahip dizilerin, bir sonraki kuşaktaki bir veya daha fazla diziye daha yüksek bir olasılıkla katkıda bulunması anlamına gelmektedir. Çoğalma, bireyleri seçme işleminden, seçilmiş bireyleri bir eşleme havuzuna kopyalama işleminden ve havuzda bireyleri çiftler halinde gruplara ayırma işleminden oluşur [19].

Uygunluk değerinin hesaplanması adımından sonra mevcut kuşaktan yeni bir populasyon yaratılmalıdır. Seçim işlemi, bir sonraki için dizi üretmek amacıyla hangi ailelerin yer alması gerektiğine karar vermektedir. Bu doğal seçimdeki en uygunun yaşaması durumuna benzerdir. Bu yöntemin amacı, ortalama uygunluğun üzerindeki değerlere çoğalma fırsatı tanımaktır. Bir dizinin kopyalanması şansı, uygunluk fonksiyonuyla hesaplanan dizinin uygunluk değerine bağlıdır. Seçim yöntemlerine rulet tekerleği seçimi, turnuva seçimi ve sıralama seçimi gibi seçim yöntemleri örnek verilebilir [19].

#### 2.2.4.3.7 Çaprazlama İşleminin Uygulanması

Mevcut gen havuzunun potansiyelini araştırmak üzere, bir önceki kuşaktan daha iyi nitelikler içeren yeni kromozomlar yaratmak amacı ile çaprazlama operatörü kullanılmaktadır. Çaprazlama (cross over) genellikle verilen bir çaprazlama oranına eşit bir olasılıkla seçilen aile çeşitlerine uygulanmaktadır [19].

Genetik algoritmanın performansını etkileyen önemli parametrelerden biri olan çaprazlama operatörü doğal populasyonlardaki çaprazlamaya karşılık gelmektedir. Çoğalma işlemi sonucunda elde edilen yeni populasyondan rastsal olarak iki kromozom seçilmekte ve karşılıklı çaprazlama işlemine tabi tutulmaktadır. Çaprazlama işleminde dizi uzunluğu L olmak üzere, 1<=k<=L-1 aralığındaki k tamsayısı seçilmektedir. Bu tamsayı değerine göre dizi çaprazlamaya uğratılır. En basit çaprazlama yöntemi tek noktalı çaprazlama yöntemidir. Tek noktalı çaprazlama yapılabilmesi için her iki kromozomun da aynı gen uzunluğunda olması gerekir. İki noktalı çaprazlamada ise kromozom iki noktadan kesilir ve karşılıklı olarak pozisyonlar yer değiştirilir [19].

#### 2.2.4.3.8 Mutasyon İşleminin Uygulanması

Çaprazlama mevcut gen potansiyellerini araştırmak üzere kullanılır. Fakat populasyon gerekli tüm kodlanmış bilgiyi içermez ise, çaprazlama tatmin edici bir çözüm üretmez. Bundan dolayı, mevcut kromozomlardan yeni kromozomlar üretme yeteneğine sahip operatör gerekmektedir. Bu görevi mutasyon gerçekleştirir. Yapay genetik sistemlerde mutasyon operatörü, bir daha elde edilemeyebilir iyi bir çözümün kaybına karşı koruma sağlamaktadır. İkili kodlama siteminin kullanıldığı problemlerde mutasyon, düşük bir olasılık değeri altında bir bit değerini (0 veya 1 olabilir) diğer bit değerine dönüştürür. İkili kodlama siteminin kullanılmadığı problemlerde ise daha farklı mutasyon yöntemleri kullanılmaktadır. Hangi yöntem kullanılırsa kullanılsın, mutasyonun genel amacı, genetik çeşitliliği sağlamak veya korumaktır [19].

#### 2.2.4.3.9 Yeni Kuşağın Oluşturulması ve Döngünün Durdurulması

Yeni kuşak çoğalma, çaprazlama ve mutasyon işlemlerinden sonra tanımlanmakta ve bir sonraki kuşağın ebeveynleri olmaktadır. Süreç yeni kuşakla çoğalma için belirlenen uygunluk kriteri ile devam eder. Bu süreç, önceden belirlenen kuşak sayısı kadar veya bir hedefe ulaşıncaya kadar ya da başka bir durdurma kriteri sağlanana kadar devam eder. İstenen bu hassasiyet derecesine göre de maksimum iterasyon sayısı belirlenebilmekte ve iterasyon bu sayıya ulaştığında döngü durdurulabilmektedir. Durdurma kriteri iterasyon sayısı olabileceği gibi hedeflenen uygunluk değeri de olabilmektedir [19].

#### 2.2.4.3.10 Genetik Algoritmalarda Parametre Seçimi

Parametreler, genetik algoritma performansı üzerinde önemli etkiye sahiptir. Optimal kontrol parametreleri bulmak için bir çok çalışma yapılmıştır fakat tüm problemler için genel olarak kullanılabilecek parametreler bulunamamıştır. Bu parametreler, kontrol parametreleri olarak adlandırılmaktadır. Kontrol parametreleri, populasyon büyüklüğü, çaprazlama olasılığı, mutasyon olasılığı, kuşak aralığı, seçim stratejisi ve fonksiyon ölçeklemesi olarak sayılabilir. Bu parametreler aşağıda açıklanmıştır [19].

**Populasyon Büyüklüğü:** Genetik algoritma kullanıcısı tarafından verilen en önemli kararlardan birisidir. Bu değer çok küçük olduğunda, genetik algoritma yerel bir optimuma takılabilmektedir. Populasyonun çok büyük olması ise çözüme ulaşma zamanını arttırmaktadır. Bu konuda Goldberg 1985'te, yalnızca kromozom uzunluğuna bağlı bir populasyon büyüklüğü hesaplama yöntemi önermiştir. Ayrıca Schaffer ve arkadaşları 1989'da çok sayıda test fonksiyonları üzerinde yaptıkları araştırmalar sonucunda, 20-30 arası bir populasyon büyüklüğünün iyi sonuçlar verdiğini belirtmişlerdir [19].

**Çaprazlama Olasılığı:** Çaprazlamanın amacı, mevcut iyi kromozomların özelliklerini birleştirerek daha uygun kromozomlar yaratmaktır. Kromozom çiftleri

P(c) olasılığı ile çaprazlamaya uğramak üzere seçilirler. Çaprazlamanın artması, yapı bloklarının artmasına neden olmakta fakat aynı zamanda bazı iyi kromozomların da bozulma olasılığını arttırmaktadır [19].

**Mutasyon Olasılığı:** Mutasyonun amacı populasyondaki genetik çeşitliliği korumaktır. Mutasyon P(m) olasılığı ile bir kromozomdaki her bitte meydana gelebilir. Eğer mutasyon olasılığı artarsa, genetik arama rastsal bir aramaya dönüşür. Fakat bu aynı zamanda kayıp genetik malzemeyi tekrar bulmada yardımcı olmaktadır [19].

**Kuşak Aralığı:** Her kuşaktaki yeni kromozom oranına kuşak aralığı denmektedir. Genetik operatörler için kaç tane kromozomun seçildiğini gösterir.yüksek bir değer bir çok kromozomun yer değiştirdiği anlamına gelmektedir [19].

**Seçim Stratejisi:** Eski kuşağı yenilemenin çeşitli yöntemleri mevcuttur. Kuşaksal stratejide, mevcut populasyondaki kromozomlar tamamen diziler ile yer değiştirir. Populasyonun en iyi kromozomu da yenilendiğinden dolayı bir sonraki kuşağa aktarılamaz ve bu yüzden bu strateji en uygun stratejisiyle beraber kullanılmaktadır. En uygun stratejisinde, populasyondaki en iyi kromozomlar hiçbir zaman yenilenmemektedir. Bundan dolayı çoğalma için en iyi çözüm her zaman elverişlidir. Denge durumu stratejisinde ise, her kuşakta yalnızca birkaç kromozom yenilenmektedir. Genellikle, yeni kromozomlar populasyona katıldığında en kötü kromozomlar yenilenir [19].

**Fonksiyon Ölçeklemesi:** Doğrusal ölçekleme, üstsel ölçekleme gibi yöntemler mevcuttur. Probleme göre en uygun ölçekleme yönteminin seçilmesi genetik algoritmanın etkin işlemesi açısından önem taşımaktadır [19].

### Genetik Algoritmaların Diğer Metotlardan Farkları

Genetik algoritmalar, diğer normal optimizasyon metotlarından dört açıdan farklılık göstermektedirler.

- Genetik algoritmalar, parametrelerin kendisi ile değil, parametre dizilerinin kodlanması ile çalışırlar.
- Genetik algoritmalar, tek noktadan değil, populasyonundan araştırılırlar.
- Genetik algoritmalar, sınırlayıcı kuralları değil, olasılık kurallarını kullanırlar. Bir çok optimizasyon metodunda, tek bir noktadan sonrakine değişim kuralı ile hareket etmektedir. Aksine genetik algoritmalar, eşzamanlı olarak geniş veri tabanlı noktalarla çalışmaktadırlar. Bu nedenle diğer metotlara göre global optimuma daha kolay ulaşırlar.
- Birçok araştırma tekniği yardımcı bilgiye ihtiyaç duyarken, genetik algoritmaların yardımcı bilgiye ihtiyaçları yoktur. Daha iyi sonuçlar elde etmek amacıyla, etkili araştırma yapabilmek için diziler ile amaç fonksiyon değerlerine ihtiyaç duyarlar.

Genetik algoritmalarda, direk kodlama kullanılması, populasyon araştırması, yardımcı bilgiye ihtiyaç duymama ve operatörlerin rastgele olması sebebiyle oluşan bu dört farklılık kararlığa katkıda bulunmaktadır [20].

# 3. YERLEŞTİRME PROBLEMLERİ

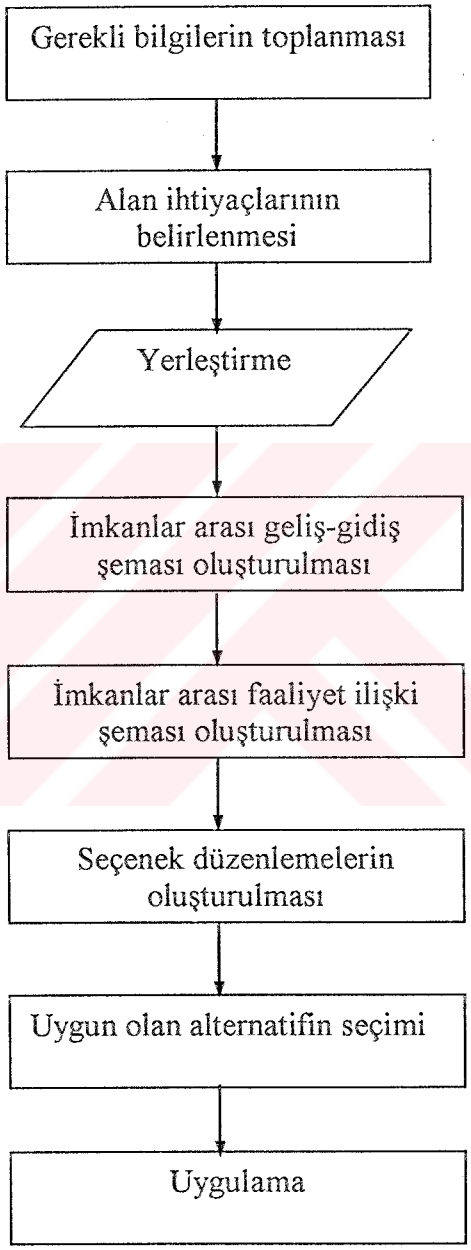
Bir imkanın nereye yerleştirileceği kullanıcılar açısından önemli olup yerleştirme ve yer seçimi problemleri ile ilgilidir. Yerleştirme problemleri yerleştirilecek imkan için en uygun yerin neresi olduğu sorusuyla başlar. Bu tür problemlerde ilk amaç yerleştirilecek imkanın en etkin, harcanan çabanın da en az olmasıdır. Ancak amaçlar problemin yer aldığı endüstri dalına göre farklılıklar gösterebilir. Günümüzde işletmeler, ekonomik olma zorunluluğundan dolayı mevcut kaynakları en etkin biçimde ve israf etmeden kullanmalıdırlar. Yerleştirme problemleri zaman ve malzemenin verimli kullanımı dikkate alındığında işletmelerin rekabet gücü için büyük öneme sahiptirler.

"Kuşlar için söylenebilecek olan her şey tesis planlama problemleri için de söylenebilir; sayıları ve çeşitleri neredeyse sonsuzdur." **(Francis)** Bu ifadeden de anlaşılacağı gibi yerleştirme problemi örnekleri çok çeşitli olabilir. Bunlardan bir kaçı;

- İlk yardım ve acil müdahale istasyonlarının yerinin seçimi,
- Fabrikalarda takım odası, yıkama alanı, depolama gibi birimlerin yerinin seçimi,
- Bir grup projede taşeron firmaların seçimi ve bunlara iş dağıtımı,
- Fabrikalarda makinaların yerinin seçimi,
- Kontrol panellerinde kontrol noktalarının seçimi, vb.

Bir işyerine ait yerleştirme çalışmalarının akış çizelgesi Şekil 3.1'de verilmiştir. Burada yerleştirilecek veya düzenlenecek olan imkanlar o işyerine ait makinalar veya bölümler olabilir. Bu akışın ilk aşaması şekilde belirtildiği gibi yerleştirme yapılacak olan bölgenin ve yerleştirilecek olan imkanların alanlarının tespiti ile başlar. Üretim veya hizmet sürecinin içerdiği kısıtlara göre bölümler veya makinalar arası ilişkiler incelenerek düzenlemeler yapılır. Az elemanlı bir yerleştirme probleminin çözümünde insan becerisi daha iyi iken çok elemanlı problemlerde tek başına bir tasarımcının belirgin ölçüde verimliliği düşmektedir. Bunu önleyebilmek için problemlerin türüne göre bilgisayar teknolojisinden yararlanılmaya çalışılmaktadır.

Çünkü insan beyni yapı olarak birim zamanda yaptığı işlem açısından bugünün en gelişmiş bilgisayarından daha hızlı olmasına rağmen, bilgisayarlar tekrarlı ve ardışık işlemleri yapmak ve de bütün olasılıkları değerlendirmek açısından daha başarılıdırlar.




Şekil 3.1 İmkan yerleştirme problemleri için akış diyagramı

## 3.1 Kesme ve Paketleme Problemlerinin Yerleştirme Problemleri İle İlişkisi

Bir işyeri düzenlemesi çalışmasında bölüm veya iş merkezi sayısının artması ile mümkün seçeneklerin sayısı çok daha fazla bir oranda artmaktadır. Bölümler arası ilişkiler, işlem sırası, hız gibi kısıtlar olması durumunda bunlar incelenecek ve seçenek sayısı azalacaktır.

Yerleştirme problemleri, kuadratik optimizasyon problemleri içinde yer alan, çözüm zamanının, problem boyutuna bağlı olarak üstel artış gösterdiği, çözülmesi zor NP-Hard problemlerdendir. Bu tür problemler, problem boyutunun polinom fonksiyonu ile sınırlı olan hesaplama zamanı içinde çözülemezler. Bu nedenle bu tür problemlerin çözümünde, optimumu bulmayı garanti etmeyen ancak, çözüm zamanı polinom sınırlar içinde kalan sezgisel metotlardan yararlanılır [21].

Yerleştirme problemleri bir işyerinde makinelerin veya bölümlerin nereye kurulacağı ile ilgilidir. Bunun için bölümlere veya makinelere ait özellikler ve kısıtlar belirlenir.

Boyutları belirli şekillerin bir yüzeye veya alana yerleştirilmesi, kesme ve paketleme problemleri konusudur. Tanımdan da anlaşılacağı gibi bu tür problemlerin temeli, eldeki imkanların en iyi şekilde yerleştirilmesini kapsar.

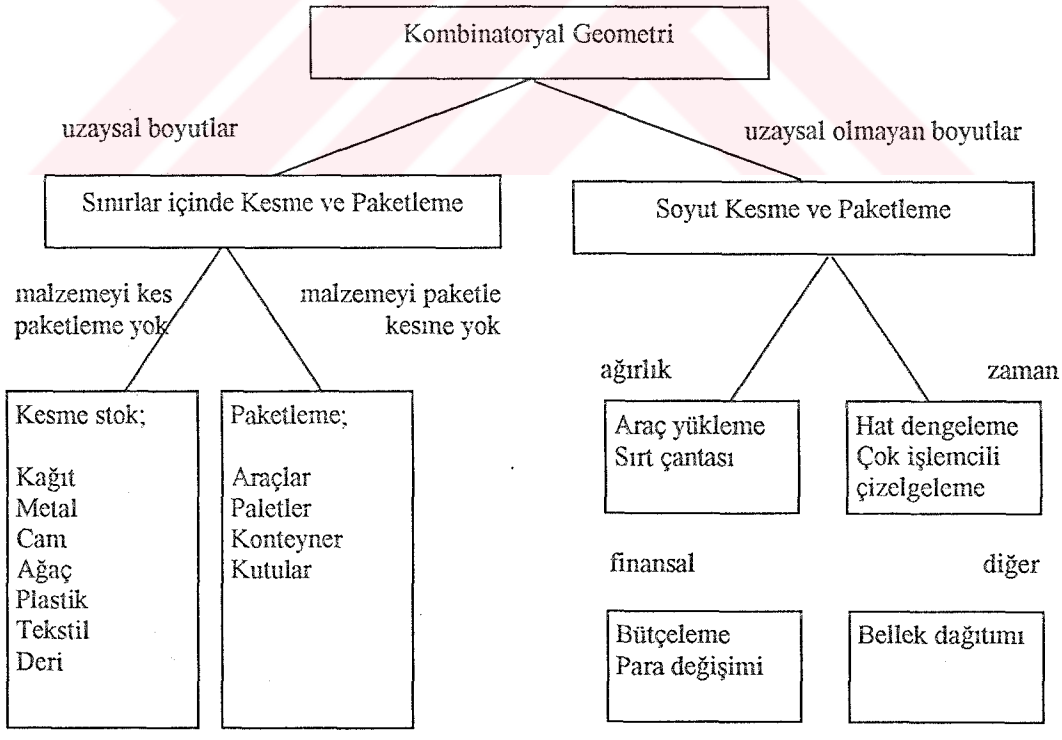
Kesme ve paketleme problemlerinde burada adı geçen kısıtlar olmayabilir. Örneğin tekstil endüstrisini ele alındığında üretilecek malzemenin kalıplarının kumaş üzerine yerleşim sırası ve parçalar arası ilişki, desen veya doku kısıtı yoksa önemli değildir. Buradaki amaç atık malzemenin en aza indirilmesidir. Kısıt olmadığı için oldukça fazla seçenek çıkabilecektir. Burada olduğu gibi çok elemanlı problemlerin grafik yoldan çözümü oldukça zor olduğundan bilgisayar destekli düzenlemelerden yararlanma yoluna gidilmektedir.

## 3.2 Kesme ve Paketleme Problemleri

Belirli bir düzlemsel alana dikdörtgensel elemanların yerleşimini kapsayan kesme ve paketleme problemleri, yöneylem araştırması alanında en önemli problemler arasında yer almaktadırlar. Uzun yıllar endüstriyel açıdan önemleri nedeni ile bu problemler üzerinde çalışılmıştır. Bu tür problemler yönetim bilimi, mühendislik, bilgi ve bilgisayar teknolojisi, matematik gibi farklı disiplinleri ilgilendirmiştir.

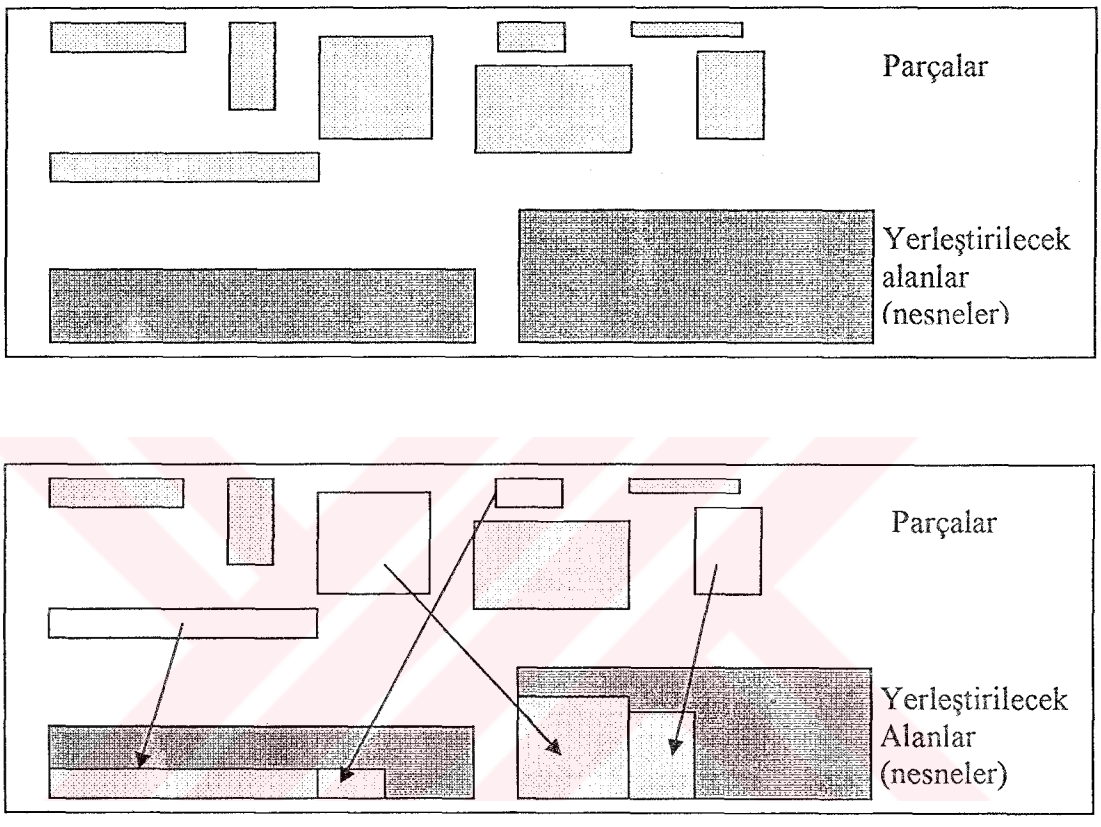
Kesme ve paketleme problemleri, farklı amaçlar ve kısıtlar içeren, çeşitli endüstrilerde görülmektedir. Gemi yapımı, tekstil, ve deri endüstrisinde rasgele şekillenmiş, düzgün biçimli olmayan parçalar paketlenirken, ahşap, cam ve kağıt endüstrisi temelde düzgün şekillerin kesilmesiyle ilgilidir [22,23]

Kesme ve paketleme problemleri kombinatoryal geometri alanının konusudur. Bu problemlerin gelişimi Şekil 3.2'de gösterilmiştir.




Şekil 3.2 Kesme ve paketleme problemlerinin oluşumu [24]

Kesme ve paketleme problemleri, Şekil 3.3'te görüldüğü gibi, en basit anlamda parçaların belirli bir zemine veya bir alana en iyi yerleşimini inceler. Amaç malzemenin kullanımının maksimum olması ile birlikte atık malzemenin en az olmasıdır.




Şekil 3.3 Yerleştirme problemlerinin işleyişinin temel yapısı [25]

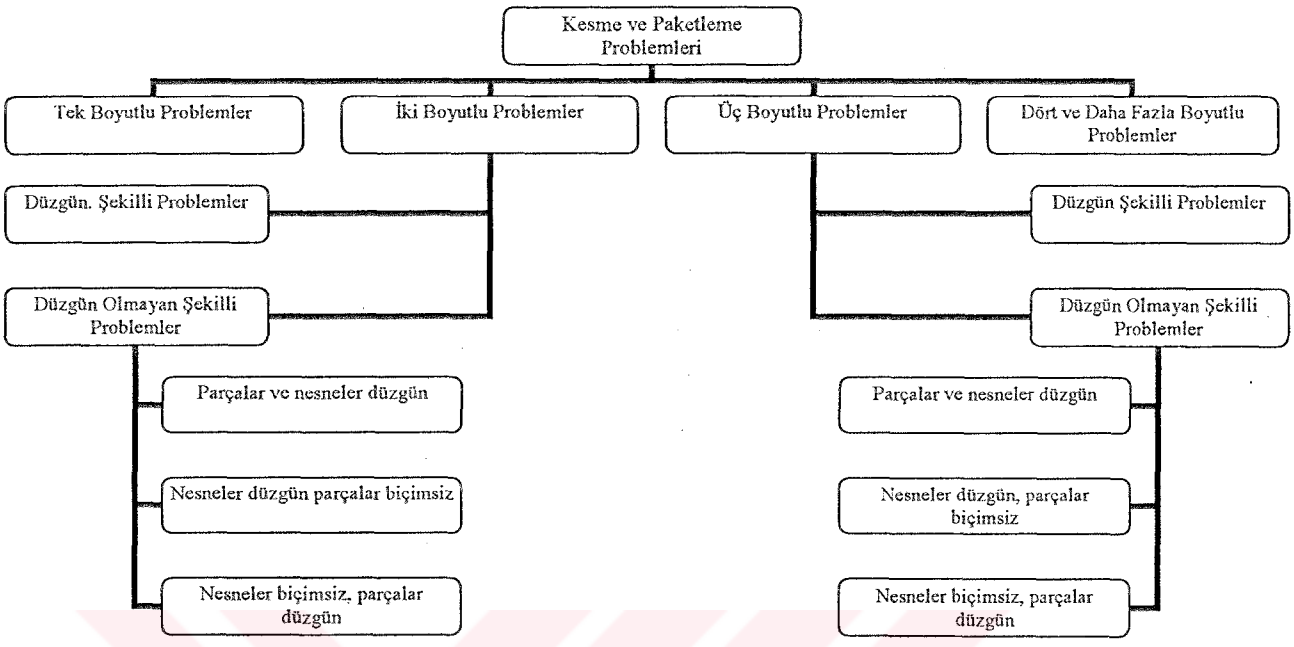
Tekstil ve deri endüstrisinde bu tür problemler yerleştirme problemleri olarak adlandırılırlar. Bu endüstride, otomatik olarak oluşturulan bir şablondan belirli şekillerin kesilerek çıkarılması için kullanılan metodu konu alır. Bu metotlarda amaç atık malzemeyi minimize etmektir. Buna ek olarak yerleştirme problemleri elde edilecek çıktıya ait kabul edilebilir atık malzeme miktarı ile de ilgilenir. Bu yolla şablonun veya yerleştirme planının kalitesi sorgulanabilir ve de elde edilen optimallik derecesinin iyileştirilmesi için çalışılabilir.

Kesme ve paketleme problemleri literatüre en çok Dyckhoff ve öğrencilerinin yaptığı çalışmalarla girmiştir. Dyckhoff, bu problemler için bir sınıflandırma geliştirmiştir. Kesme ve paketleme problemlerinin sınıflandırılmasını 4 kritere göre ayırmıştır [24]. Bunlar;

1. Boyutlanabilirlik
    N) Boyutların sayısı
2. Atama çeşidi
    B) Tüm büyük nesneler ve küçük parçaların seçimi
    V) Büyük nesnelerin ve tüm küçük parçaların seçimi
3. Büyük nesnelerin çeşitleri
    O) Bir büyük nesne
    I) Çok sayıda özdeş büyük nesneler
    D) Farklı boyutlarda büyük nesneler
4. Küçük parçaların çeşitleri
    F) Farklı boyutlarda birkaç parça
    M) Çok farklı boyutlarda çok sayıda parça
    R) Nispeten daha az sayıda boyutlarda çok sayıda parça
    C) Özdeş olan çok sayıda parça

Ahşap ve çelik profil, boru gibi malzemeler için bir boyutlu, metal, cam, kağıt için iki boyutlu, kargo yükleme ve motor montaj uygulamaları için ise üç boyutlu kesme ve paketleme problemleri örnek verilebilir.

Boyutlanabilirlik özelliği bu sınıflandırma kriterlerinden en önemli değere sahiptir. Bu kritere göre kesme ve paketleme problemlerinin sınıflandırılması aşağıdaki Şekil 3.4'te gösterilmiştir.




Şekil 3.4 Kesme ve paketleme problemlerinin boyutlanabilirlik özelliğine göre sınıflandırılması [26]

Çok sayıda problem, Kesme ve Paketleme Problemlerinin mantıksal yapısına benzerdir. Bunlar;

Kesme stok problemleri ve Kesme Kaybı, Kalıp Yerleştirme, Rulo Kesme, Yerleştirme, Paketleme ve Yerleştirme, Kutu Paketleme, Şerit Paketleme, Sırt Çantası Problemleri, Araç Yükleme, Palet Yükleme, Konteynır Yükleme, Kargo Yükleme, Nesting Problemler, Sermaye Bütçeleme, ve Çokişlemcili Çizelgeleme Problemleri'dir [27].

Kesme ve paketleme problemlerinde parçaların ve bunların yerleştirileceği büyük parçaların geometrik şekilleri önemli bir özellik ve bununla bir birlikte kısıt oluşturmaktadır. Tablo 3.1'de geometrik şekillerine göre iki boyutlu paketleme problemlerinin uygulama alanlarından örnekler verilmiştir.

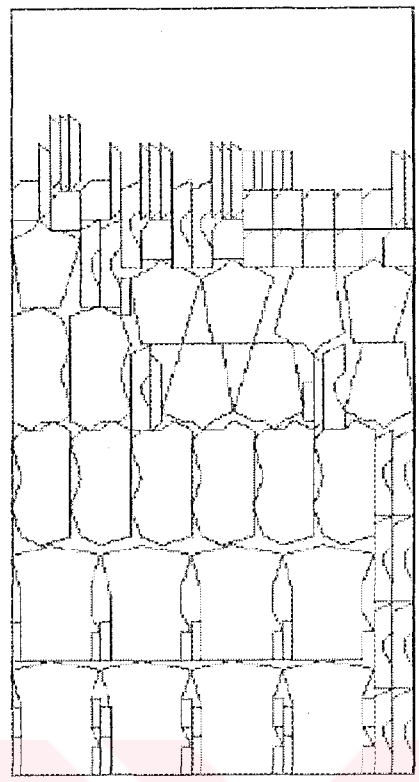
Tablo 3.1 Geometrik şekillerine göre, 2D paketleme problemleri örnekleri [25]

| **Geometrik şekil** | **Örnek uygulama** |
|---|---|
| Daire, elips | Boruların konteynıra yüklenmesi |
| Dikdörtgenler | Kağıt endüstrisi, palet yükleme |
| Çokgenler | Metal endüstrisinde düzgün biçimli olmayan şekillerin paketlenmesi |
| Serbest form | Tekstil endüstrisinde kalıpların yerleştirilmesi |
| Tamamıyla veya bir kısmı kapalı şekiller | Gemi yapımı |

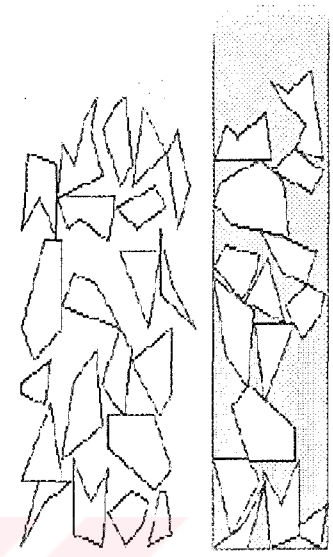
Yerleştirilecek büyük parçanın veya diğer parçaların düzgün geometrik biçime sahip olmadığı kesme ve paketleme problemlerine **Nesting Problemleri** denir. Şekil 3.5'te bu tür problemlere deri endüstrisinden bir örnek verilmektedir.

Şekil 3.5 Deri mobilya üretiminde kullanılan parçalar için yapılmış bir yerleştirme [28]

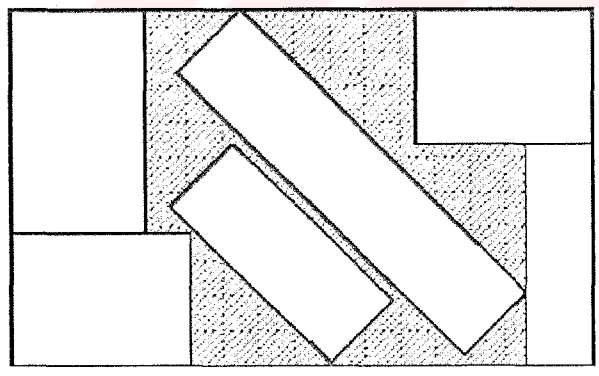
Kesme ve paketleme problemlerinde, parçaların geometrik durumlarına göre yerleştirme düzenleri ve ait oldukları sınıflar aşağıdaki şekillerde görülmektedir [1].



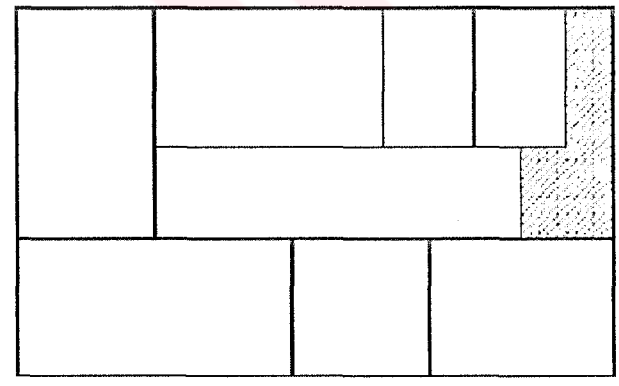
Şekil 3.6 Tekstil endüstrisinde düzgün olmayan paketleme problemi [1]



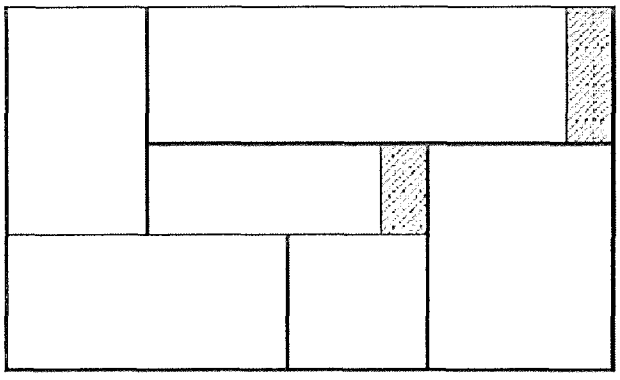
Şekil 3.7 2D düzgün olmayan şerit paketleme problemi [1]



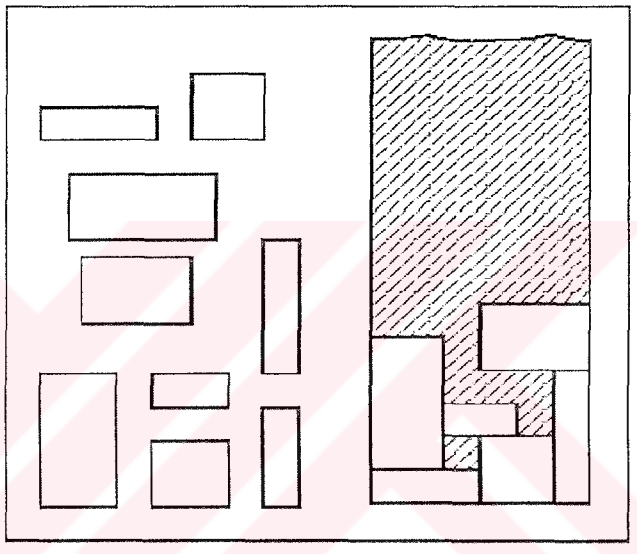
Şekil 3.8 Ortagonal olmayan yerleştirme [1]



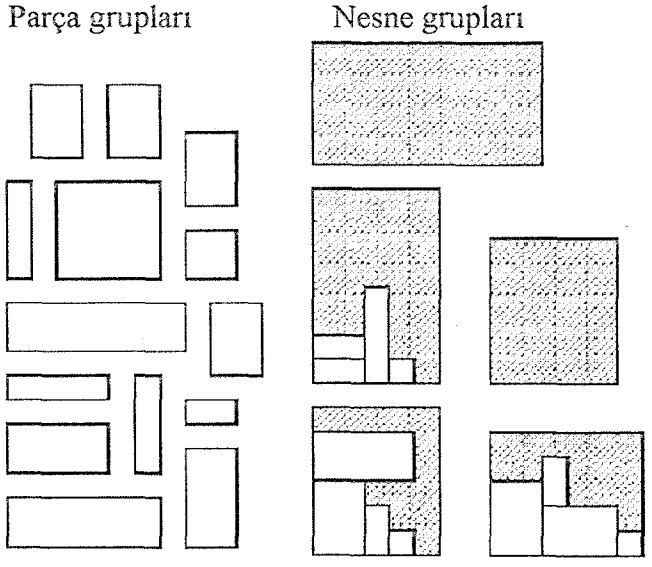
Şekil 3.9 Giyotinli yerleştirme [1]



Şekil 3.10 Giyotinsiz yerleştirme [1]



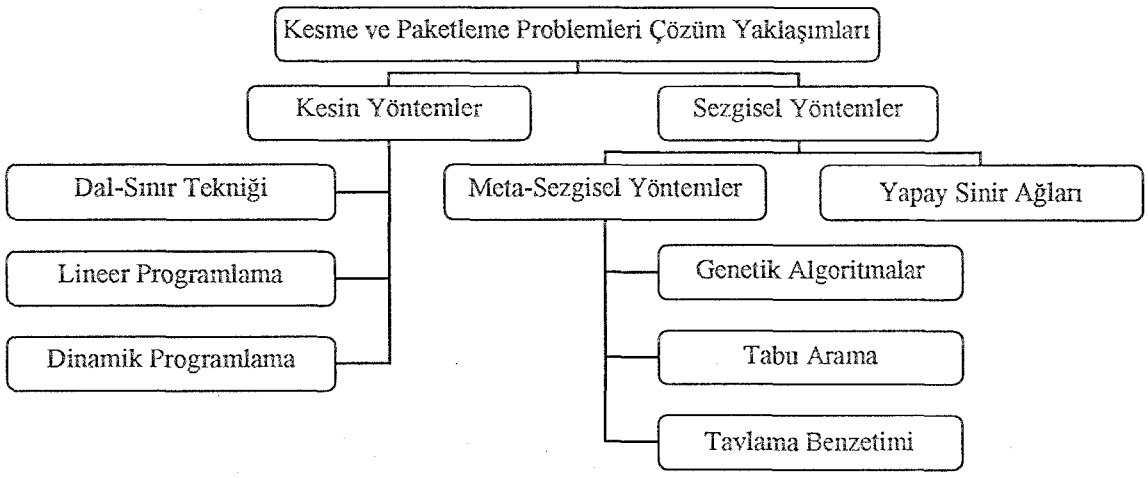
Şekil 3.11 2D şerit paketleme problemi [1]




Şekil 3.12 2D kutu paketleme problemi [1]

## 3.3 Kesme ve Paketleme Problemleri Çözüm Metotları

Kesme ve paketleme problemleri, küçük bir çalışma uzayı içinde olması durumu hariç, bu problemlere en iyi çözümün üretilmesinin imkansız olduğu NP-Complete problemler olarak bilinir. Probleme farklı eniyi çözümlerin bulunması için, büyük arama uzayı içinde düzenli bir arama gereklidir. Bu problemlerin çoğu NP-complete olduğundan, yönlendirilmemiş bir arama oldukça verimsiz olur. Hesaplama zamanının büyüklüğü yüzünden kesin yöntemler bu tür problemlerde yetersiz ve etkisiz kalabilmektedir. Bu nedenle araştırmalar, en iyi çözüme yakın iyi çözümleri verimli bir şekilde bulan sezgisel teknikler üzerinde yoğunlaşmaktadır. Kesme ve paketleme problemleri için uygulanabilir bu sezgisel teknikler arasında Aşağı-Sol (Bottom-Left) Algoritması, meta-sezgisel tekniklerden arasında, tavlama benzetimi, tabu arama ve genetik algoritmaları gibi olasılıksal tabanlı teknikler yer almaktadır [29]. Kullanılan sezgisel bir tekniğin başka bir problemde aynı etkinliği göstermesi beklenemez. Bu yüzden farklı problem konfigürasyonları için farklı sezgisel teknikler geliştirilmiştir. Bu gruplardan, problemlerin sınıflandırılması kısmında bahsedilmiştir. Kesme ve paketleme problemlerinin çözümünde kullanılan teknikler Şekil 3.13'de gösterilmiştir.




Şekil 3.13 Kesme ve paketleme problemleri çözüm yaklaşımları

Meta-sezgisel yöntemlerden ve uygulama alanlarından kombinatoryal optimizasyon bölümünde bahsedilmiştir. Kesme ve paketleme problemleri için geliştirilen sezgisel tekniklerden birisi de, Aşağı-Sol algoritmasıdır. Bu algoritmanın çalışma şekli ise aşağıda açıklanmıştır.

### 3.3.1 Aşağı-Sol Algoritması

1996 yılında Jakobs tarafından tanımlanan Aşağı-Sol yerleştirme algoritması, her parçanın, yerleştirildiği zeminin mümkün olduğunca aşağıya ve soluna hareket etmesini içermektedir. Herhangi bir dikdörtgen parça, yerleşeceği büyük parça üzerinde daha aşağıya veya sola hareket ettirilemiyorsa, yerleşim düzeni BL koşulunu gerçekleştiriyordur. Bir paketleme veya kesme yerleşim düzeni bir permütasyonuyla gösterilebilir [23,29]

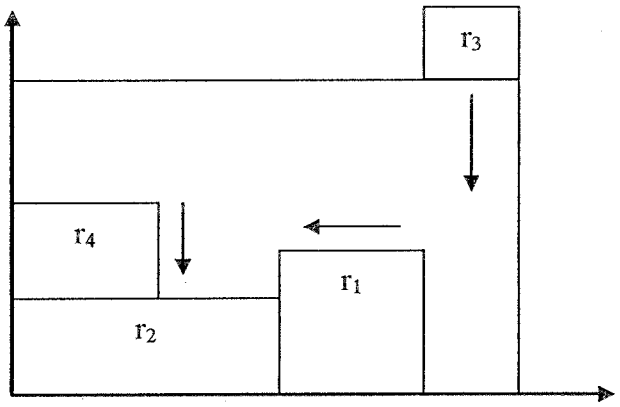
$\pi = (i_1,...,i_n)$ Permütasyon

i, dikdörtgen parça sırası ($r_i$)

Permütasyon yerleştirilecek parçaların sırasını gösterir. BL algoritması çalışırken aşağıdaki adımları izler.

Adım 1. $r_{\pi(1)}$ parçası büyük parçanın en sol alt köşesine yerleştirilir.

Adım i. Verilen permütasyon sırasıyla, yerleştirilecek parça büyük parçanın sağ üst köşesinden başlayarak mümkün olduğunca aşağıya ve sola doğru kaydırılır. Öncelik her zaman için aşağıya doğru kaydırmaya verilir.




Permütasyon:

$\pi(1) = 2$
$\pi(2) = 1$
$\pi(3) = 4$
$\pi(4) = 3$

veya $\pi = 2,1,4,3$

Şekil 3.14 BL algoritmasının gösterimi [29]

Şekil 3.14'e göre, $r_2$ parçası öncelikle en alta daha sonra mümkün olduğunca en sola doğru kaydırılarak yerleştirilir. $r_1$ ve $r_4$ parçaları da aynı şekilde yerleştirilir. Şekil 3.14'te gösterilen oklar en iyi yerleşim için $r_3$ parçasının hareket yönünü göstermektedir [29].

# 4. UYGULAMA

## 4.1 İki Boyutlu Düzgün Dikdörtgensel Parçalar İçin Oluşturulmuş Sezgisel Bir Algoritma

Bir kesme problemi, küçük parçaların yerleştirileceği büyük parçanın kullanılabilirliğinin arttırılmasını böylelikle kullanılmayan alanın en aza indirilmesini hedeflemektedir.

İki boyutlu dikdörtgensel parçaların, yine dikdörtgensel büyük bir parça üzerine yerleşimi söz konusu olduğunda en genel anlamda iki kısıt oluşmaktadır.

Yerleştirilme yapılacak büyük parçanın boyutları $X$ ve $Y$, n adet yerleştirilecek parçanın boyutları da $x$ ve $y$ olarak temsil edildiğinde;

Küçük parçaların alanları, $a_i = x_i \times y_i \qquad i=1, 2, 3, \ldots, n$

Büyük parçanın alanı $A = X \times Y$ olarak belirlenecektir.

Bu durumda;

$$x_i \ \& \ y_i \leq X \ veya \ Y, \qquad \forall i \tag{4.1}$$

$$\sum_{i=1}^{n} a_i \leq A \tag{4.2}$$

(4.1) numaralı denklem ile parçaların verilen zemine yerleşip yerleşemeyecekleri tespit edilir. Burada her veri grubu için toplam dört seçenek söz konusudur. Bu dört seçenekten en az üçünün doğru olması, o parçanın zemine yerleştirilmek için uygun olduğunu göstermektedir.

Bu kısıtlar doğrultusunda amaç fonksiyonu aşağıdaki gibi belirlenebilir.

$$\min. \ Z = A - \sum_{i=1}^{n} a_i$$

Burada optimal çözüm Z değerinin 0 olmasıdır. Bu, tüm parçalar, verilen büyük parçanın üzerine yerleştirilmiş olduğu anlamına gelmektedir. Fakat her zaman optimal çözüm bulunamayabilir. Çünkü rasgeleliğin matematiksel formülü yoktur. Bu durumda Z değeri 0'dan büyük bir değer olacaktır. Buradan hareketle, $\left(1-\frac{Z}{A}\right)\times 100$ değeri yerleşimin verimini gösterecektir.

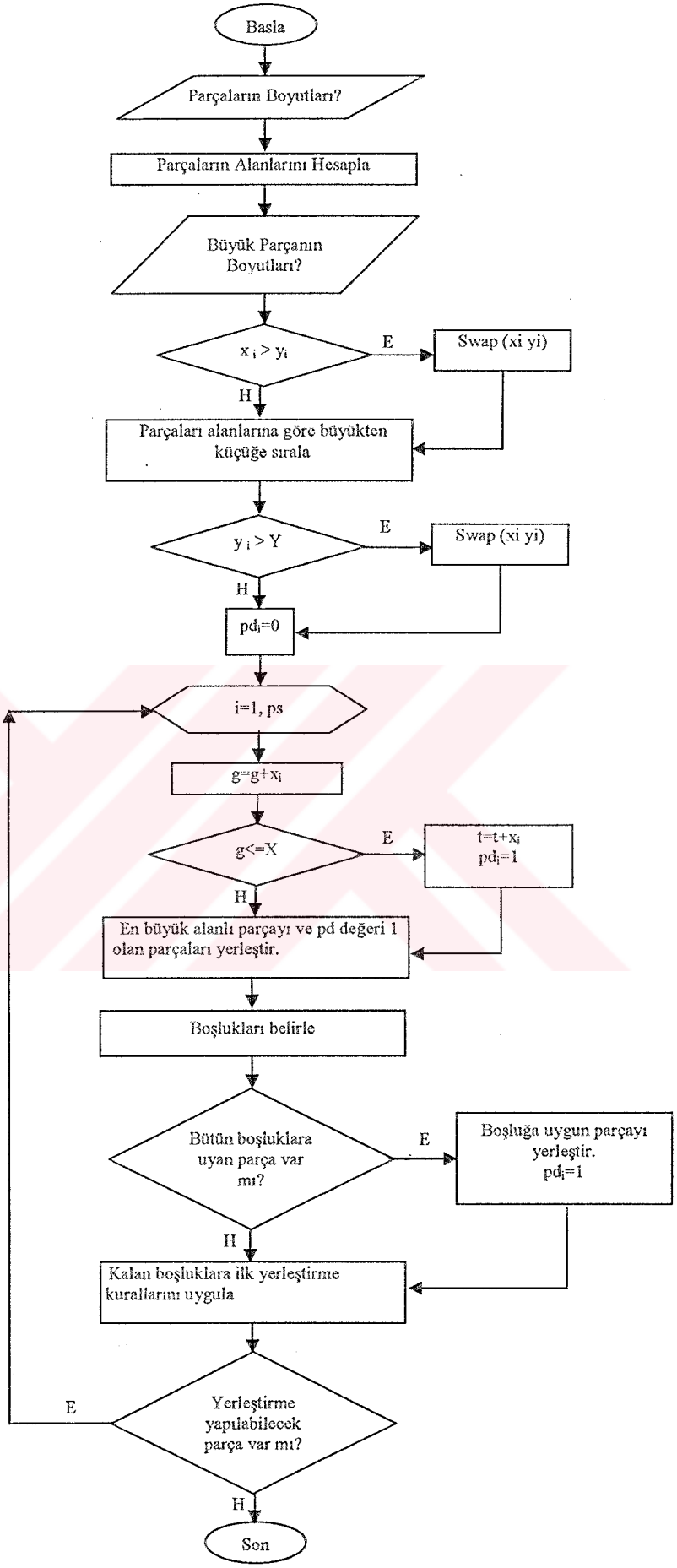
Bu amaç fonksiyonu ve kısıtlar göz önüne alınarak, bu tür problemler için yeni bir algoritma geliştirilmiştir. Algoritmaya ait akış çizelgesi şekil de görülmektedir. Akış çizelgesinde kullanılan bazı kısaltmaların anlamı aşağı gösterildiği gibidir.

$pd_i$ = Parça değeri,

ps = Parça sayısı.

Başlangıçta her parçanın değeri sıfırdır. $pd_i$ =1 olması durumu ise o parçanın yerleştirildiği anlamına gelmektedir.

Bu algoritmanın kodlaması, Matlab 6.5 programında yapılmıştır. Çözüm için parça sayısını ve boyutlarını temsil eden matris oluşturulmuştur. Bu algoritma, optimum çözümü mevcut olan problemler üzerinde denenmiştir. Algoritmanın akış diyagramı Şekil 4.1'de verilmiştir.




Şekil 4.1 Algoritmanın akış diyagramı

## 4.2 Algoritmanın Adımlarının Açıklaması

Algoritmanın satır işlemleri için dört ayrı alternatif vardır. Başlangıçta seçilen bir alternatif kuralı, tüm satırlarda aynı şekilde uygulanır. Bu şekilde parçalar, aynı anda çevrilmiş olur. Bu alternatifler;

| Alternatif | Büyük parça | Küçük parçalar |
|---|---|---|
| 1. | Uzun Kenar $X$ | Kısa Kenarlar $x$ |
| 2. | Uzun Kenar $X$ | Uzun Kenarlar $x$ |
| 3. | Kısa Kenar $X$ | Kısa Kenarlar $x$ |
| 4. | Kısa Kenar $X$ | Uzun Kenarlar $x$ |

Bu alternatiflerin oluşturduğu kısıtlamalar ile her bir parçanın rastgele sırada 90 çevrilmesiyle oluşacak alternatif çözümler azaltılmış olacaktır.

Şekil 4.1'de belirtilen akış diyagramı, 1. alternatife aittir. 2. alternatif, yalnızca, küçük parçaların uzunluklarının karşılaştırıldığı adımda eşitsizliğin yön değiştirmesi ile elde edilebilir. 3. ve 4 . alternatif ise, 1. ve 2. alternatiflerin, veri girişinde büyük parçanın kısa kenarının $X$ olarak girilmesiyle türetilmiştir. 1. alternatif için algoritmanın açıklaması aşağıdaki gibidir.

İlk veri grubu olarak küçük parçaların sayısı ve boyutları girilmesiyle *nx3* boyutunda bir matris oluşturulur. Küçük parçaların alanları hesaplanarak matrise ilave edilir. Küçük parçaların boyutları ile karşılaştırma yapılabilmesi için büyük parçanın boyutları da matrise eklenir.

Büyük parçanın boyutlarında daha büyük olan $X$ olarak seçilirken, küçük parçaların kısa kenarları $x$ boyutu olarak seçilmektedir. Gerekli kontroller yapıldıktan sonra küçük parçaların boyutları kendi içlerinde yer değiştirirler. Bu kısıtın konulmasının nedeni, parçaların döndürülmesi ile oluşacak yerleştirme olasılıklarını azaltmaktır.

Parçalar alanlarına göre büyükten küçüğe doğru sıralanırlar. Tüm parçaların parça değeri 0 olarak matrise eklenir. En büyük alanlı ilk parça, büyük parçanın sol

alt köşesine yerleştirilir. büyük alanlı parçaların öncelikle yerleştirilmesindeki amaç, yerleştirilemeyen parçaların kalması durumunda bunların daha küçük alanlı parçalar olmasını sağlamak ve böylece yerleştirme verimini arttırmaktır.

İlk parça yerleştirildikten sonra, alan sırasına göre diğer parçalar, büyük parçanın $X$ boyutunu göre kontrol edilerek, onun sağına yerleştirilirler. Yerleştirilen parçaların $x$ boyutlarının toplamı, büyük parçanın $X$ boyutuna eşit olduğunda ilk satıra yerleştirilmesi tamamlanmış olur ve yerleştirilen parçaların, parça değeri 1 olarak değiştirilir.

İlk satıra yerleştirilen parçaların, büyük parça üzerinde oluşturduğu koordinatlardan yararlanılarak boşluklar tespit edilir.

Boşluk doldurma kısmında iki durum üzeride durulur. Öncelikle oluşabilecek tüm boşluklar tespit edildikten sonra bu boşluklara birebir uyan parçaların olup olmadığına bakılır. Tüm boşluklara birebir uyan parçalar yerleştirilip, bu parçaların da parça değerleri 1 değerini alır. Burada boşluk sayısına göre, çeşitli kombinasyonlarda yerleşim sözkonusu olacaktır. Yine de bu ilk durum ikinci duruma göre bilgisayar ortamına aktarılması daha kolay bir durumdur.

İkinci durum ise, tüm boşluklara uyan parçaların olmaması ya da boşlukların birkaçının dolmasıdır. Burada doldurulamayan boşluklar için parçaların birleştirilmesi çözüm sağlayabilir. Bundan dolayı kalan boşlukların boyutları işlemin ilk başında olduğu gibi yerleştirilme yapılacak büyük parça olarak alınıp, uygun parçalar yerleştirilir. Algoritmanın başında oluşturulan mantık, boşluk doldurma işleminde gerektiğinde yeniden çalıştırılmak durumundadır. Burada içiçe geçmiş bir yapı görülmektedir. Boşluk doldurma kısmında karşılaşılan iki durum Tablo 4.1'de gruplandırılmıştır.

Tablo 4.1 Boşluk doldurma sırasında karşılaşılan durumlar

| | **Boşluk doldurma** |
|---|---|
| A Durumu | Tüm boşlukları temsil eden parçaların bulunması |
| B Durumu | Doldurulamayan boşlukların yeni bir büyük parça olarak belirlenip, kalan parçalarla yeni yerleştirme |

İlk satır işlemleri boşluk doldurma işlemleri ile tamamlandıktan sonra kalan parçalar ile ikinci satır işlemlerine geçilmektedir. Burada da ilk satırda uygulanan işlemler aynı sırada yürütülür. Bu işlemler yerleştirilebilecek parça kalmayınca kadar sürmektedir.

Parçaların döndürülmesi ile yerleşim için çok fazla seçenek olacağından, parçalar sadece boşluk doldurma işleminde döndürülecektir. Ancak küçük parçaların ayrı ayrı döndürülmesi yerine bir grup olarak döndürülmesi ve büyük parçanın boyutlarının yerinin değiştirilmesi ile çözüm için dört farklı başlangıç durumu elde edilir. Bu da yerleşim alternatiflerini algoritmanın başında dörde çıkarmaktadır.

## 4.3 Algoritma ile Çözülmüş Örnekler

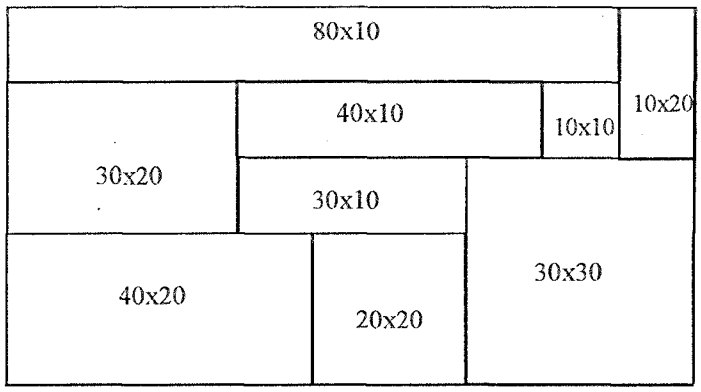
**Örnek 1.** Optimum çözümü bilinen, boyutları birbirinden farklı, 9 adet parça 90x50 boyutundaki bir büyük parçanın üzerine yerleştirilmek istenmektedir.

Parça sayısı : 9
Büyük parçanın boyutları : 90x50
Parçaların boyutları Tablo 4.2'de verilmiştir.

Tablo 4.2 Örnek 1'e ait parçaların boyutları

| Parça no | X | Y |
|---|---|---|
| 1 | 40 | 20 |
| 2 | 20 | 20 |
| 3 | 30 | 30 |
| 4 | 30 | 20 |
| 5 | 30 | 10 |
| 6 | 40 | 10 |
| 7 | 10 | 10 |
| 8 | 20 | 10 |
| 9 | 80 | 10 |

Verilen bu parçalar ile mevcut optimum yerleşim Şekil 4.2'de görülmektedir. Burada amaç algoritmanın bu probleme uygulanarak başka bir optimum çözüm bulup bulamayacağının tespitidir.




Şekil 4.2 Örnek 1'e ait mevcut optimum yerleşim

İlk olarak parça sayısını ve boyutlarını gösteren bir 9x3 boyutunda aşağıdaki gibi bir matris oluşturulur.

A=

| | | |
|---|---|---|
| 1 | 40 | 20 |
| 2 | 40 | 10 |
| 3 | 30 | 30 |
| 4 | 30 | 20 |
| 5 | 30 | 10 |
| 6 | 20 | 20 |
| 7 | 10 | 10 |
| 8 | 20 | 10 |
| 9 | 80 | 10 |

Parçaların alanları hesaplanıp büyük parçanın boyutları girildikten sonra A matrisi aşağıdaki duruma gelmektedir.

A=

| | | | | | | |
|---|---|---|---|---|---|---|
| 3 | 30 | 30 | 900 | 90 | 50 | 0 |
| 9 | 80 | 10 | 800 | 90 | 50 | 0 |
| 1 | 20 | 40 | 800 | 90 | 50 | 0 |
| 4 | 20 | 30 | 600 | 90 | 50 | 0 |
| 6 | 20 | 20 | 400 | 90 | 50 | 0 |
| 2 | 10 | 40 | 400 | 90 | 50 | 0 |
| 5 | 10 | 30 | 300 | 90 | 50 | 0 |
| 8 | 10 | 20 | 200 | 90 | 50 | 0 |
| 7 | 10 | 10 | 100 | 90 | 50 | 0 |

Bu matriste, 1. sütun parçaların numaralarını, 2. ve 3. sütun parçaların boyutlarını, 4. sütun parçaların alanlarını, 5. sütun büyük parçanın $X$ boyutunu, 6. sütun büyük parçanın $Y$ boyutunu, 7. sütun ise parça değerini göstermektedir.

Bu sıralamadan sonra parçaların x boyutu kümülatif olarak, büyük parçanın $X$ boyutuna eşit olana kadar toplanır. Kümülatif olarak x boyutu toplanan parçalar, yerleştirilecek parçalar arasında olacaktır ve bu parçaların, parça değeri 1 olarak değiştirilecektir. Bu durumda A matrisinin yeni durumu aşağıdaki gibi olacaktır ve C matrisi ile temsil edilmektedir.

C =

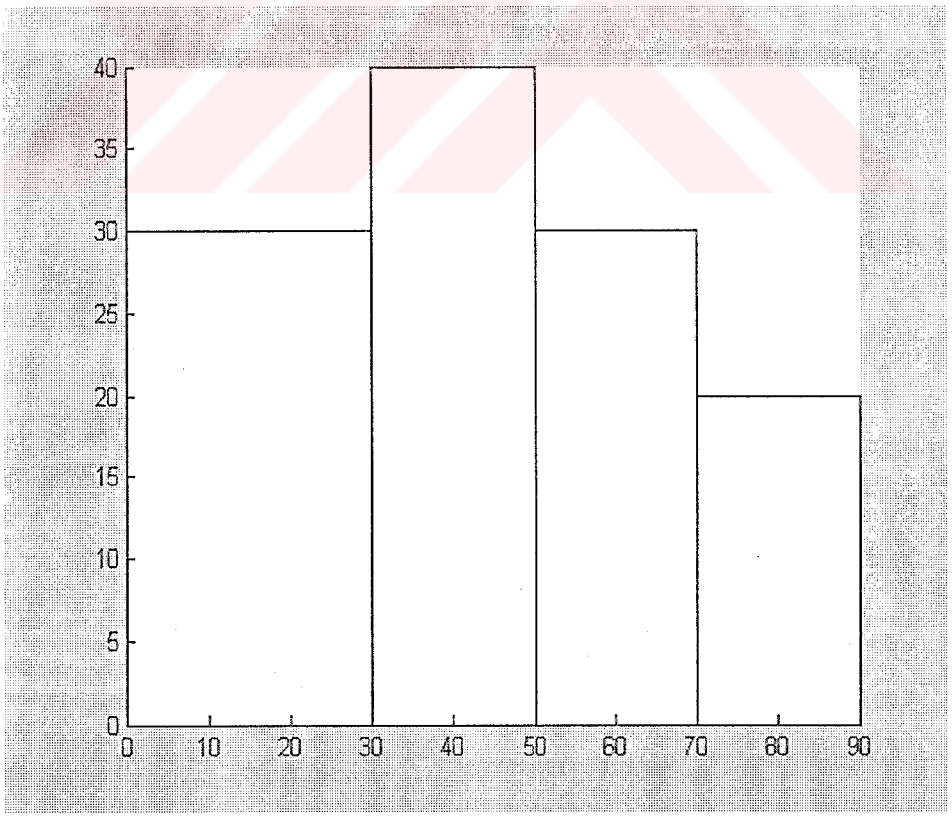
| | | | | | | |
|---|---|---|---|---|---|---|
| 3 | 30 | 30 | 900 | 90 | 50 | 1 |
| 1 | 20 | 40 | 800 | 90 | 50 | 1 |
| 4 | 20 | 30 | 600 | 90 | 50 | 1 |
| 6 | 20 | 20 | 400 | 90 | 50 | 1 |
| 9 | 80 | 10 | 800 | 90 | 50 | 0 |
| 2 | 10 | 40 | 400 | 90 | 50 | 0 |
| 5 | 10 | 30 | 300 | 90 | 50 | 0 |
| 8 | 10 | 20 | 200 | 90 | 50 | 0 |
| 7 | 10 | 10 | 100 | 90 | 50 | 0 |

En büyük alanlı parça 3 numaralı parçadır. Arkadan gelen 800 $br^2$'lik iki parçadan biri olan 9 numaralı parça, ilk satıra yerleştirilmesi uygun olmadığından 0 değerini almıştır. Buradan parça değeri 1 olan parçaların koordinatları belirlenip çizilmesine geçilmektedir. Bu parçaların koordinatlarını gösteren cz matrisi aşağıdaki gibi oluşmaktadır.

cz=

| | | | |
|---|---|---|---|
| 0 | 0 | 30 | 30 |
| 30 | 0 | 20 | 40 |
| 50 | 0 | 20 | 30 |
| 70 | 0 | 20 | 20 |

Şekil 4.3'te ilk olarak yerleştirilen parçaların çizimi yer almaktadır. Yerleştirilen parçalar içinde en büyük y boyutu 40 br olduğundan, büyük parçanın *Y* boyutu da 40 br şeklinde gözükmektedir. Fakat yerleştirme işlemi ilerlediğinde büyük parçanın boyutları gerçek boyutlarını alacaktır.

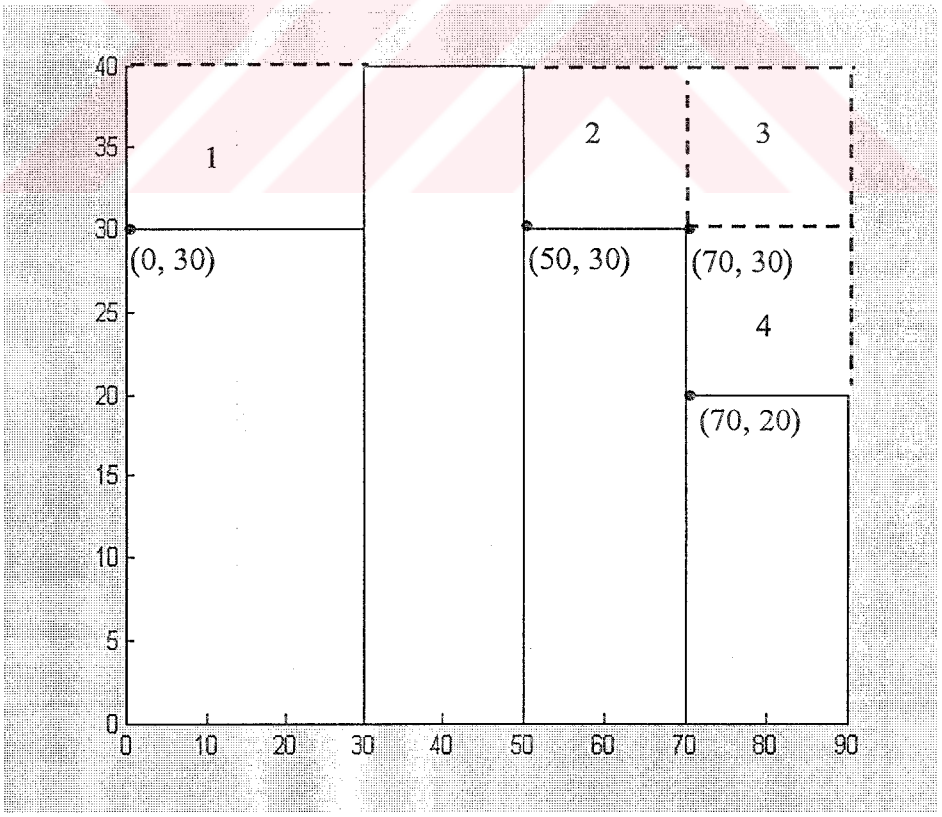



Şekil 4.3 Örnek 1.'de ilk yerleştirilen parçaların çizimi

Bu aşamadan sonra algoritmaya göre oluşan boşluklar tespit edilecektir. Boşluklar için belirlenen tüm koordinatlar aşağıda verilmiştir.

0 30
30 30
30 40
50 40
50 30
70 30
70 20
0 20

Fakat Şekil 4.4'te görüldüğü gibi, boşluk doldurma işlemi sırasında 4 nokta yerleştirilecek yeni parçalar için orjin noktası olabilecektir. Bu koordinatlar, (0, 30), (50, 30), (50, 20) ve (70, 30) noktalarıdır.




Şekil 4.4 Örnek 1.'de boşluk doldurma işlemi için koordinatların durumu

Boşluk doldurma sırasında oluşan alternatifler şöyledir.

(0,30) noktası için 1 numaralı alan ile gösterilen 30x10 boyutunda bir parçanın yerleşimi söz konusudur. (50,30) noktası için iki alternatif vardır. Birincisi, 20x10 boyutunda 2 numara ile gösterilen alana sahip parça, ikincisi ise 2 ve 3 numaraları alanların toplamından oluşan 40x10 boyunda parçadır. (70, 30) noktası için tek bir durum söz konusudur. O da 20x10 boyutunda bir parça ile 3 numara ile gösterilen alanın doldurulmasıdır. (70, 20) noktası için de 2 alternatif söz konusudur. 20x10 boyutunda 4 numaralı alanın doldurulması ve 20x20 boyutunda 3 ve 4 numaralı parçaların toplam alanlarının doldurulmasıdır. Noktaları ve oluşan alan alternatiflerini özetleyecek olursak,

| **Koordinat** | **Alan** |
|---|---|
| (0, 30) | 1 |
| (50, 30) | 2, (2+3) |
| (70, 30) | 3 |
| (70,20) | 4, (4+3) |

şeklinde gösterilebilir. Algoritmada tüm bu alternatifleri boşlukları tespit etmek durumundadır.

(2+3), 3 ve 4 alanlarını temsil eden matris,

50 30 40 10
70 30 20 10
70 20 20 10

1, 2, 3 ve (4+3) alanlarını temsil eden matris ise

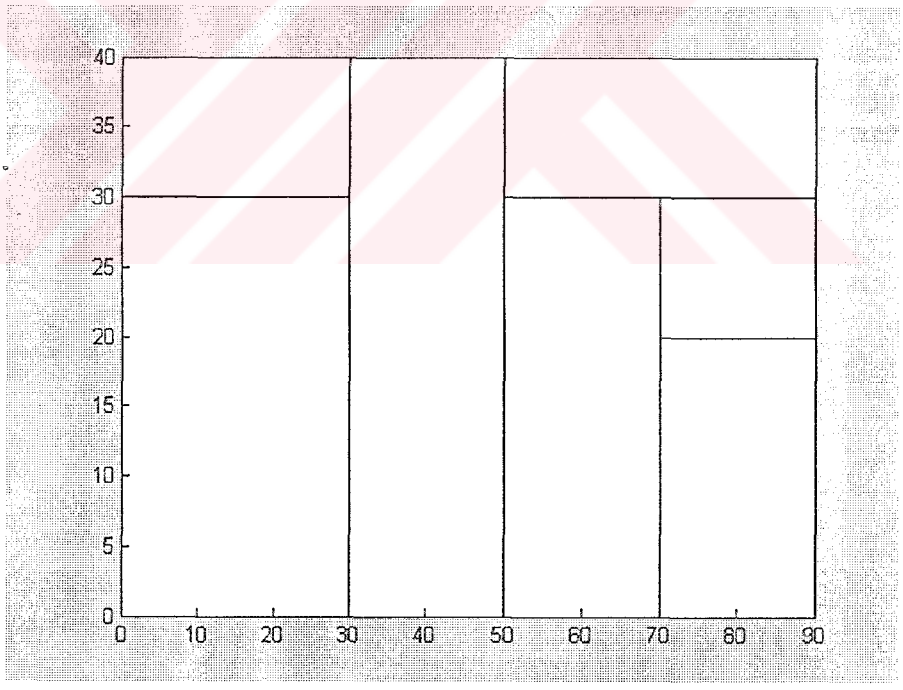
0 30 30 10
50 30 20 10
70 30 20 10
70 20 20 20 , şeklindedir.

Bu matrislerin birleşimi oluşan her boşluğa yerleştirilebilecek parçanın boyutunu ve koordinatlarını göstermektedir. Yerleştirilmemiş parçalar ile oluşan boşlukların

karşılaştırılması yapılır. Boşluklara birebir uyan parçalar bulunduğunda bu parçalar ve koordinatları başlangıçta 1. satır elemanlarını yerleştirme için kullanılan çizim matrisine eklenir. Sonuçta aşağıdaki gibi yeni bir matris elde edilir.

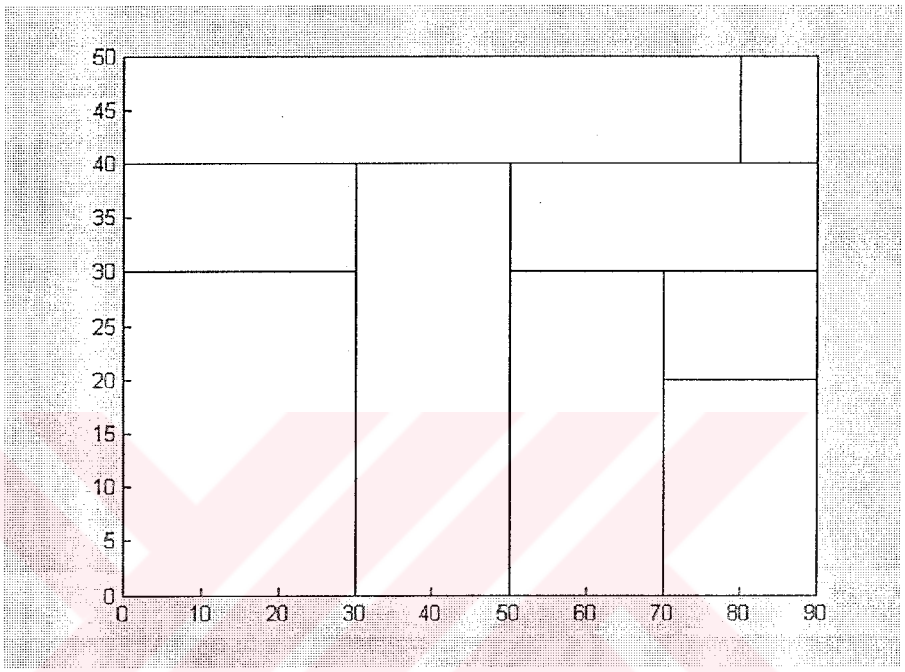
| | | | |
|---|---|---|---|
| 0 | 0 | 30 | 30 |
| 30 | 0 | 20 | 40 |
| 50 | 0 | 20 | 30 |
| 70 | 0 | 20 | 20 |
| 50 | 30 | 40 | 10 |
| 0 | 30 | 30 | 10 |
| 70 | 20 | 20 | 10 |

Bu matris 1. satır işlemlerinin tamamlandığını göstermektedir. Yeni parçaların eklenmesi ile Şekil 4.5'te görülen bir yerleşim oluşur.



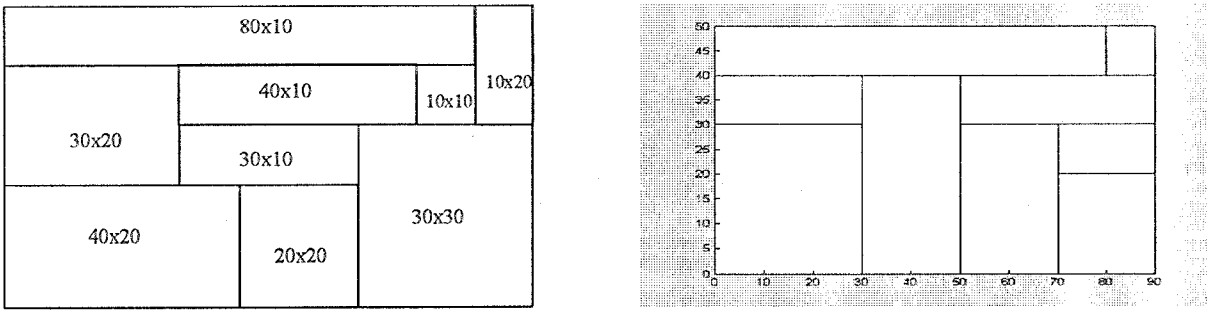

Şekil 4.5 Örnek 1.'de 1. satırın tamamlanmış yerleşimi

Bu aşamadan sonra kalan parçalar dikkate alınarak 2. satır işlemlerine geçilir. 1. satır için uygulanan kurallar uygulandığında, yerleşim düzeni aşağıdaki Şekil 4.6'da görüldüğü gibi son halini alacaktır.




Şekil 4.6 Örnek 1.'in algoritmaya göre son yerleşimi

Optimum çözümü bilinen bu problemde, geliştirilen algoritmanın da başka bir optimum sonuç ürettiği görülmüştür. İki farklı yerleşim düzeni Şekil 4.7'de gösterilmektedir.




Şekil 4.7 Örnek 1.'in mevcut optimum yerleşimi ile algoritma çözümünün karşılaştırılması

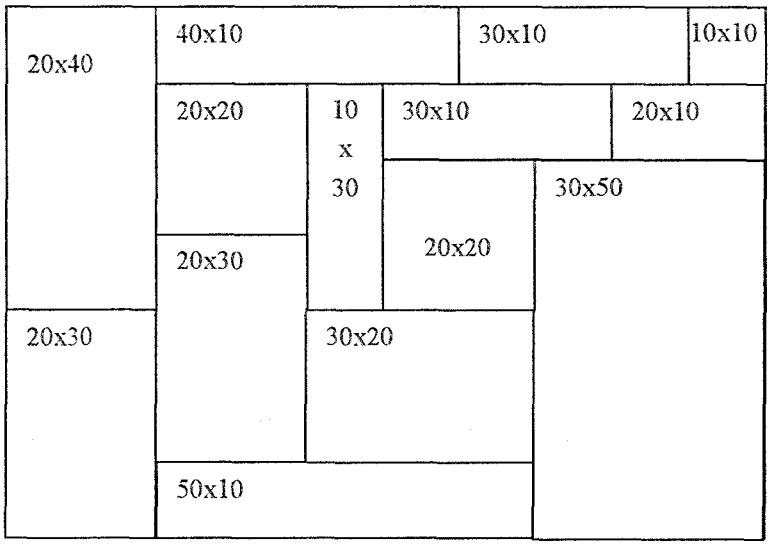
**Örnek 2.**

Parça sayısı : 14

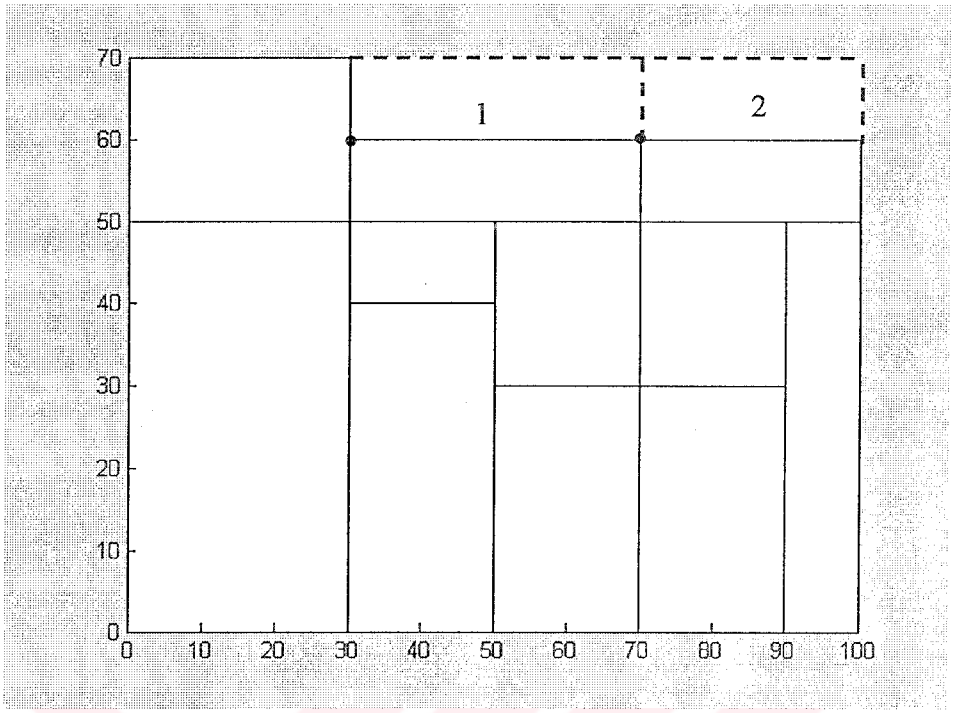
Büyük parçanın boyutları : 100x70

Tablo 4.3 Örnek 2.'ye ait parçaların boyutları

| Parça No | X | Y |
|---|---|---|
| 1 | 20 | 40 |
| 2 | 40 | 10 |
| 3 | 30 | 10 |
| 4 | 10 | 10 |
| 5 | 20 | 20 |
| 6 | 10 | 30 |
| 7 | 30 | 10 |
| 8 | 20 | 10 |
| 9 | 20 | 20 |
| 10 | 30 | 50 |
| 11 | 20 | 30 |
| 12 | 20 | 30 |
| 13 | 30 | 20 |
| 14 | 50 | 10 |




Şekil 4.8 Örnek 2.'ye ait mevcut optimum yerleşim

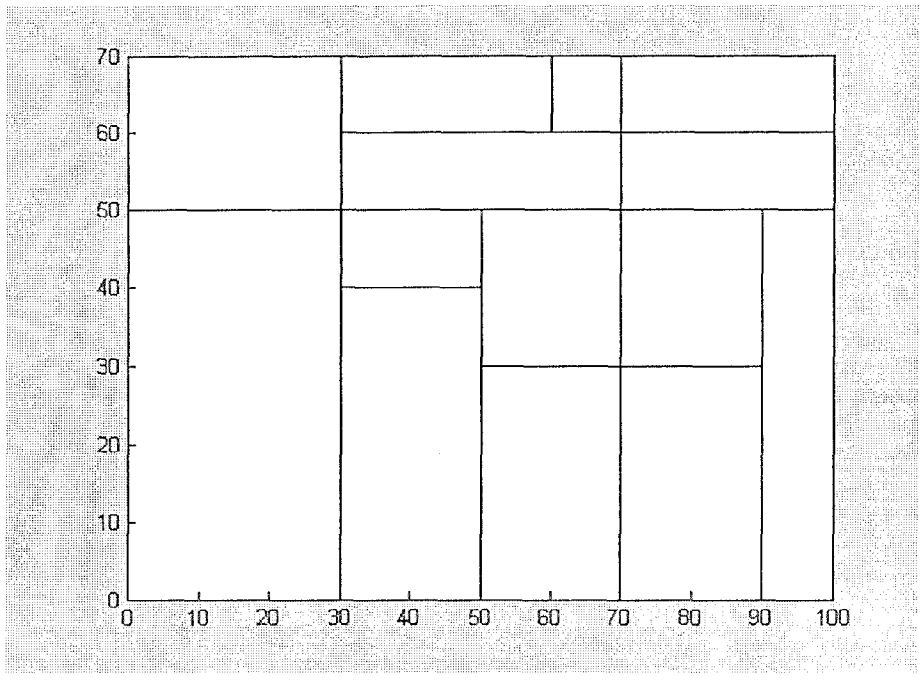



Şekil 4.9 Örnek 2.'nin çözümünde B durumu boşlukların gösterilmesi

Algoritmaya göre, 1. satır işlemleri tamamlanıp, 2. satır işlemlerine geçildiğinde Şekil4.9'daki gibi bir yerleşim görülecektir. Burada oluşan boşluklardan 2 numaralı alan için aynı boyutları taşıyan parça vardır fakat 1 numaralı alan için yoktur. Kalan parçalar ve boyutları aşağıdaki gibidir.

6 30 10 300
3 30 10 300
4 10 10 100

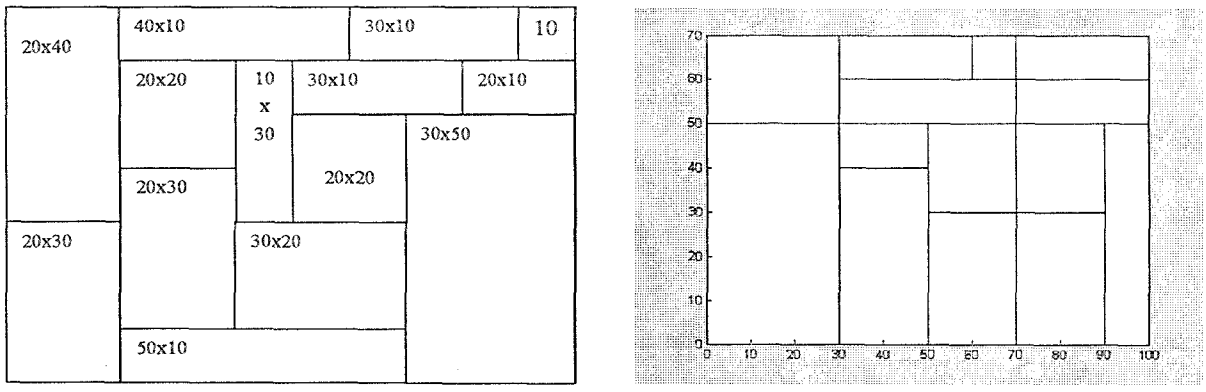
Boşluklara birebir uygun parça bulunmaması durumu algoritmanın alternatif seçeneğin kullanılması gerektiği durumdur. Her boşluk doldurulamıyorsa kalan boşluklar büyük parça gibi kabul edilir ve kalan parçaların aynı kurallarla buralara yerleştirilmesi yapılır. 6 numaralı parça, 2 numaralı alana yerleştirildiğinde, problemin kalan kısmı 30x10 ve 10x10 boyutlarında parçaların 40x10 boyutunda bir zemine yerleştirilmesi halini alacaktır. Bu durumda son çözüm Şekil 4.10'da görüldüğü gibi olacaktır.




Şekil 4.10 Örnek 2.'nin algoritmaya göre son yerleşimi

Fakat algoritmadaki bu ikinci durum bilgisayar ortamına aktarılamamıştır.

Örnek 2 için mevcut ve algoritmaya göre oluşturulan fakat farklı olan çözüm Şekil 4.11'de görülmektedir.



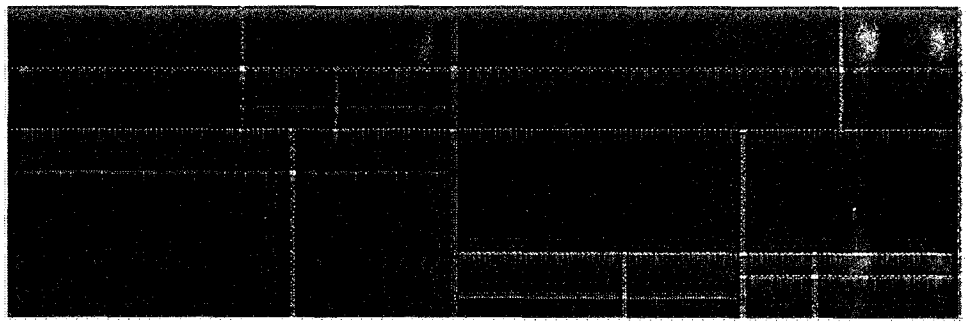

Şekil 4.11 Örnek 2.'nin mevcut optimum yerleşimi ile algoritma çözümünün karşılaştırılması

**Örnek 3.[30]**

Parça Sayısı : 25
Büyük Parçanın Boyutları : 40x15

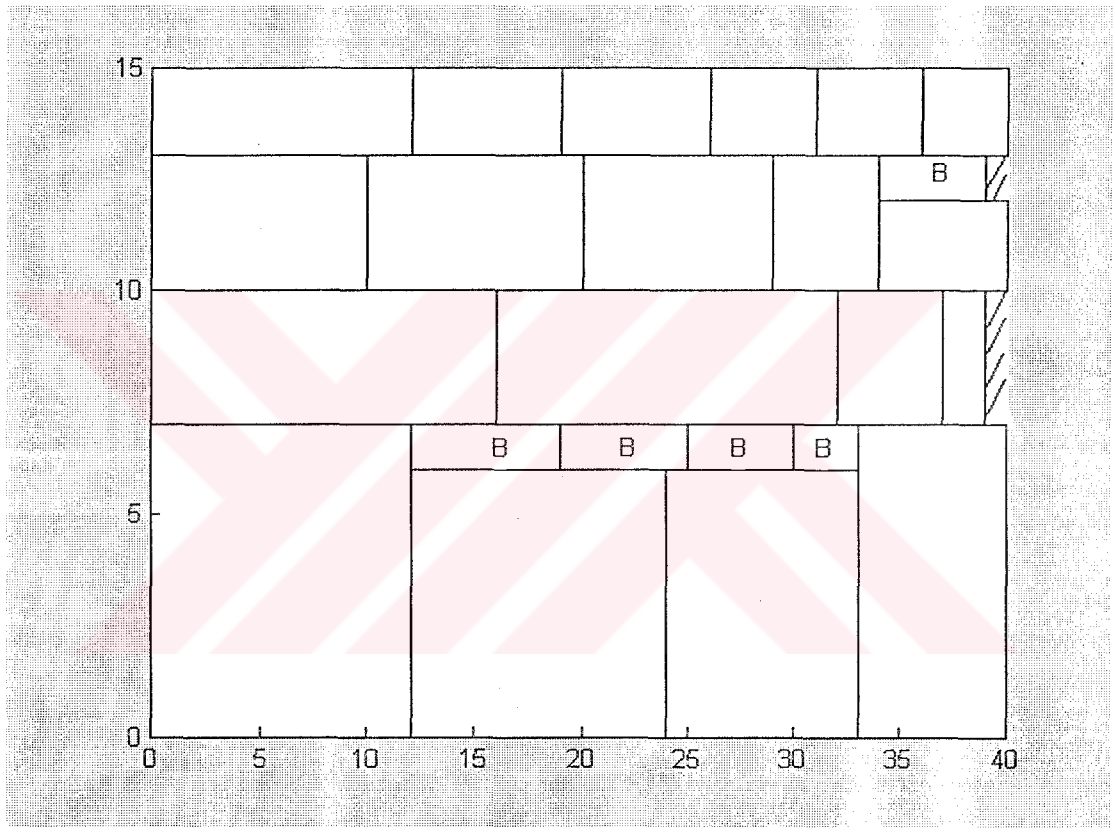
Tablo 4.4 Örnek 3.'e ait parçaların boyutları

| Parça no | X | Y |
|---|---|---|
| 1 | 12 | 7 |
| 2 | 7 | 7 |
| 3 | 7 | 1 |
| 4 | 5 | 1 |
| 5 | 3 | 2 |
| 6 | 6 | 2 |
| 7 | 7 | 2 |
| 8 | 5 | 2 |
| 9 | 3 | 1 |
| 10 | 6 | 1 |
| 11 | 12 | 6 |
| 12 | 9 | 6 |
| 13 | 12 | 2 |
| 14 | 7 | 2 |
| 15 | 10 | 3 |
| 16 | 4 | 1 |
| 17 | 5 | 1 |
| 18 | 16 | 3 |
| 19 | 5 | 3 |
| 20 | 4 | 2 |
| 21 | 5 | 2 |
| 22 | 10 | 3 |
| 23 | 9 | 3 |
| 24 | 16 | 3 |
| 25 | 5 | 3 |

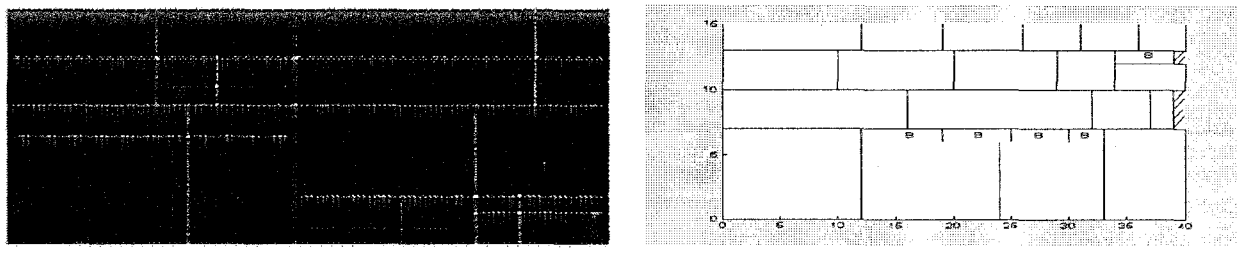


Şekil 4.12 Örmek 3.'e ait mevcut optimum yerleşim [30]

Bu problemin algoritmaya göre en iyi çözümü 2. alternatifle sağlanmış ve 1. ve 3. satır işlemlerinden sonra kalan boşluklarda B durumu ile karşılaşılmıştır. Fakat B durumu için çözüm aşamaları uygulandığında, 1. satırda 1x3 $br^2$'lik , 3. satırda ise 1x1 $br^2$'lik bir alana yerleştirilme yapılamamıştır. Bu durumda yerleştirilen parça sayısı 24, yerleştirilemeyen parça sayısı ise 1'dir. Yerleştirilemeyen parça, 4x1 boyutunda 16 numaralı parçadır. Buna göre Örnek 3.'e ait yerleşim düzeni Şekil 4.13'te görülmektedir.




Şekil 4.13 Örnek 3.'ün algoritmaya göre yerleşimi



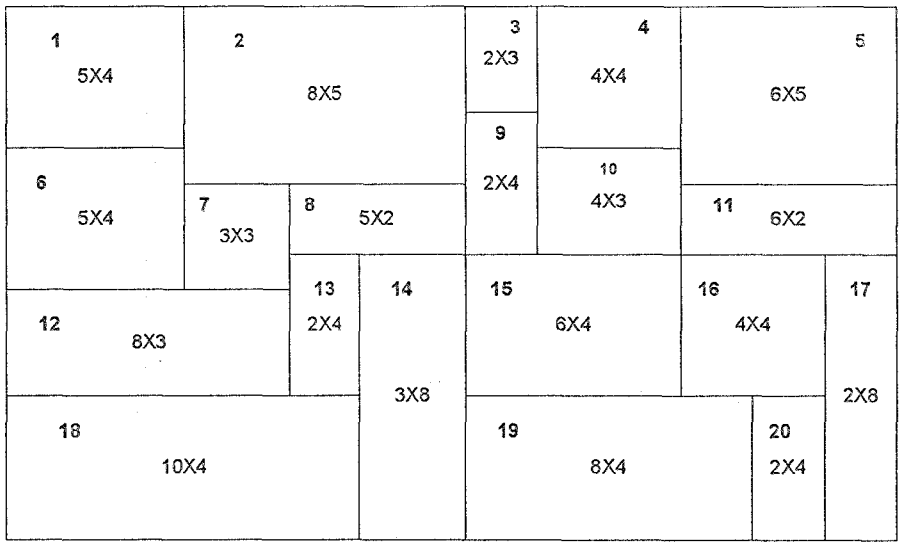
Şekil 4.14 Örnek 3.'ün mevcut optimum yerleşimi ile algoritma çözümün karşılaştırılması

**Örnek 4.**

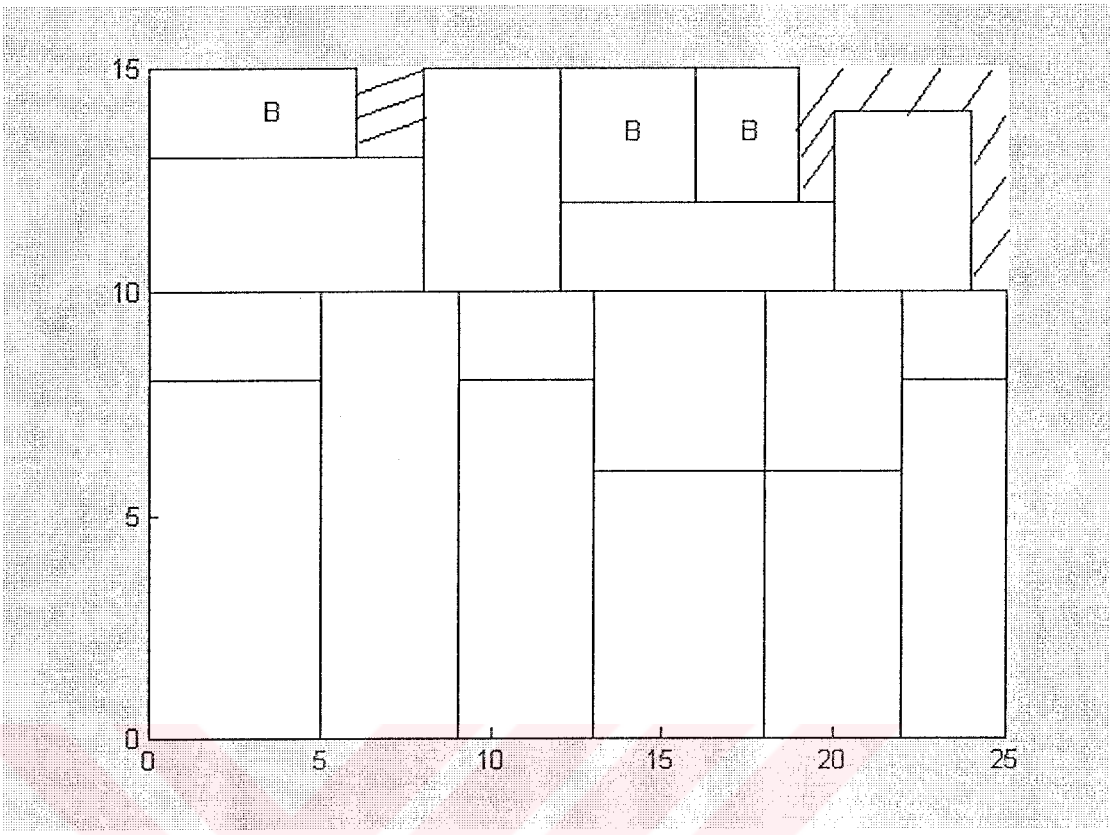
Parça Sayısı : 20
Büyük Parçanın Boyutları : 25x15

Tablo 4.5 Örnek 4.'e ait parçaların boyutları

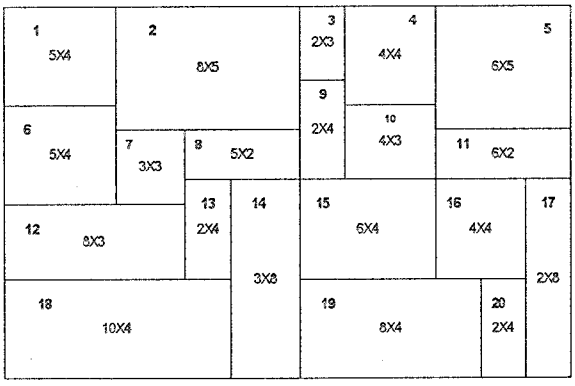
| Parça no | X | Y |
|---|---|---|
| 1 | 5 | 4 |
| 2 | 8 | 5 |
| 3 | 2 | 3 |
| 4 | 4 | 4 |
| 5 | 6 | 5 |
| 6 | 5 | 4 |
| 7 | 3 | 3 |
| 8 | 5 | 2 |
| 9 | 2 | 4 |
| 10 | 4 | 3 |
| 11 | 6 | 2 |
| 12 | 8 | 3 |
| 13 | 2 | 4 |
| 14 | 3 | 8 |
| 15 | 6 | 4 |
| 16 | 4 | 4 |
| 17 | 2 | 8 |
| 18 | 10 | 4 |
| 19 | 8 | 4 |
| 20 | 2 | 4 |

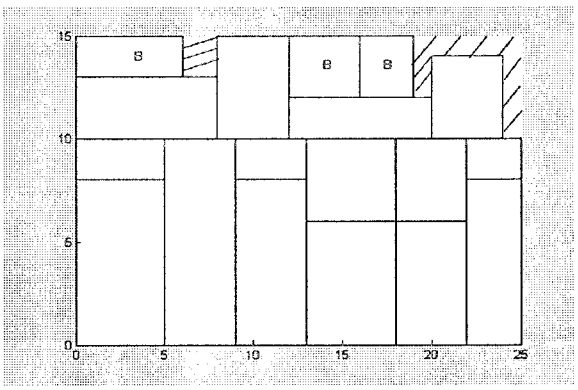



Şekil 4.15 Örnek 4.'e ait mevcut optimum yerleşim




Şekil 4.16 Örnek 4.'ün algoritmaya göre yerleşimi



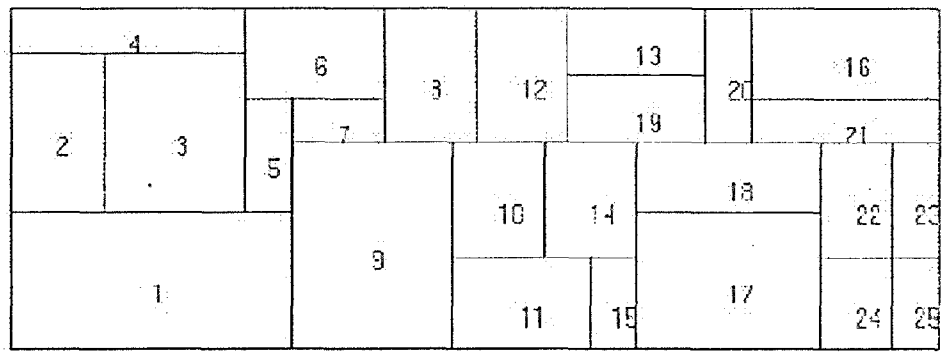

Şekil 4.17 Örnek 4.'ün mevcut optimum yerleşimi ile algoritma çözümünün karşılaştırılması

**Örnek 5. [30]**

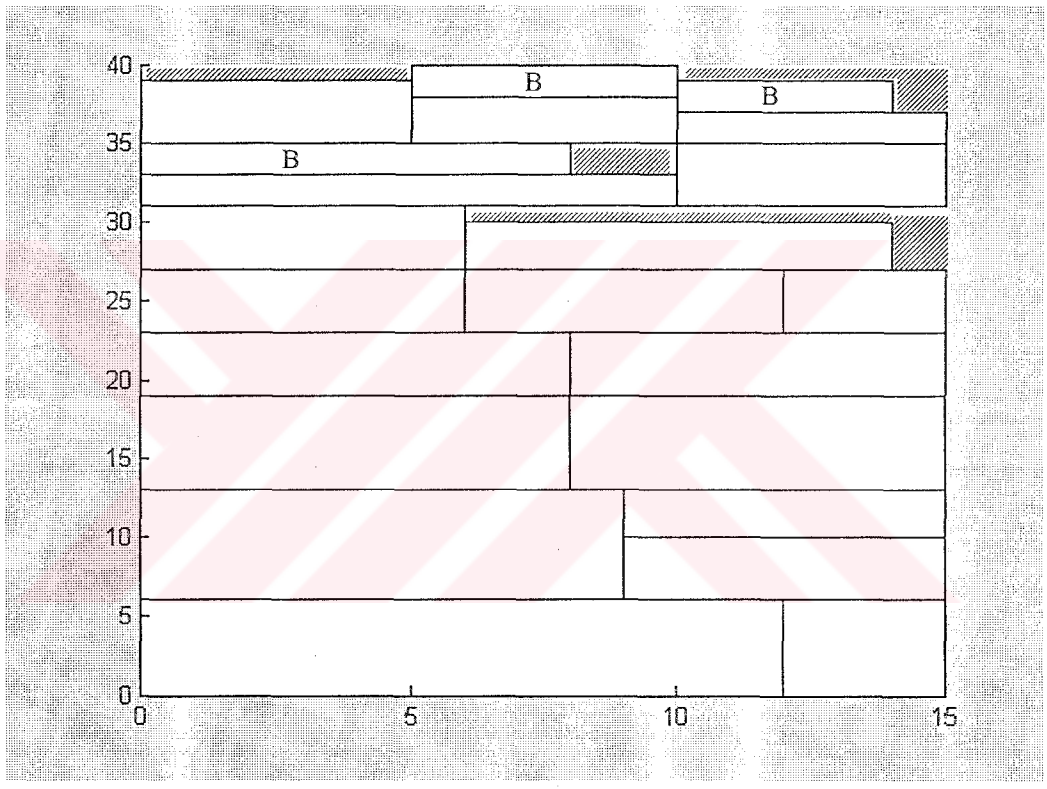
Parça Sayısı : 25
Büyük Parçanın Boyutları : 40x15

Tablo 4.6 Örnek 5.'e ait parçaların boyutları

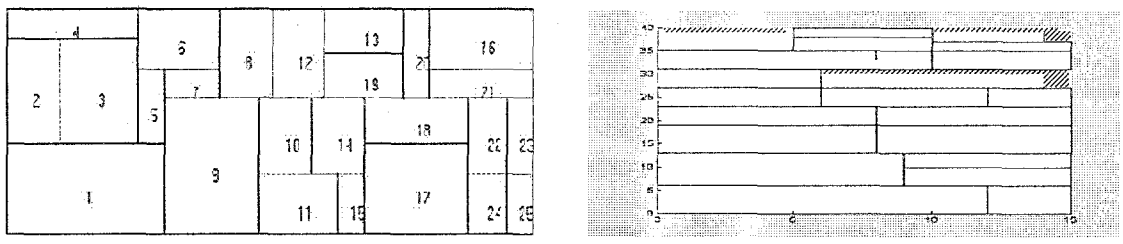
| Parça no | X | Y |
|---|---|---|
| 1 | 12 | 6 |
| 2 | 4 | 7 |
| 3 | 6 | 7 |
| 4 | 10 | 2 |
| 5 | 2 | 5 |
| 6 | 6 | 4 |
| 7 | 4 | 2 |
| 8 | 4 | 6 |
| 9 | 7 | 9 |
| 10 | 4 | 5 |
| 11 | 6 | 4 |
| 12 | 4 | 6 |
| 13 | 6 | 3 |
| 14 | 4 | 5 |
| 15 | 2 | 4 |
| 16 | 8 | 4 |
| 17 | 8 | 6 |
| 18 | 8 | 3 |
| 19 | 6 | 3 |
| 20 | 2 | 6 |
| 21 | 8 | 2 |
| 22 | 3 | 5 |
| 23 | 2 | 5 |
| 24 | 3 | 4 |
| 25 | 2 | 4 |




Şekil 4.18 Örnek 5.'e ait mevcut optimum yerleşim [30]




Şekil 4.19 Örnek 5.'in algoritmaya göre yerleşimi



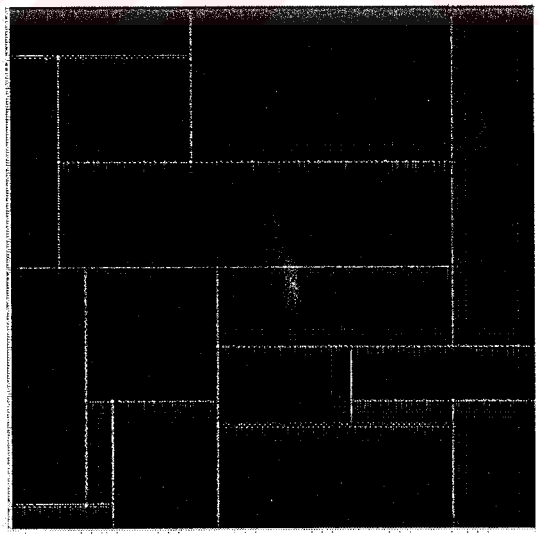
Şekil 4.20 Örnek 5.'in mevcut optimum yerleşimi ile algoritma çözümünün karşılaştırılması

**Örnek 6. [30]**

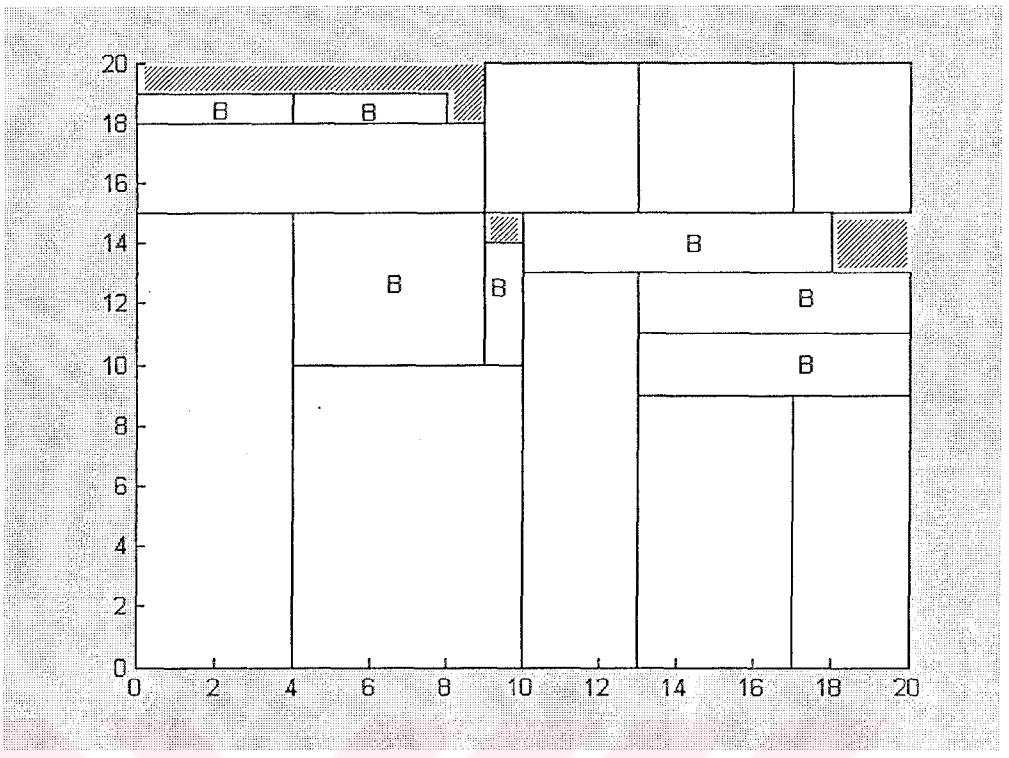
Parça Sayısı : 17
Büyük Parçanın Boyutları : 20x20

Tablo 4.7 Örnek 6.'e ait parçaların boyutları

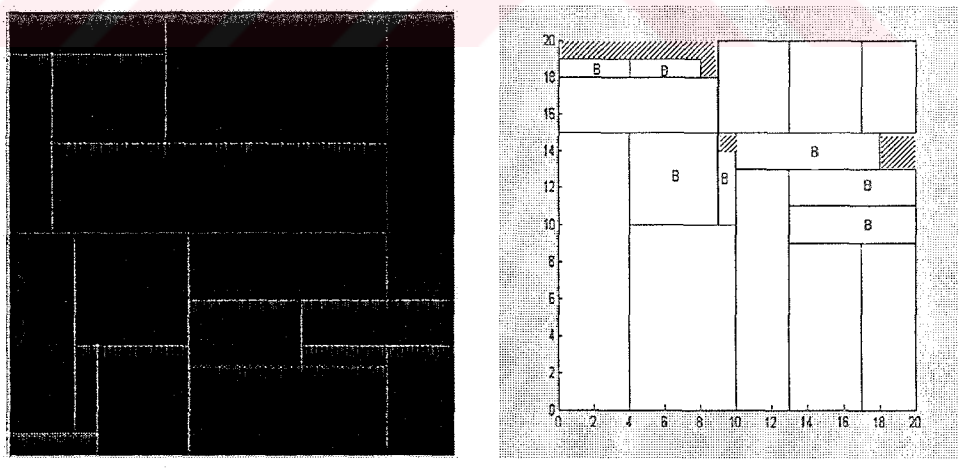
| Parça no | X | Y |
|---|---|---|
| 1 | 4 | 1 |
| 2 | 4 | 5 |
| 3 | 9 | 4 |
| 4 | 3 | 5 |
| 5 | 3 | 9 |
| 6 | 1 | 4 |
| 7 | 5 | 3 |
| 8 | 4 | 1 |
| 9 | 5 | 5 |
| 10 | 7 | 2 |
| 11 | 9 | 3 |
| 12 | 3 | 13 |
| 13 | 2 | 8 |
| 14 | 15 | 4 |
| 15 | 5 | 4 |
| 16 | 10 | 6 |
| 17 | 7 | 2 |



Şekil 4.21 Örnek 6'ya ait mevcut optimum yerleşim



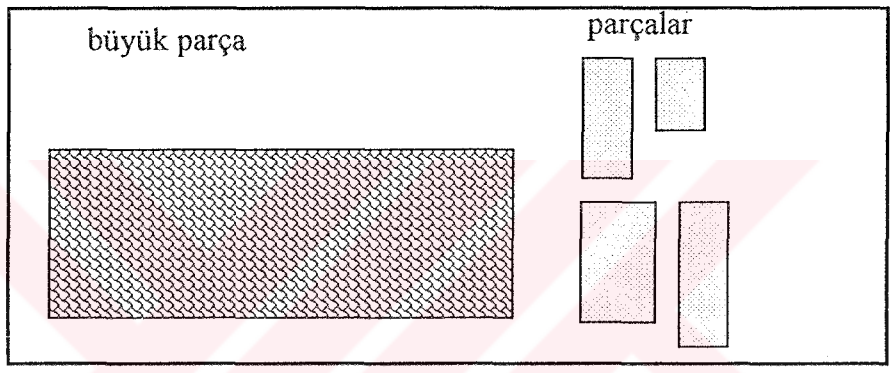

Şekil 4.22 Örnek 6.'nın algoritmaya göre yerleşimi



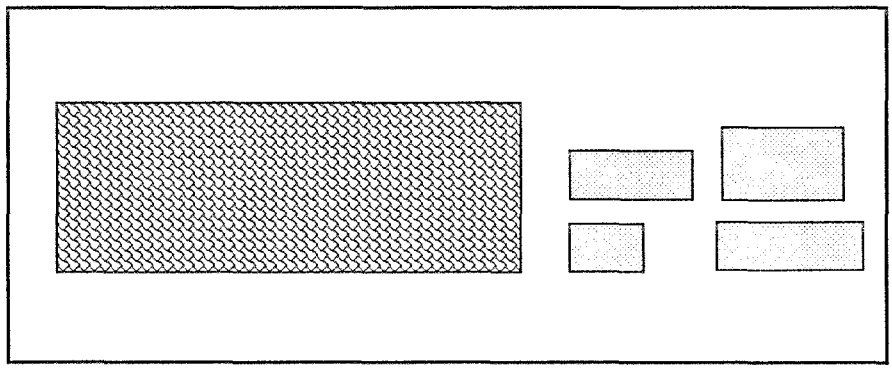
Şekil 4.23 Örnek 6.'nın mevcut optimum yerleşimi ile algoritma çözümünün karşılaştırılması

## 4.4 Algoritmanın Sonuçları

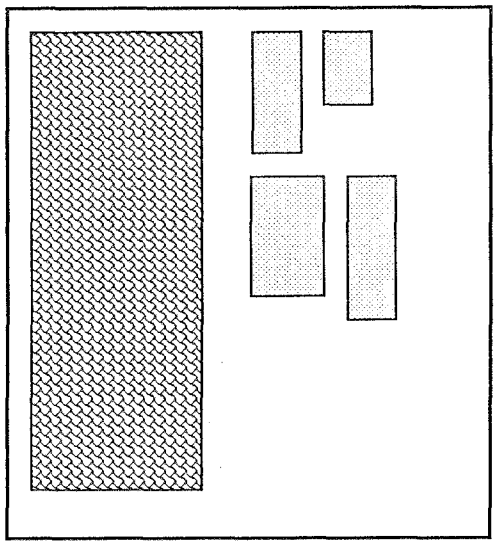
İki boyutlu kesme ve paketleme problemleri için oluşturulan bu algoritma, parçaların her birinin ayrı ayrı döndürülmesi sonucunda oluşan alternatif yerleşimlerden kurtulmak için, başlangıçta, küçük parçaları tek bir grup olarak değerlendirilmektedir. Bu yüzden, ilk olarak çözüme, 4 alternatif ile başlanmaktadır. Küçük parçaların döndürülme şartları diğer satır işlemlerinde değişmeyip, sadece boşluk doldurma işlemlerinde değiştirilmektedir. Bu başlangıç alternatifleri aşağıdaki şekillerde gösterilmiştir.



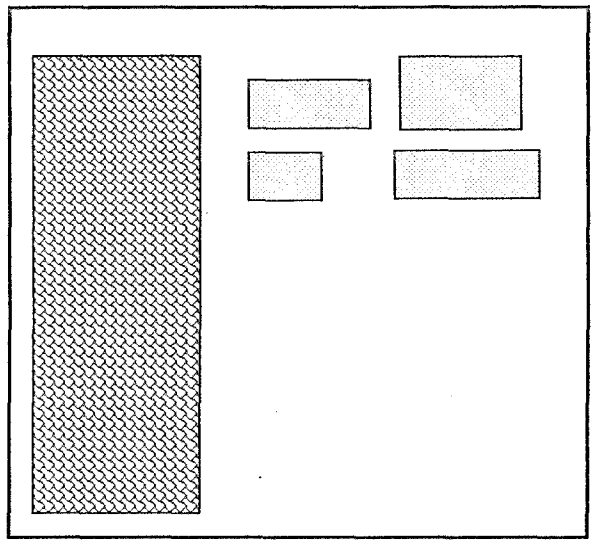

Şekil 4.24 Çözüme başlamada 1. alternatif



Şekil 4.25 Çözüme başlamada 2. alternatif



Şekil 4.26 Çözüme başlamada 3. alternatif



Şekil 4.27 Çözüme başlamada 4. alternatif

Algoritma yerleştirmeye, $X$ eksenine göre başlamaktadır. $X$ eksenine göre, yerleştirilebilecek parça kalmadığında, boşluk doldurma işlemine geçmektedir. Boşluk doldurma işlemi tamamlandıktan sonra ise, aynı kurallarla diğer satır işlemlerine geçilmektedir.

Algoritmanın Aşağı-Sol (BL) algoritmasından farkı, parçaların dizilişlerinin, permütasyon yerine alan büyüklerine göre sıralanması ile elde edilmesidir. Diğer taraftan, satırlara ilk yerleştirilen elemanlar için kural aşağı ve sola mantığı ile aynıdır. Fakat boşluk doldurma işlemi sırasında, bu algoritma farlılık göstermektedir.

Algoritmanın çözüm performansını test etmek için çeşitli örnekler, kullanılmıştır. Kesme ve paketleme problemlerinin en yaygın sınıflandırılmasına göre, örneklerde tek bir büyük parçanın üzerine, farklı veya özdeş boyutta küçük parçaların yerleşimi yapılmıştır.

Örnek 1., 2., ve 3.'ün algoritmaya göre çözümü, aynı zamanda şerit kesme/paketleme problemleri için çözüm oluşturmaktadır. Diğer çözümler çözümün belli bir aşamasından sonra giyotinsiz kesme problemleri çözümleri olarak ortaya çıkmaktadır.

# 5. SONUÇ

Aranan en iyi çözüm, tüm bilim dallarında, optimum adı altında geçmektedir. Bazı problemlerin en basit çözümü, mümkün olan tüm alternatifleri sıralamak ve bunların arasından en iyi olanı seçmektir. Alternatiflerin az olması durumunda bu yöntem kesin sonuca ulaştıran en etkili yöntem olabilir. Fakat alternatifler veya problemdeki değişkenler arttırıldığında, optimal çözümün bulunması çok uzun zaman alacaktır. Karşılaşılan bu zorluğu önleyebilmek için, bu tür problemlerin çözümünde, belirli kurallar geliştiren, sezgisel teknikler kullanılmaktadır. Sezgisel tekniklerin yapısıyla, konulan kısıtlamalar nedeniyle, çözüm alternatiflerinin hepsi denenmeden, optimum sonuç ya da optimuma yakın sonuç bulunmaktadır.

Optimizasyon problemlerinin sınıflandırılması ile başlayan bu çalışma, Yöneylem Araştırmasının en zor problemleri içinde yer alan, kesme ve paketleme problemleri ile devam etmektedir. Kesme ve paketleme problemlerinin yapısı, sınıflandırılması ve çözüm metotları belirtilmiştir. Kesin yöntemlerin uygulamalarında karşılaşılan güçlüklerden dolayı, daha çok sezgisel teknikler üzerinde durulmuştur. Ekonomik değerlerinin fazla olması nedeniyle her endüstri için ayrı çözümler geliştirilmektedir. Çünkü bu tür problemler, endüstri dallarına göre farklı karakteristikler göstermektedirler. Bu nedenle, geliştirilen bir sezgisel teknik her problem türünde etkili olmamaktadır.

Bu çalışmada, iki boyutlu kesme ve paketleme problemleri için yeni bir sezgisel yaklaşım geliştirilmiştir. Bu algoritmada, bir büyük parçanın üzerine, küçük parçaların en iyi yerleşimi amaçlanmaktadır. Algoritmanın temel aşamaları Matlab 6.5 programı ile kodlanmıştır. Optimum çözümü bilinen örnekler ele alınarak, algoritmaya göre tekrar çözülmüştür. Örneklerin sonuçlarından görülebileceği gibi, algoritma her zaman optimum sonucu vermemektir. Çünkü parça dizilişlerinin gerçekte bir formülü yoktur. Çözüm alternatiflerini daraltan başka bir durum ise parça birleştirerek, kare parçalar elde edilmesidir. Algoritmanın bilgisayara aktarılması ve parça birleştirme ile yeni çözümler oluşturulması gelecek çalışmalar arasında yer almaktadır.

Şirketlerin en büyük rekabet unsurlarından biri hızdır. Bu tür problemlerle karşı karşıya kalan şirketler, problem çözümünde, çalışan elemanların tecrübesine bağlı olarak manuel çözümler ile bazı tekniklerden yararlanarak oluşturulan bu çözümler arasında tercih yapmalıdırlar.

## KAYNAKLAR


[1] Hopper, E. and Turton, B. C. H., "A Review of the Application of Meta-Heuristic Algorithms to 2D Strip Packing Problems", *Artificial Intelligence Review,*(2001), **16**, 257-300.

[2] Özgüven, C., Doğrusal Programlama ve Uzantıları, Detay Yayıncılık, Ankara, (2003).

[3] Moldibi, Ş., "Optimizasyonlar", http://www.istanbul.edu.tr/Bolumler/enformatik/seminer/Optimizasyonlar.pps, erişim tarihi:04.07.2004.

[4] Chinneck, J. H., Practical Optimization: A Gentle Introduction, System and Computer Engineering, Carleton University, Ottowa, Canada, (2000).

[5] Gill, P. E., Murray W., Wright, M.H., Practical Optimization, Academic Pres, Scotland, (1989).

[6] Mathematics and Computer Science Division, http://www-fp.mcs.anl.gov/otc/Guide/OptWeb/index.html erişim tarihi:13.05.2004

[7 ] Öztürk, A., Yöneylem Araştırması, 6. Baskı, İstanbul, (1997).

[8 ] Kocaeli Üniversitesi Endüstri Mühendisliği Kulübü, http://www.kouemk.com/linkler/optimizasyon.asp erişim tarihi:25.07.2004.

[9 ] Reeves, C., Wiley, J., Modern Heuristic Techniques for Combinatorial Problems, John Wiley & Sons, Inc. New York, NY, USA, (1993).

[10] Dengiz, B., Sezgisel Optimizasyon Ders Notları, Gazi Üniversitesi Endüstri Mühendisliği Bölümü, http://www.mmf.gazi.edu.tr/~berna/turkce/courses/enm543.html, erişim tarihi:30.10.2002.

[11] Türkbey, O., Karmaşıklık Analizine Bakış, Gazi Üniversitesi Endüstri Mühendisliği, http://muendiz.freehomepage.com/arastirmakonusu.html, erişim tarihi: 19.07.2004

[12] Bali, Ö., Gencer, C., "K.K. Lojistik Komutanlığı 3., 4. ve 5. Kademe Depolarında Küme Örtüleme Yaklaşımı ile Bir İyileştirme Çalışması", *Savunma Bilimleri Dergisi*, (2002), 1-17.

[13] Gökmen, Y., "Tugaylar İle Depo Ve Fabrika Komutanlıkları Arasındaki Malzeme Nakliyatının Optimize Edilmesi", *Kara Harp Okulu Bilim Dergisi*, (2003), **2.**

[14] www.biltek.tubitak.gov.tr/dergi/00/haziran/sondenklem.pdf, erişim tarihi: 30.07.2004

[15] Hirayama, K., Studies on Solving Methods of Some Combinatorial Problems by GA, Course in Engineering of Social Development at Tottori University, Tottori, Japan, (1997).

[16] National Institute of Standards and Technology, awww.nist.gov/dads/HTML/determinalgo.html erişim tarihi: 16.05.2004

[17] Güner, E., Altıparmak, F., "İki Ölçütlü Tek Makinalı Çizelgeleme Problemi için Sezgisel Bir Yaklaşım", *Gazi Üniv. Müh. Mim, Fak. Dergisi*, (2003) cilt 18, **3**, 27-42

[18] Cedimoğlu, İ. H., Geyik, F., "Tabu Arama Tekniği ile Klasik İş-Atölyesi Çizelgeleme", YAEM'99-Yöneylem Araştırması ve Endüstri Mühendisliği XX. Ulusal Kongresi.

[19] Emel, G. G., Taşkın, Ç., "Genetik Algoritmalar ve Uygulama Alanları", *Uludağ Üniversitesi İ.İ.B.F. Dergisi*, (2002), cilt 21, **1**, 129-152

[20] Sosyal, O., Çok Değişkenli Fonksiyonların En İyilemelerinde Genetik Algoritmaların Davranışları, İzmir Fen Lisesi, (1998).

[21] Yiğit, V., Türkbey, O., "Tesis Yerleşim Problemlerine Sezgisel Metotlarla Yaklaşım", *Gazi Üniv. Müh. Mim. Fak. Dergisi,*(2003), cilt 18, **4**, 45-56.

[22] Hopper, E. and Turton B. C. H., "An Empirical Investigation of Meta-heuristic and Heuristic Algorithms for a 2D Packing Problem", *European Journal of Operational Research*, (2000), 128/**1**, 34-57.

[23] Hopper E., Turton, B., "A Genetic Algortihm for a 2D Industrial Packing Problem", *Computers and Industrial Engineering*, (1999), **37**, 375-378.

[24] Karelahti, J., Solving the cutting stock problem in the steel industry, Helsinki University of Tecnology, Deparment of Engineering Physics and Mathematics, M.Sc. Thesis, (2002), Otaniemi, Finland.

[25] Hopper E., Cutting and Packing Problems, http://circuits.cf.ac.uk/hopper, erişim tarihi: 26.05.2003

[26] Yaman, R., Ergün, K., Cutting and Packing Problems and A Heuristic Algorithm for Rectangular Element Placement, (Çalışılan makale).

[27] Elmaghraby A. S., Abdelhafiz E.,. Hassan M F., (2004), An Intelligent Approach to Stock Cutting Optimization, http://www.louisville.edu/speed/emacs/mrl/publications/pdffolder/caine00_ehab.pdf

[28] Yargıç, B., Kesme ve Paketleme İşlemleri için Yüzey Tanıma Temelli Yazılım Geliştirme, Balıkesir Üniversitesi Endüstri Mühendisliği Bölümü Bitirme Projesi, (2001).

[29] Söke, A., Bingül, Z., "İki Boyutlu Kesme Problemlerinin Genetik Algoritma Yardımıyla Çözümünün İncelenmesi", *International XII. Turkish Symposium on Artificial Intelligence and Neural Network-TAINN*, (2003).

[30] Euro Special Interest Group on Cutting and Packing, SICUP, (2004 ) http://www.apdio.pt/sicup