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

İKİ BOYUTLU DİKDÖRTGEN ŞEKİLLİ STOK KESME PROBLEMLERİ İÇİN SEZGİSEL-METASEZGİSEL ALGORİTMA VE YAZILIM GELİŞTİRME

YÜKSEK LİSANS TEZİ

EMRAH ALBAYRAK

BALIKESİR, OCAK - 2013

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

İKİ BOYUTLU DİKDÖRTGEN ŞEKİLLİ STOK KESME PROBLEMLERİ İÇİN SEZGİSEL-METASEZGİSEL ALGORİTMA VE YAZILIM GELİŞTİRME

YÜKSEK LİSANS TEZİ

EMRAH ALBAYRAK

BALIKESİR, OCAK - 2013

KABUL VE ONAY SAYFASI

Emrah ALBAYRAK tarafından hazırlanan **"İKİ BOYUTLU DİKDÖRTGEN ŞEKİLLİ STOK KESME PROBLEMLERİ İÇİN SEZGİSEL-METASEZGİSEL ALGORİTMA VE YAZILIM GELİŞTİRME"** adlı tez çalışmasının savunma sınavı 17.01.2013 tarihinde yapılmış olup aşağıda verilen jüri tarafından oy birliği / oy çokluğu ile Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalı Yüksek Lisans Tezi olarak kabul edilmiştir.

Jüri Üyeleri	İmza
Danışman Prof.Dr. Ramazan YAMAN	.....
Üye Yrd.Doç.Dr. Demet GÖNEN	.....
Üye Yrd.Doç.Dr. Gültekin KUVAT	.....

Jüri üyeleri tarafından kabul edilmiş olan bu tez BAÜ Fen Bilimleri Enstitüsü Yönetim Kurulunca onanmıştır.

Fen Bilimleri Enstitüsü Müdürü

Prof. Dr. Hilmi NAMLI .....

ÖZET

İKİ BOYUTLU DİKDÖRTGEN ŞEKİLLİ STOK KESME PROBLEMLERİ İÇİN SEZGİSEL-METASEZGİSEL ALGORİTMA VE YAZILIM GELİŞTİRME

YÜKSEK LİSANS TEZİ

EMRAH ALBAYRAK

BALIKESİR ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

(TEZ DANIŞMANI:PROF.DR. RAMAZAN YAMAN)

BALIKESİR, OCAK - 2013

Günümüzde küresel rekabet her geçen gün artmaktadır. Rakiplerine göre fiyat, kalite ve zaman noktasında üstünlük sağlayan firmalar başarı elde etmektedir. Bu noktada stok kesme problemleri doğru çözüldüklerinde tasarruf açısından büyük önem kazanmaktadır.

Stok kesme problemi; büyük boyutlardaki stok malzemesinden küçük ebatlardaki sipariş malzemelerinin en az fire ile elde edilmesidir. Stok kesme problemleri mobilya, kağıt, demir, çelik, plastik, cam gibi birçok sektörde görülmektedir.

Bu çalışmada iki boyutlu dikdörtgen elemanlı stok kesme problemleri hakkında bilgi verilmiş, çözüm yaklaşımları anlatılmıştır. Çözüm yaklaşımları ışığında sezgisel ve metasezgisel yazılım geliştirilmiştir. Bu yazılımın geliştirilmesinde Visual Basic 6.0 kullanılmıştır. Geliştirilen bu yazılımlar ile kare ve dikdörtgen parçalardan oluşan, parça sayıları farklı örnekler çözülmüştür. Aynı örnekler ticari bir program olan Diamino programı ile çözülerek geliştirilen sezgisel ve metasezgisel yazılımın performansı değerlendirilmiştir. Elde edilen sonuçlar geliştirlen metasezgisel yazılımın kabul edilebilir sürede iyi çözümler ürettiğini göstermiştir.

ANAHTAR KELİMELER: iki boyutlu stok kesme problemi, genetik algoritma, sezgisel ve metasezgisel yöntem, aşağı sol algoritma

ABSTRACT

RECTANGULAR SHAPED TWO-DIMENSIONAL CUTTING STOCK PROBLEM FOR HEURISTIC -META HEURISTIC ALGORITHM AND SOFTWARE DEVELOPMENT

MSC THESIS

EMRAH ALBAYRAK

BALIKESIR UNIVERSITY INSTITUTE OF SCIENCE

INDUSTRIAL ENGINEERING

(SUPERVISOR:PROF.DR. RAMAZAN YAMAN)

BALIKESİR, JANUARY 2013

At the present, global competition is increasing every day. Compared to its competitors on price, quality and time point is obtained which is superior to their success. Cutting stock problems at this point are becoming of great importance in terms of saving when solved correctly.

Cutting stock problem; a large-dimensioned(sized) stock material (a large stock material sizes) with a minimum of waste materials in order to achieve small-dimensioned order materials (small sizes).

Cutting stock problems; furniture, paper, iron, steel, plastic, glass such as seen in many sectors.

In this study, two-dimensional rectengular element provides information on cutting stock problems and solution approaches are described. According to solution approaches, heuristic and metaheuristic algorithms are developed. Visual Basic 6.0 is used for the development of this software. With these softwares, different part number of examples consisting square and rectangular pieces, are solved. The same examples are solved by acommercial program of Diamino Program. According to this solution, heuristic and metaheuristic software performance is evaluated. These results show that developed metaheuristic software produces better solutions in an acceptable time.

KEYWORDS: two dimensional cutting stock problem, genetic algorithms, heuristic and metaheuristic methods, bottom left algorithm

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

Sayfa

TABLO LİSTESİ

Sayfa

KISALTMALAR LİSTESİ

NP : Belirleyici Olmayan Polinom
SKP : Stok Kesme Problemi
BL : Aşağı Sol Algoritma
DP : Dinamik Programlama
GA : Genetik Algoritma
TB : Tabu Algoritması
BLF : Aşağı Sol Dolgu Algoritması
GRASP : Açgözlü Rassallaştırılmış Uyarlamalı Arama

ÖNSÖZ

Stok kesme problemleri günümüzde çeşitli sektörlerde karşımıza çıkmaktadır. Stok kesme problemlerini henüz en iyiyi yeterince kısa zamanda çözen bir algoritmanın olmaması bu konuya olan ilgi ve araştırmaları arttırmaktadır. Bu çalışmada bir kısım stok kesme problemlerine çözüm aranmaktadır.

Öncelikle bu çalışmayı hazırlarken benden desteğini esirgemeyen, yol gösteren değerli hocam Prof.Dr. Ramazan YAMAN'a sonsuz teşekkür ederim.

Bu çalışmada yardımlarını esirgemeyen ve projenin gelişmesinde katkıları olan değerli hocam Arş.Gör.Dr. Kadriye ERGÜN' e sonsuz teşşekkür ederim.

Benden maddi ve manevi desteğini esirgemeyen annem, babam ve kardeşlerime sonsuz teşekkür ederim.

Bu çalışmanın geliştirilmesinde emeği olan Ahmet GÜNER ve Tuba Evrim AYDIN' a sonsuz teşekkür ederim.

1. GİRİŞ

Günümüzde ekonomik olarak ayakta kalabilen kuruluşlar rakiplerine karşı avantaj elde edebilen kuruluşlardır. Bu avantajlardan en önemlisi malzeme kullanımındaki verimliliktir. Malzeme kullanımındaki verimliliği arttırmak için fireleri en aza indirmek gerekmektedir. Bu da rekabetin geliştiği bir çok ortamda kesme paketleme problemleri ile ilgili araştırmaları arttırmaktadır.

Kesme paketleme problemi "Belirli ölçülerde tedarik edilebilen parçalardan daha küçük boyutlarda parçalar elde etmek" olarak tanımlanabilir.

Bu çalışmaya konu olarak gösterilen stok kesme problemlerinin en basit hali olan tek boyutlu stok kesme problemi bile NP-hard grubundadır. Değişik türdeki iki boyutlu kesme problemleri kaynaklarda geniş bir biçimde yer almaktadır. Bu tür problemler; büyük bir dikdörtgen ana malzemeden küçük parçaların kesilmesi problemi olarak düşünülebildiği gibi; küçük parçaların büyük bir dikdörtgen olan ana malzemeye yerleştirilmesi problemi olarak adlandırılmaktadır [1].

Kesme paketleme problemlerinin zorluğu, büyük parça üzerine yerleştirilecek olan parçaların geometrisi ve problemin yapısından kaynaklanan kısıtlamalara bağlıdır. Problemin çözümü için bütün olasılıkların denendiği kombinasyonel en iyileme algoritmaları çok geniş bellek alanına ve yüksek hesaplama hızlarına ihtiyaç duymasının yanında çok uzun sürelere ihtiyaç duymaktadır. Bu da bizi problemin çözümünde sezgisel ve metasezgisel yöntemlerin kullanılmasına yönlendirecektir. Fakat optimal sonucu polinomial zamanda veren sezgisel ve metasezgisel algoritma henüz geliştirilememiştir. Bu nedenle bu konuyla ilgili araştırmalar hergeçen gün artmaktadır. Paketleme veya kesme problemleri üzerindeki araştırmaların büyük bir kısmını kesin çözüm üretmek için kullanılan doğrusal programlama teknikleri oluşturmaktadır. Kesin çözümler için geliştirilen algoritmalar sadece ufak kümeler için etkili çalışabilmektedir. Bu sebeple daha fazla elemanı olan kümeler için başka teknikler uygulanmaktadır. Kesme paketleme problemleri ile ilgili yapılan bazı araştırmaları kronolojik olarak sıralayacak olursak;

SKP ilk olarak 1939' da Rus matematikçi Kantorovich tarafından ele alınmıştır. 1960' da yayınlanan bu çalışmada, tek boyutlu stok kesme problemi, her biri ***n*** adet makinada işlenebilecek ***m*** adet işin çizelgelenmesi probleminin bir uzantısı olarak tanımlanmış ve bu doğrultuda fire minimizasyonuna yönelik tamsayılı doğrusal programlama modeli geliştirilmiştir [2].

Literatürde SKP'nin temeli, Gilmore ve Gomory' nin (1961,1963,1964,1966) çalışmalarına dayandırılmaktadır. Gilmore ve Gomory (1964), 1961 ve 1963' te yayınlanan çalışmalarında tek boyutlu SKP için geliştirdikleri sütun oluşturma tekniğini iki ve üç boyutlu problemler için de kullanmışlardır. Bu yıllardan itibaren SKP ile ilgili çok fazla sayıda çalışma yapılmıştır [3].

Bischoff ve Dowsland üretim dizaynı ve dağıtımı ile ilgili problemlerin mikro bilgisayar ortamındaki uygulamaları üzerinde çalışmışlardır. İki boyutlu kesme problemlerinde sezgisel çözümlerin yardımı ile büyük ilerlemeler kaydedilmiştir. Bu çalışmaların yanı sıra, analitik çözümler yardımı ile yapılan çalışmalar bulunmaktadır [4].

Dowsland tarafından geliştirilen analitik teknik, kargo yükleme ile ilgili her türlü probleme uygulanabilmekte ve sonuçları bilgisayardan birkaç dakikada alınabilmektedir ancak optimal sonucu garanti etmemektedir [4].

Analitik çözümlerle ilgili önemli bir çalışma Beasley tarafından yapılmıştır. Bu çalışmada kesme kaybı problemlerini tamsayılı programlama yapısıyla çözüm aranmıştır. Beasley, büyük bir dikdörtgen ana parçadan küçük parçaların kesildigi durumda kesilen küçük parçaların toplam değerini maksimum yapan bir algoritma geliştirmistir. Beasley iki boyutlu kesme problem için dal ve sınır algoritması ve Lagrange gevşetmesi yöntemlerini kullanarak yeni bir algoritma geliştirmiştir [1].

Chen, vd. (1991) iki boyutlu kesme problemlerinde optimal sonuç veren doğrusal programlama modeli kurmuşlardır. Chen, Sarin ve Ram ise birden fazla sayıdaki yerleştirelecek alanlara kısıtlar ekleyen bir model geliştirmişlerdir [5].

Lodi, vd. (2003) matematiksel modeller, sınır algoritmaları, yaklaşım algoritmaları, sezgisel ve sezgi üstü (meta-heuristic) yöntemlerin iki boyutlu kesme problemlerindeki uygulamalarının incelendiği kapsamlı bir araştırma yapmıştır [6].

Genetik algoritma tekniği, Michigan Üniversitesi' nde yer alan John Holland tarafından 1970' li yıllarda ortaya çıkmıştır. Smith (1985), ilk kez paketleme problemlerine çözüm bulmak için genetik algoritmayı kullanmıştır. Jain ve Gea (1998) yaptığı çalışmada genetik algoritma kullanarak çokgen parçaların yerleşimi ile ilgili bir çalışma hazırlamışlardır. Genetik algoritma işlemcileri geliştirilerek hybrid bir yaklaşımda bulunulmuştur [7].

Ayrıca Hopper ve Turton (2001) tarafından sezgisel bir yerleşim yöntem geliştirerek ve genetik algoritma kullanarak önemli bir çalışma yapmışlardır [8].

Bir ve iki boyutlu SKP ile ilgili çalışmalara paralel olarak literatürde yer alan uygulamaya yönelik çalışmaların da belirli endüstrilerde yoğunlaştığı görülmektedir. Özellikle kağıt (Pierce 1964, Johnson; Rennick; Zak 1997, Menon; Schrage 2002; Correia; Oliveira; Ferreira 2004), metal (Tokuyama; Ueno 1985, Thorsen; Vidal 1991, Chu, Antonio 1999, Hung; Sumichrast, Brown 2003), ağaç ve mobilya (Carnieri; Mendoza; Gavinho 1994, Rönnqvist 1995, Wagner 1999), cam (Chambers; Dyson 1976, Farley 1983) ve tekstil (Farley 1988, Martens 2004) sektörlerine yönelik çalışmalar yapılmıştır [9].

Binkley ve Hagiwara (2007), parçaların döndürülmesine izin verildiği ve izin verilmediği durumları göz önüne alarak aynı amaç üzerine çalışmışlardır. Bu amaçla geliştirilen bir dört köşe sezgiseli ayrı ayrı GA ve Paralel Birleşimli TB(Tavlama benzetimi) algoritması ile melezlenmiştir. 21 tanesi Hopper ve Turton (2001), 10 tanesi Leung ve ark. (2003) makalelerinden alınmış olan 10-197 dikdörtgen parça arasında değişen 31 test problemi üzerinde algoritmalar denenmiş ve Leung ve ark. (2003) makalesinde geliştirilen GA ve TB melez algoritması sonuçları ile karşılaştırılmıştır [10].

Bu çalışmada birinci ve ikinci bölümde kesme ve paketleme problemlerinin tanımları yapılmış, kısıtları, amaçları, sınıflandırılması ve matematik yapıları hakkında bilgi verilmiştir.

Üçüncü bölümde kesme ve paketleme problemlerinin önemli bir sınıfı olan stok kesme problerinden bahsedilmiştir. Stok kesme problemlerine çözüm yaklaşımı

olarak yerleştirme algoritmaları, analitik, sezgisel ve bazı metasezgisel metodlardan bahsedilmiştir.

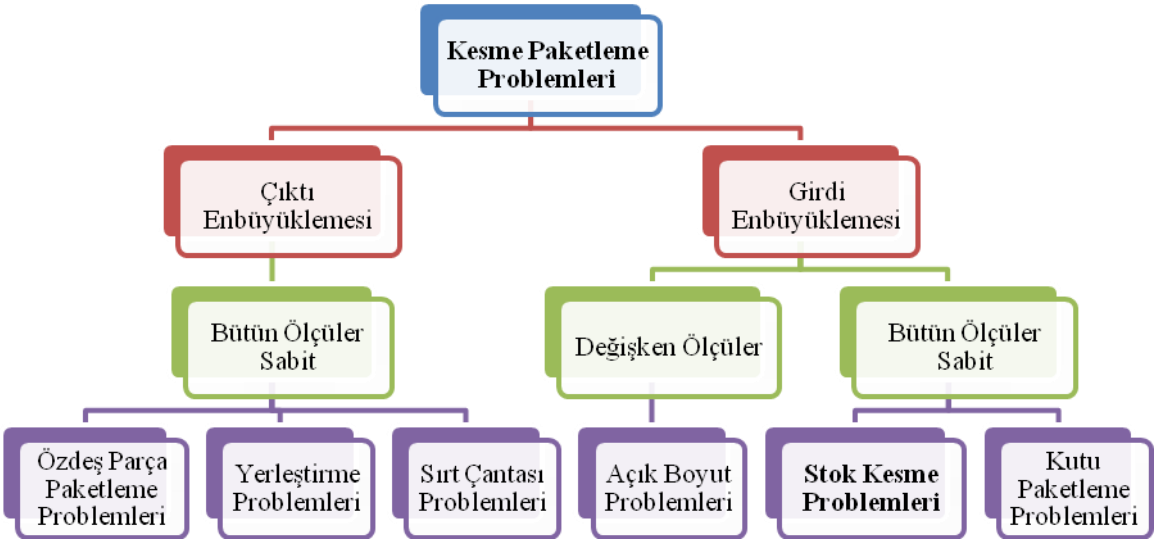
Dördüncü bölümde aşağı sol dolgu algoritması kullanılarak sezgisel ve metasezgisel yazılım geliştirilmiştir. Bu yazılım Visual Basic 6.0 kullanılarak geliştirilmiştir. Geliştirilen yazılımın performansı test örnekleri ile incelenmiştir. Elde edilen sonuçlar, aynı test örneklerinin Diamino program ile çözülmesiyle oluşan sonuçlarla karşılaştırılmış ve geliştirilen algoritma ve yazılımın performansı değerlendirilmiştir.

Beşinci bölümde ise uygun sürede daha iyi sonuçlar elde edilmesi için yapılması gerekenlerden ve bir işletmenin tasarım sürecinde nelere dikkat etmesi gerektiğinden bahsedilmiştir.

2. KESME PAKETLEME PROBLEMLERİ

2.1 Kesme Paketleme Problemlerine Genel Bakış

Kesme ve paketleme problemleri, bir parça üzerinden birden çok sayıda küçük parçaların ayrılmasının ya da yerleştirmenin bulunması ile ilgilenen eniyileme problemleridir. Yani belirli boyuttaki malzemeden daha küçük boyutlarda parçalar elde edilmesidir. Kesme paketleme problemlerinin gruplandırılması Şekil 2.1'de gösterilmiştir.



Şekil 2.1: Kesme paketleme problemlerinin gruplandırılması [11]

Kesme paketleme problemleri, küçük bir çalışma uzayı içinde olması durumu hariç, bu problemlere en iyi çözümün üretilmesinin imkansız olduğu için NP-complete (non-deterministic polynomial-complete) problemler olarak bilinir. NP (non-deterministic polinomial) problemler, belirsiz Turing Makinesi ile çok terimli zamanda çözülebilen karar problemlerini içeren karmaşıklık sınıfıdır. Bu sınıftaki problemler belirli Turing Makinesi ile çok terimli zamanda doğrulanabilirler ve bu

şekilde doğrulanabilen her problem NP (non-deterministic polinomial) sınıfındadır. Bu nedenle NP, (belirli Turing Makinesi ile) çok terimli zamanda doğrulanabilen problemlerin sınıfı olarak da tanımlanabilir. Belirli Turing makinesi aynı zamanda belirsiz Turing Makinesi olduğundan, P sınıfındaki bütün problemler aynı zamanda NP' dedir. Eğer NP sınıfındaki her problem polinomial olarak bir P problemine dönüştürülebiliyorsa bu durum P problemi NP-hard olarak adlandırılır. Ayrıca eğer P probleminin kendisi de NP sınıfına ait ise, P problemi NP-complete olarak bilinmektedir. NP-complete problemler NP sınıfındaki problemlerin en zorudur [9].

$$\boldsymbol{NP-Hard}: \quad [12] \qquad \textbf{(1)}$$

(1) nolu denklemde, $L \leq_p H$ L probleminin, H problemine çok terimli zamanda indirgenebildiği anlamına gelir. Bir başka deyişle, NP-hard sınıfındaki her hangi bir problem çok terimli zamanda çözülebilirse, NP sınıfındaki bütün problemler çok terimli zamanda çözülebilir. NP-complete , hem NP olup hem NP-hard olan problemlerin sınıfıdır [12].

Arama uzayının büyüklüğü nedeniyle kesme paketleme problemlerinin çözümü için yönlendirilmemiş, bir arama yapmak oldukça verimsiz olduğundan, probleme ait en iyi çözümün bulunabilmesi için büyük arama uzayı içinde düzenli bir arama yapılması gerekir. Bu sebeple araştırmalar, en iyi çözüme yakın iyi çözümleri verimli bir şekilde bulan olasılıksal yaklaşım teknikleri üzerinde yoğunlaşmaktadır [13].

2.2 Kesme Paketleme Problemlerinin Sınıflandırılması

Kesme ve paketleme problemleri, Dyckhoff tarafından geliştirilen sınıflandırma şeması ile gruplandırılabilir. Problemlerin sınıflandırması dört temel özellik altında yapılabilir [14].

- **Boyutluluk**
  - Boyut sayısı (N)

- **Stok malzemelerinin seçimi**
  - Bir adet stok malzemesi (O)

- Birden fazla aynı ebatta stok malzemesi (I)
- Farklı stok malzemesi (D)

- **Atama biçimi**
  - Tüm stok malzemeler ve küçük parçaların seçimi (B)
  - Stok malzemelerinin seçimi ve tüm küçük parçalar (V)

- **Küçük parçaların ayrımı**
  - Farklı boyutlardaki az sayıda küçük parçalar (F)
  - Farklı boyutlardaki çok sayıda küçük parçalar (M)
  - Birbirine benzer boyutlardaki çok sayıdaki küçük parçalar (R)
  - Benzer çok sayıdaki küçük parçalar (C)

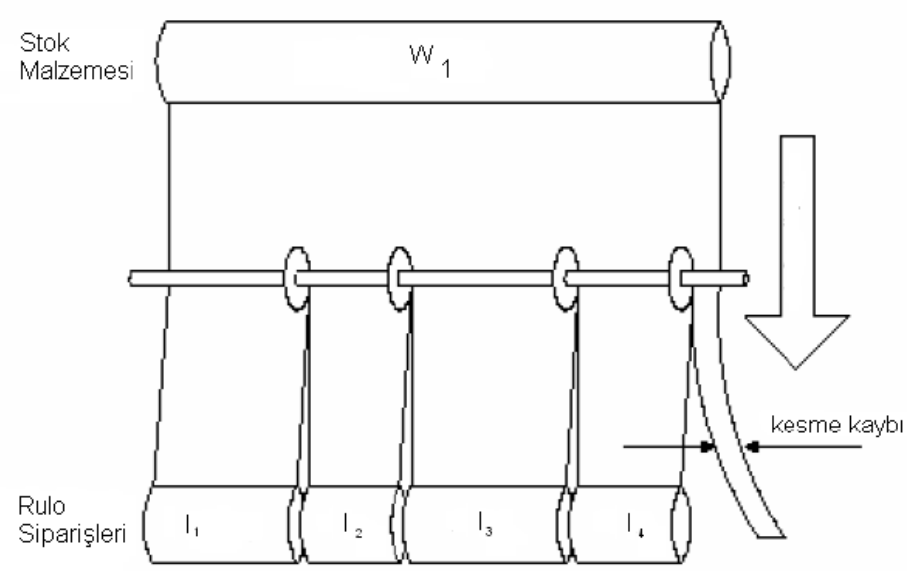
Tablo 2.1: Bazı kesme paketleme problemlerinin gruplandırılması [14]

Problem	**Simgesel Gösterim**
Klasik Sırt Çantası	1 / B / O /
Palet yükleme	2 / B / O / C
Konteyner yükleme	3 / V / I
Montaj hattı dengeleme	1 / V / I / M
Bellek tahsisi	1 / V / I / M
Çok dönemli sermaye bütçeleme	n / B / O /
Genel malzeme kesme veya kesme kaybı	n / / /

2.2.1 Kesme Paketleme Problemlerinde Boyut Durumu

Tek boyutlu kesme: Demir çubuklar ve kağıt rulolara uygulanan bu çeşit kesimde, ikinci ve üçüncü boyutların anlamı kalmamaktadır. Buna örnek Şekil 2.2'de görülmektedir. Gilmore ve Gomory (1964) kağıt rulolarda bir boyutlu stok kesme problemini ele almış ve bir stok kesme problemine en iyi sonucu veren ilk yöntemi geliştirmiştir [15]. Kullanıldığı bazı alanlar;

- Klasik sırt çantası,
- Demir çubukların kesilmesi,
- Montaj hattı dengeleme,
- Bellek tahsisidir.

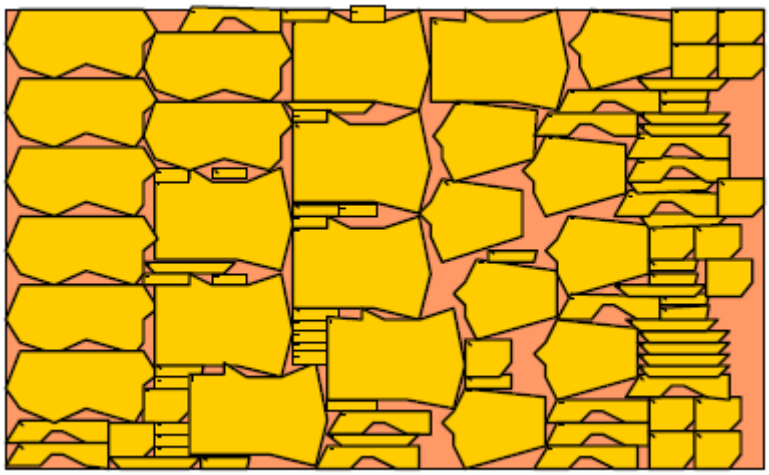


Şekil 2.2: Bir boyutlu stok kesme problemi

İki boyutlu kesme: İki boyutlu stok kesme problemi (Şekil 2.3), sac, cam, kontrplak, sunta gibi levhalardan kesilecek parçaların bu levhalar üzerine, kullanılan levha sayısını veya toplam fireyi minimize edecek şekilde dizilmesini ele alır [9]. Kullanıldığı bazı alanlar;

- Gemi yapımında çelik levhaların yerleştirilmesi

- Mobilya yapımında kerestelerin kesilmesi
- Makale, haber ve reklamların gazete sayfalarına yerleştirilmesi
- Kutu yapımında mukavva ve kartonların kesilmesidir.



Şekil 2.3: İki boyutlu stok kesme problemi [16]

Üç boyutlu Problemler: Üç-boyutlu stok yerleştirme problemlerinde (Şekil 2.4) aynı veya farklı biçim ve ölçülerdeki ***n*** adet kutunun sabit bir biçim ve hacime sahip olan bir konteynere toplam hacimi en küçükleyecek şekilde yerleştirilmesi amaçlanmaktadır .

Gerçek hayat uygulamalarında, üç-boyutlu istif problemlerinin birçoğu iki boyutluya indirgenebilir. Örneğin, süt kapları (şişeleri) sadece şişe dipleri aşağıda kalacak şekilde yerleştirilebilir [9]. Kullanıldığı bazı alanlar;

- Konteyner yükleme,
- Ambalaj kutularına yerleştirme yapılması

olarak verilebilir.

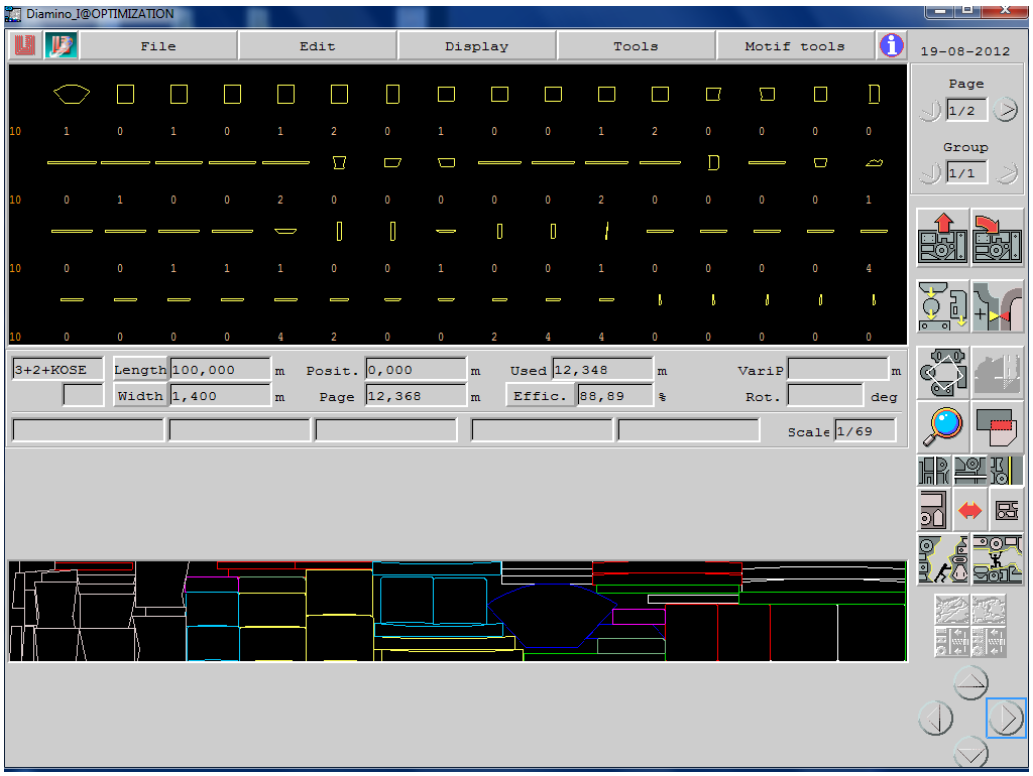
Şekil 2.4: Üç boyutlu stok yerleştirme problemi [17]

2.3 Stok Kesme Problemleri

Kesme ve paketleme problemleri arasında Stok Kesme Problemleri (SKP) en çok karşımıza çıkan kesme paketleme problem çeşididir. Tekstil, deri, kağıt, cam, kereste ve metal gibi birçok endüstri dalında sık bir biçimde SKP ile karşılaşılmaktadır. Stok kesme problemi ile ilgili ilk formülasyonu 1939 yılında Kantorovich tarafından geliştirilmesine rağmen ancak 1960 yılında yayınlanabilmiştir [2].

Kesilecek malzemeye, ***ana malzeme***, ana malzemeden kesilen küçük parçalara ise ***sipariş parçası*** denilmektedir. ***Kesme planları*** (Pattern), kesme işleminin nasıl yapılacağını, bir başka deyişle parçaların ana malzemeye ne şekilde yerleştirileceğini gösteren geometrik modellerdir. Sipariş listesindeki parçaların tümünün üretilmesini sağlayacak şekilde birden fazla kesme planı hazırlanabilir ve bu planlar kendilerini defalarca tekrarlayabilir [18].

Ana malzemeden kesilen sipariş parçaları sonrasında kalan parçalar fire olarak tanımlanmaktadır. Çoğu SKP' de amaç fonksiyonu fireyi minimize etmektir. Şekil 2.5' de Diamino ile yapılan bir kesim planı gösterilmiştir.



Şekil 2.5: Diamino ile yapılan bir kesim planı

2.3.1 Stok Kesme Problemlerinin Kısıtları

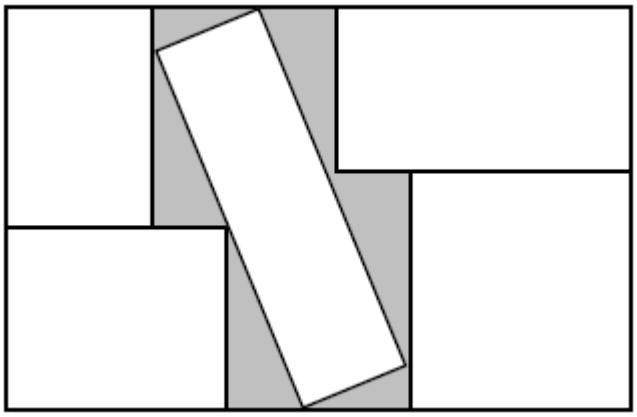
Stok kesme problemlerindeki önemli bazı kısıtlar;

- Kesilecek olan stok malzemelerinin ana stok malzemesinin sınırları içerisinde olması,
- Kesilecek parçaların koordinat eksenlerinin kesişmemesi yani kesilecek olan malzemelerin üst üste gelmemeleri,
- Kesilecek olan malzemenin homojen olup olmamasına göre, parçaların kesilirken döndürülmesine izin verilip verilmemesi,
- Kesilecek olan stok malzemesinin talep miktarı ile sınırlandırılıp sınırlandırılmayacağı,
- Kullanılacak teknolojiye ait kısıtlar (Giyotinli kesme vb.)
- Kesilecek olan stok malzemelerinin kesilme önceliğinin verilmesi,
- Teknolojik kısıt ve zaman kısıtı bazı önemli kısıtlardır[17].

2.3.1.1 Dikdörtgen Parçalara Özgü Durumlar

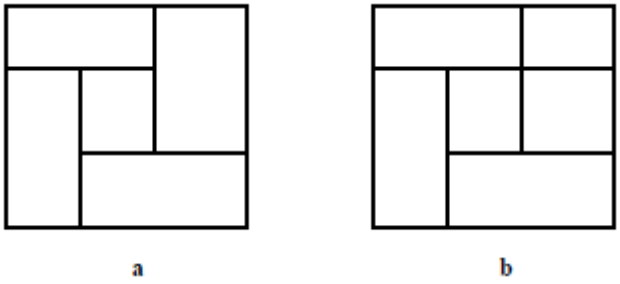
Dikdörtgen parçaların stok malzemesinden kesim işlemi donanım gerektirmeden basit ve ucuz tezgahlarla yapılabilmektedir. Dikdörtgen parçaların kesiminde SKP' nin türüne göre karşılaşılan bazı kısıtlar vardır. Bunlar; dik açılılık (orthogonality), giyotinle kesim, kademeli kesme ve döndürmedir.

Dik açılılık (orthogonality): Dik açılı (orthogonal) malzeme kesme-istifleme problemlerinde dikdörtgen şeklindeki parçalar kümesi, dikdörtgen ana malzeme üzerine kenarları sırasıyla x ve y eksenlerine paralel olacak şekilde yerleştirilmektedir. Dik açılı olmayan kesme (Şekil 2.6), bazı özel durumlarda kayıpları azaltabilse bile kesme işleminde kullanılan bazı tezgahlar genelde dik açılı olmayan (non-orthogonal) kesime izin vermemektedir [13].



Şekil 2.6: Dik açılı olmayan (non-orthogonal) kesme [13]

Giyotinle Kesim: Gerçek veya kavramsal bir giyotin tezgahında her ardışık kesim (birbirini izleyen her kesim) levhanın bir ucundan diğerine uzayacak şekilde yapılmalı, dikdörtgen parçalar bu şekilde kesilmelidir (Şekil 2.7). Sac levhaları uzun bir bıçağın tek bir düşey hareketiyle kesen gerçek giyotin tezgahı da kesme masasında cam levhaları elmasla çizip, çizginin zayıflattığı yerden kırarak ikiye ayıran kavramsal bir giyotinde, malzemeyi boydan boya kesmektedir [13].



Şekil 2.7: Kesme Planı a) Giyotinsiz Kesme b) Giyotinli Kesme [13]

Kademeli Kesim: İki-boyutlu malzeme kesme problemlerinde kademeli kesme işleminin modellenmesi Gilmore ve Gomory' nin (1965) çalışmalarıyla başlamıştır . Bu kesme işlemi giyotinle kesmenin özel bir durumudur. İlk aşamada levha, birkaç alt parçaya ayrılır. İkinci aşamada, bu alt parçalar da zıt yönde kesilip daha alt düzeydeki parçalar elde edilir [15].

Döndürme: SKP türüne göre kesilecek olan parçanın döndürülmesine izin verilebilir. Uygulamalarda ahşap malzeme, kumaş gibi akış yönü olan parçalarda döndürülmeye izin verilmez. Deri, kaplamasız sunta gibi malzemelerde döndürmeye izin verilir.

2.3.2 Stok Kesme Problemlerinde Dikkate Alınan Amaçlar

Stok kesme problemlerinde amaç maliyeti minimize etmektir. Dolayısıyla amacımız fire miktarının en aza indirilmesidir. Bu sayede malzememizi daha verimli şekilde kullanıp maliyet avantajı elde edebiliriz. Bazen parçalara öncelik değeri atamamız gerekebilir. Örneğin uzun süre depoda bekleyen bir ana malzemenin değerinin azaltılması veya acil öncelikli bir parçanın değerinin artırılması durumlarını modele yansıtmak amacıyla bu değerin bir katsayıyla (ağırlık) çarpılması kullanıcılar tarafından talep edilebilmektedir [17].

Bazı uygulamalarda, daha sonra kullanılmak üzere, mümkün olduğu kadar büyük bir fire elde etmek amacıyla bir kutudaki kullanılmayan alanların en büyüklenmesi olarak adlandırılan ikinci bir amaç dikkat çekicidir [13].

Makine kapasite kullanımının yüksek olması, işçilik oranının düşük olması, kesme kalitesinin yüksek olması diğer benimsenebilir amaçlardır.

2.3.3 Stok Kesme Probleminin Matematiksel Modeli

Stok kesme problemlerinde stok kesme malzemesi belirli ölçülerde tedarik edilebilmektedir. Ana malzemeden kesilecek olan parçaların ana malzemenin sınırları içerisinde olması gerekmektedir. Kesim safhasının amacı ise ham malzemenin en verimli biçimde kullanılması olarak tanımlanmaktadır.

SKP ile ilgili çalışmalar hızla yaygınlaşmasına karşın problemin karmaşıklığı nedeniyle bu türdeki problemleri polinomial zamanda çözen her durumda geçerli bir çözüm henüz bulunamamıştır [19].

Küçük dikdörtgen parçaların, dikdörtgen bir ana malzemeden kesilmesi problemi için Beasley tarafından önerilen matematiksel model aşağıda yer almaktadır [1]:

S_0 : Dikdörtgen ana malzeme,

L_0 : Dikdörtgen ana malzemenin boyu,

W_0 : Dikdörtgen ana malzemenin eni,

L_i : i. dikdörtgenin boyu,

W_i : i. dikdörtgenin eni,

n : Sipariş parçalarının sayısı,

v_i : i. dikdörtgenin fayda değeri,

P_i : i. dikdörtgenin S_0 ' dan kesilebilecek en küçük sayısı,

Q_i : i. dikdörtgenin S_0 ' dan kesilebilecek en büyük sayısı,

z_{ip} : Eğer i. dikdörtgenin p. kopyası ($p = 1,\ldots,Q_i$) S_0' dan kesilirse 1' e, aksi durumda sıfıra eşittir.

x_{ip} : i. dikdörtgenin p. kopyasının merkezinin x koordinatı,

y_{ip} : i. dikdörtgenin p. kopyasının merkezinin y koordinatı,

α_{ij} : i. ve j. dikdörtgenlerin boyları toplamının yarısı ( )

β_{ij} : i. ve j. dikdörtgenlerin enleri toplamının yarısı ( )

Olmak üzere karar modeli:

enb ($|xip - xjq| - \alpha ij$, $|yip - yjq| - \beta ij$)Zix i , j = 1,…,n;

p, q = 1,…, Q_i

$(i \neq j$ veya $p \neq q)$, **(1)**

z_{ip} =1, $i = 1,\ldots,n;\ P_i > 0;\ p = 1,\ldots, P_i$ **(2)**

$L_i / 2 \leq x_{ip} \leq L_0 - L_i / 2$, $i = 1,\ldots,n;\ p = 1,\ldots, Q_i$ **(3)**

$W_i / 2 \leq y_{ip} \leq W_0 - W_i / 2$, $i = 1,\ldots,n;\ p = 1,\ldots, Q_i$ **(4)**

$z_{ip} \in (0,1)$, $i = 1,\ldots,n;\ p = 1,\ldots, Q_i$ **(5)**

Amaç Fonksiyonu

$$\text{Enb} \sum_{i=1}^{n} \sum_{p=1}^{Qi} v_i . z_{ip}$$

(6)

Amaç fonksiyonu toplam faydayı en büyüklemek veya

$$\text{Enk} \sum_{i=1}^{n} \sum_{p=1}^{Qi} v_i . z_{ip}$$

(7)

amaç fonksiyonu toplam faydayı en küçüklemektir.

Kısıt 1, S_0' dan kesilmek üzere seçilen herhangi iki parçanın birbirleriyle çakışmamalarını sağlamaktadır.

Kısıt 2, i. parçanın ilk P_i kopyasının S_0' dan kesilmek üzere keyfi olarak seçildiğini göstermektedir.

Kısıt 3 ve 4, herhangi bir i parçasının p. kopyasının, ana malzeme sınırlarının içinde kalmasını sağlamaktadır.

Son olarak kısıt 5, modelde yer alan z_{ip} değişkenine sadece 0 veya 1 değeri atanabileceğini göstermektedir.

3. STOK KESME PROBLEMLERİNE ÇÖZÜM YAKLAŞIMLARI

3.1 Yerleştirme Algoritmaları

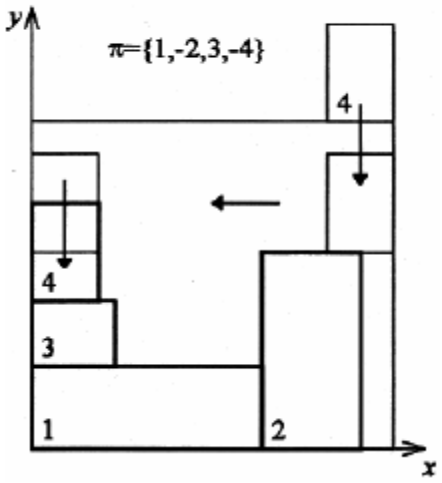
SKP' de genelde iki aşamalı çözüm yöntemi geliştirilmiştir. Birinci aşamada yerleştirilecek olan stok malzemesinin sırası belirlenir. İkinci aşamada ise kullanılacak olan yerleştirme algoritmasına göre bu malzemeler ana stok malzemesine yerleştirilir. Literatürde SKP için çok sayıda yerleştirme algoritması mevcuttur. Bu bölümde aşağı sol algoritması, dinamik algoritma, fark işlem algoritması ve temas yüzeyi algoritmasından bahsedilmiştir.

3.1.1 Aşağı Sol Algoritması (Bottom Left Algorithm)

Jakobs tarafından geliştirilen BL algoritmasında, stok malzemesi başlangıçta ana malzemenin en aşağı ve en solunu dolduracak şekilde yerleştirilir. Daha sonra kesilecek olan parçalar permütasyon sırasına göre yerleştirilmeye devam edilir. Algoritma işleyişi aşağıdaki şekilde verilmiştir [20]:

Adım 1: Yüksekliği L ve genişliği W olan büyük dikdörtgen ve genişliği w_i ve yüksekliği l_i ve âdeti k_i olan küçük dikdörtgenleri belirle.
Adım 2: Küçük dikdörtgenleri rasgele veya belirtilen sıralama şekline göre sırala.
Adım 3 : i = 1
Adım 4 : Öncelik sırasına göre ilk stok malzemesini ilk sol alt boş köşeye al ve i. dikdörtgenin oraya yerleşip yerleşmediğini kontrol et yerleşiyor ise alana yerleştir.
Adım 5 : i = i + 1 .
Adım 6 : Eğer i ≤ kutu sayısı ise adım 4' e git değil ise adım 7' ye git.
Adım 7 : Bitir

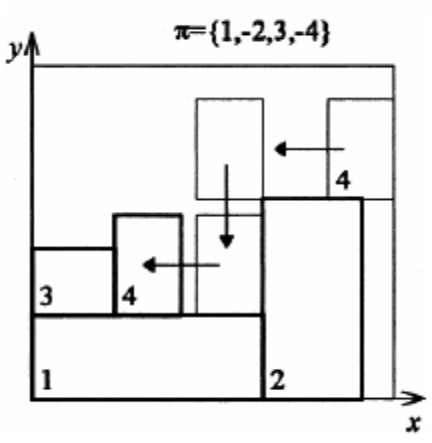
BL algoritmasını Şekil 3.1' de gösterilecek olursa;



Şekil 3.1: BL algoritması gösterimi [20]

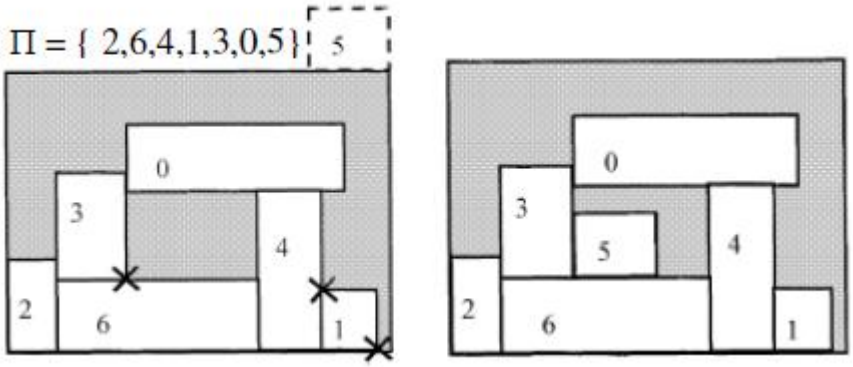
Yerleştirilecek olan n adet parça için BL algoritması ile hesaplanan yerleşim planları sayısı en çok $2^n.n!$' dir. Örneğin n=15 için en fazla $2^{15}.15!$ = 1307674368000 yerleşim planı vardır. Bu da problemin karmaşıklığını ve arama uzayının önemini ortaya koyar. Ayrıca BL algoritmasının hesapsal karmaşıklığı $O(n^2)$ ile ifade edilmektedir [8].

Geliştirilmiş aşağı sol algoritmasında ise kaydırma tekniği kullanılmaktadır. Bu algoritmada stok malzemesi aşağıya öncelikli yerleştirilir. Daha sonra parça solda bir başka parça veya ana stok malzemesinin yüzeyiyle karşılaşıncaya kadar hareket ettirilir [21]. Şekil 3.2' de gösterilecek olursa;



Şekil 3.2: Geliştirilmiş aşağı sol algoritması gösterimi [21]

Aşağı sol dolgu (Bottom Left Fill) algoritmasında ise daha yoğun bir şekilde boşluk doldurma gerçekleştirilir. En büyük dezavantajı $O(n^3)$ hesaplama yüküdür [8]. BLF algoritmasını Şekil 3.3' de gösterilecek olursa;



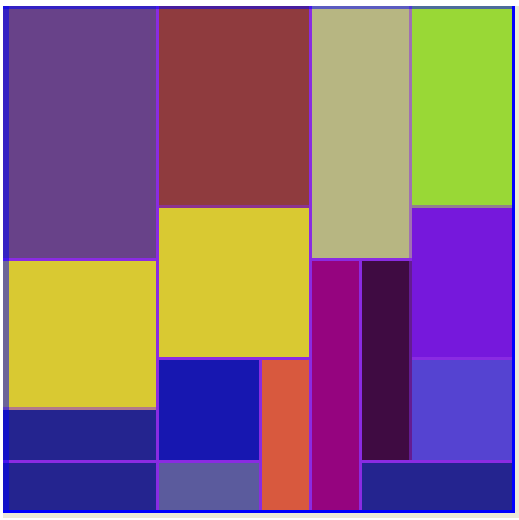
Şekil 3.3: BLF algoritması gösterimi [8]

BLF algoritması bir örnekle gösterilecek olursa;

Tablo 3.1 Örnek BLF algoritması için kesilecek malzeme boyutları ve adetleri [18]

Genişlik	Yükseklik	Adet
50	30	1
40	30	1
50	20	1
30	30	2
40	20	1
30	20	1
50	10	1
20	20	2
40	10	1
30	10	4
20	10	1

Ana stok malzemesi 100 x 100 boyutlarında olsun. Mevcut problem bu algoritma ile çözülürse;



Şekil 3.4: Mevcut problemin BLF algoritması ile çözülmesi [18]

3.1.2 Dinamik Programlama

Dinamik algoritma BL algoritmasına göre çok daha karmaşık ve hesaplama süresi daha uzundur. Bir parçanın boşluklara yerleştirilmesinin birçok olası yolu olduğundan dolayı boşlukları saymak zordur. Bunun yanında boşlukları doldurduğu için daha iyi yerleşim planlarının oluşmasını sağlamaktadır [20]. Başlangıçta ana malzemenin boyutu boşluk olarak alınır. Daha sonra parçalar yerleştikçe kalan boşluklar x ve y ekseninde sürekli güncellenir. Bu boşluklara en uygun parçalar permütasyon sırasına göre aranıp yerleştirme yapılır. DP algoritması basit bir örnekle gösterilirse;

Tablo 3.2: Örnek DP algoritması için kesilecek malzeme boyutları ve adetleri

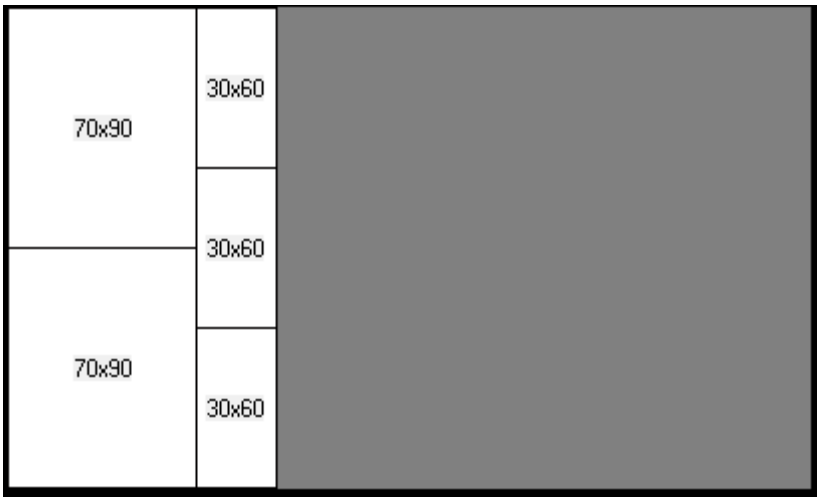
Genişlik	Yükseklik	Adet
90	70	2
60	30	3
40	20	3
60	20	1

L=180 olsun.

İlk olarak l = 90 w = 70 olan 2 adet parçamızı L = 180’ i tamamlayacak şekilde yerleştirilsin.

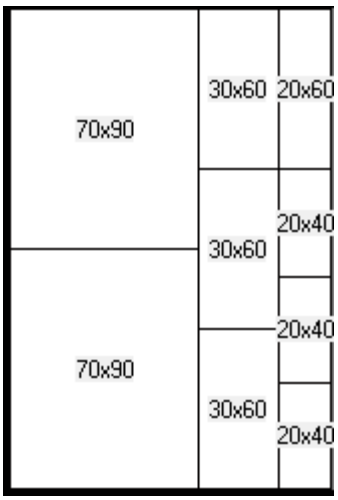
Şekil 3.5: Tablo 3.2 verilerine göre birinci adımda oluşan yerleşim planı

İkinci olarak l = 60 w = 30 olan 3 adet parçamızı L = 180’ i tamamlayacak şekilde yerleştirilsin.



Şekil 3.6: Tablo 3.2 verilerine göre ikinci adımda oluşan yerleşim planı

Üçüncü adımda l = 40 w = 20 ‘den 3 adet ve l = 60 w = 20 ‘ den 1 adeti yerleştirilecek olursa;

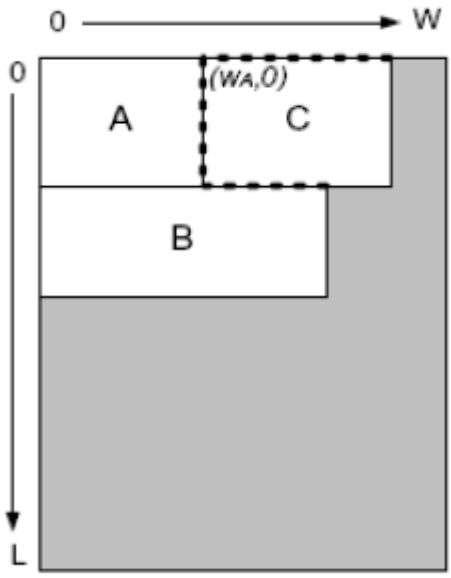


Şekil 3.7: Tablo 3.2 verilerine göre üçüncü adımda oluşan yerleşim planı

Sonucuna ulaşılmış olur.

3.1.3 Temas Yüzeyi Algoritması

Temas yüzeyi algoritmasında permütasyon sırasına göre ilk parça ana malzemenin sol alt köşesine yerleştirilir. Daha sonra ise temel amaç yerleştirilecek parçaların temas yüzeyi en yüksek olanı seçip yerleştirmektir. Temas edilen yüzey skor olarak adlandırılır. Örneğin Şekil 3.8'de görüldüğü gibi C parçası (wA,0) noktasına yerleştirilmek istensin. Bu nokta için C parçasının skoru kesik koyu ile belirtilen çizginin uzunluğunun toplamı yani $l_C + w_C + (w_B \_ w_A)$'dır [22].



Şekil 3.8: Temas yüzeyi algoritması yerleştirme örneği [22]

3.2 Analitik Metodlar

Kesin optimal sonuçları üretmeyi hedefleyen ve bunu yüksek oranda başaran algoritmik metotlar, stok kesme probleminin büyüklüğüne, detayına bağlı olarak değişmekle birlikte uzun süren uygulama zamanına ihtiyaç gösterebilmektedir. Saatlerce ve günlerce süren uygulama zamanı ihtiyacı bu yöntemlerin temel eksikliği olmakla birlikte bu tür çalışmalara çoğu kez rastlanabilmektedir.

Kullanılan analitik yöntemler; doğrusal programlama, tamsayılı programlama, dinamik programlama, dal sınır algoritması ya da tamsayılı programlama ile doğrusal programlamanın bileşimi gibi analitik yöntemlerin bileşiminden oluşan çözüm yöntemleridir.

Doğrusal programlara dayalı olarak geliştirilen yöntemlerin çoğu Gilmore ve Gomory' nin (1963) bir boyutlu kesme problemi için geliştirdiği "Sütun Oluşturma Tekniği" temel alınarak bazı ek algoritmalar geliştirmişlerdir. Bu teknikte, her sütun bir kesme planını temsil etmekte ve sırt çantası problemi çözülerek her tekrarda temele girecek olan sütun hesaplanmaktadır. Doğrusal programlama problemleri grafik ve simpleks yöntemlerden biri ile çözümlenebilir [3].

Tamsayılı doğrusal programlama, değişkenlerinden bazılarının veya tamamının tamsayılı (ya da kesikli) değerler aldığı bir doğrusal programlama türüdür. Doğrusal programlama modeli ile tamsayılı doğrusal programlama arasındaki fark, doğrusal programlama modelinde karar değişkenlerinin sıfır ve sıfırdan büyük olma koşulu aranırken, tamsayılı doğrusal programlama da değişken değerlerinin sıfıra eşit ve sıfırdan büyük tamsayı almaları şartının istenmesidir [23].

Tam sayılı doğrusal programlama çözümleri için, dal-sınır ve kesme düzlemi olarak iki yöntem geliştirilmiştir. İki yöntemden hiçbiri tam sayılı doğrusal programlama problemlerinin çözümünde sürekli daha iyi sonuç verememektedir. Bununla birlikte, deneyimler dal-sınır yönteminin kesme düzlemi yöntemine göre çok daha başarılı olduğunu göstermektedir [23].

3.3 Sezgisel Metodlar

Heuristic kelimesi Yunanca kökenlidir ve bulmak keşfetmek anlamına gelmektedir. Bir çözüm tekniği olarak sezgisel, herhangi bir şeyin bulunmasını garanti etmeyen bir arama metodu olarak tanımlanmaktadır.

En genel anlamda, sezgisel, iyi yani en iyi çözüme yakın çözümleri araştıran bir tekniktir ve bu işlemi fizibiliteyi ya da en iyi sonucu bulmayı garanti etmeksizin makul bir hesaplama maliyeti ile yapar [24].

İzleyen kısımda bu metodların SKP için oluşturulan yapılarından bahsedilecektir.

3.3.1 Lineer Programlama Prosedürü

Lineer programlama prosedürü, algoritmik ve sezgisel metotların kombinasyonu olup stok kesme probleminin lineer programlama gevşetmesinin çözümü için kullanılan bir metottur.

Çözüm uzayında bulunan yerleşim planlarının sayısının çok fazla olması ile problemin çözümünün hesaplanması oldukça zor hale gelebildiği durumlar ile

karşılaşabilinmektedir. Bu yüzden bu tür durumlarda probleme ait tüm olası yerleşim planlarını kullanmadan çözüme ulaşma ihtiyacı doğabilmektedir. Bunun için sütun oluşturma yönteminden faydalanılır. Sütun oluşturma yöntemi uygulaması sırasında yeni eklenecek yerleşim planlarının bulunmasında sırt çantası probleminin algoritması kullanılır [23].

3.3.2 İlk Uygun Azalan Sezgisel Metot

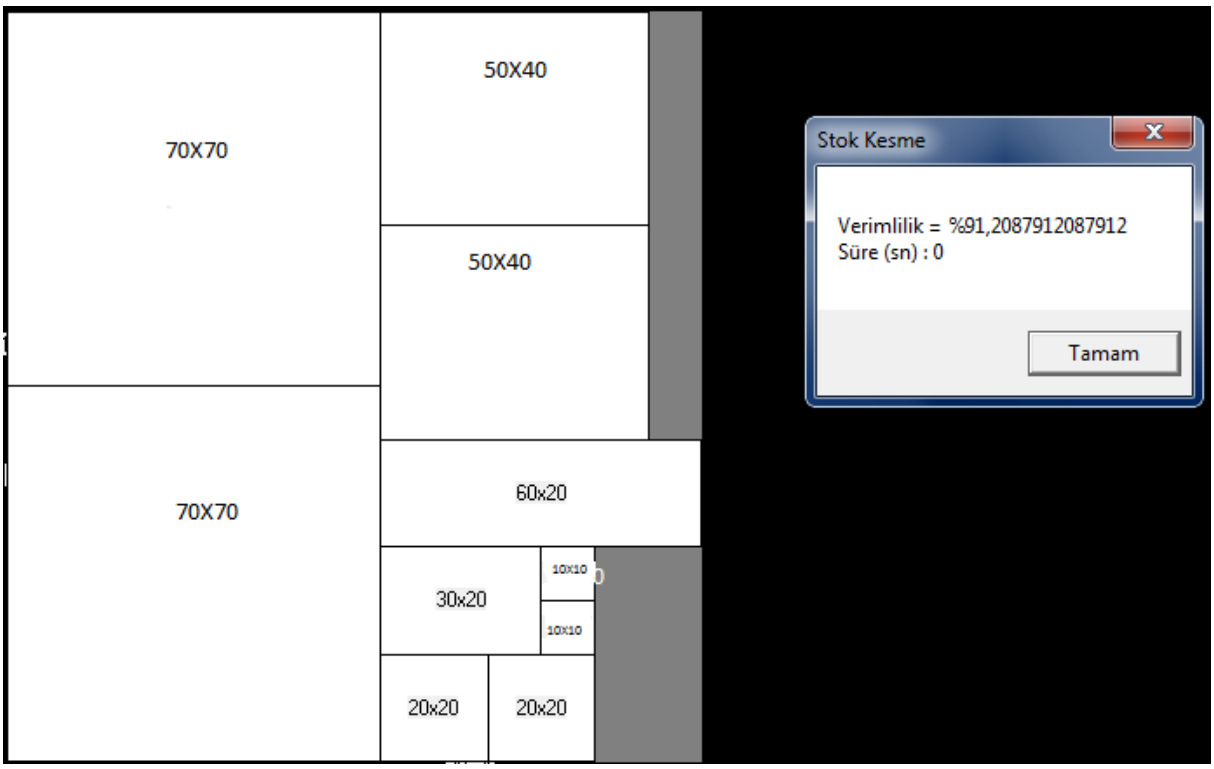
İlk uygun azalan sezgisel metodun uygulaması şu şekildedir. Öncelikle yerleştirilecek stok malzemeleri alanlarına göre büyükten küçüğe doğru sıralanır. Daha sonra ise ana stok malzemesine alanı en büyük olan yerleştirilmeye çalışılır. Eğer yerleştirilebiliyorsa bir sonraki alanı büyük malzemenin yerleştirilmesi yapılır. Yerleştirme yapılırken yerleştirme algoritmalarından biri seçilir. Problemin türüne göre talep karşılaşılanana kadar veya ana stok malzemesine yerleştirilecek boşluk kalmayıncaya kadar süreç devam ettirilir [23]. Örnek verilecek olursa;

Tablo 3.3: Örnek İlk uygun azalan sezgisel metot için malzeme boyutları ve adetleri

Parça Numarası	Genişlik	Yükseklik	Adet
1	70	70	2
2	50	40	2
3	30	20	1
4	60	20	1
5	20	20	2
6	10	10	2

Ana malzememizin boyutları 130 x 140 olsun ve yerleştirme algoritması olarak BLF algoritması kullanılsın;

Alanlarına göre sıralarsak $A_1>A_2>A_4>A_3>A_5>A_6$



Şekil 3.9: İlk uygun azalan sezgisel metoda göre oluşturulan programla çözümü

3.3.3 İlk Uygun Ardışık Sezgisel Metot

Ardışık sezgisel prosedür yaklaşımında, stok malzemelerinden üretilmesi istenen küçük ebatlardaki tüm siparişler karşılanıncaya kadar yerleşim planlarının teker teker oluşturulmasıyla çözüm elde edilir. Ardışık sezgisel prosedür ile ilgili ilk çalışma Haessler tarafından gerçekleştirilmiştir [25].

Ardışık sezgisel prosedürün lineer programlamaya göre en belirgin avantajı, fire kaybının yanı sıra diğer faktörlerin de kontrol edilmesini sağlamasıdır. Örnek olarak, yüksek yerleşim planı kullanımının araştırılması, yerleşim planlarının değişiminin azaltılması verilebilir. Diğer bir yandan ardışık sezgisel prosedür yüksek fire kayıplarına yol açabilmektedir. Bu prosedürün en belirgin dezavantajıdır. Ayrıca bu prosedür sadece tam sayılar ile çalışmasından dolayı yuvarlamalardan kaynaklanan problemleri elemiş olur. En önemli avantajı yuvarlamaya gereksinim duyulmamasıdır.

3.3.4 Hibrit Çözüm Prosedürü

Stok kesme problemlerinin çözümünde kullanılan diğer bir yöntem, yukarıda belirtilen lineer programlama prosedürünün ve ardışık sezgisel prosedürün kombine edilmesidir. Örneğin, öncelikle ardışık sezgisel prosedür ile bir sonuç elde edilir ve bu sonuç lineer programlama prosedüründe başlangıç olarak kullanılır. İlave doğrusal programlama yinelemeleri (iterasyonları) yapılarak kesme kaybı minimize edilmeye çalışılır. Yinelemeler sonrası en iyi ardışık sezgisel prosedür çözümü ve yuvarlanmış lineer programlama çözümü seçilir [23].

Diğer bir uygulama şu şekilde olabilir; optimum gölge maliyetleri elde etmek amacıyla problem ilk önce lineer programlama prosedürü ile çözülür ve daha sonra gölge maliyetler, yerleşim planlarının ardışık sezgisel prosedüründe kabul edilebilirliliğini test etmek için kullanılır. Bu yöntemde, ardışık sezgisel prosedürü sona yaklaştığında yerleşim planı seçimi kararı zor belirlenir hale gelir ve bu aşamada karşılanamayan sipariş miktarları lineer program prosedürü ile çözülür [23].

Bu yaklaşımın en önemli avantajı, lineer programlama prosedürünü ve ardışık sezgisel prosedürünü içerisinde bulundurması ve iki metodun ayrı olarak verdiği çözümlerden iyi olanı seçmesidir. Bu yaklaşım genellikle yerleşim planlarının değişiminin minimum olmasının istendiği ve firenin enküçüklenmesinin istendiği problemler literatürde karşılaşılan bir uygulamadır [25].

3.4 Bazı Metasezgisel Metotlar

Son yıllarda stok kesme problemlerinin çözümü için metasezgisel metotlara yönelinmiştir. Metasezgisel metotlar, sezgisel tekniklerle iyileştirilmiş fakat yeterli olmayan durumlarda, etkili olan, sezgisel metotların üst mertebesidir. Metasezgisel metotlar, yerel optimallikten öte çözümler üretmek için, normal sezgileri değiştiren ve onlara kılavuzluk eden amir bir taktiği işaret eder [24].

Metasezgisel metotlar üzerinde önemli gelişmeler katedilerek çok sayıda algoritmalar sunulmuştur. Metasezgisel metotların bazıları sırasıyla şu şekildedir; tabu araması, benzetilmiş tavlama algoritması, karınca kolonisi algoritması, grasp ve genetik algoritmalardır.

Metasezgisel metotlar bazı doğal oluşum proseslerini taklit etmeye çalışmaktadırlar. Benzetilmiş tavlama, termodinamik prosesleri taklit etmekte, tabu arama ise bir çeşit hafıza uygulamasını dikkate alarak zeki prosesleri taklit etmektedir. Genetik algoritmalar ise, genetik yapılara benzer bir şekilde problemleri formüle ederek doğanın en iyi olan hayatta kalır prensibini taklit etmektedirler [23].

3.4.1 Tabu Araması

Tabu arama tekniğine bağlı ilk çalışmalar Glover tarafından yapılmıştır. Teknik genel zeki problem çözme eğilimlerinden kaynaklanmaktadır ve zekice problem çözme prensipleri ortaya çıkarmaya çalışır. Tabu arama tekniğinin yapay zeka ve optimizasyon alanlarını birleştiren kavramlara dayandığı söylenebilir [24].

Tabu araştırması temel olarak belirli bir problem üzerine elde edilen ilk çözüm etrafındaki komşuluklar oluşturmaktır. Komşuluk mekanizması ele alınarak birbirini izleyen seçilmiş çözümlerden daha iyi olanı tabu listesi olarak atanır. Tabu arama algoritmasında tabu listesi olarak oluşturulan ilk aday çözüm ve değişken komşu çözüm sayısı, bir tür tabulaştırma görevi yapmaktadır. Kötü sonuç veren bölgelerde daha fazla işlem yapılmaması (bu elemanların komşu sayılarının azaltılması), istenen çözüme daha az hesaplamayla, dolayısıyla daha hızlı ulaşmayla sağlamaktadır. İyi sonuç veren parametrelerin bir sonraki iterasyonda komşu sayıları artmakta, böylece algoritmanın verimliliği de artmış olmaktadır. Tabu listesinin en önemli özelliklerinden birisi, mevcut tabu listesinin aday komşu çözümler ile karşılaştırıldıktan sonra bir sıralama ve karşılaştırma işlemi yaparak kendisini yenileyebilmesidir. Bu amaçla, geliştirilen algoritmada, önceki döngülerde elde edilen çözümlerin listesinin tutulduğu bir tabu listesi oluşturulmuştur. Eğer bir komşu çözüm adayı, tabu listesinde yer alan bir çözümle aynıysa (bu durumda aday çözüm, zaten daha önce denenmiştir), bu çözüm değerlendirme dışı bırakılmaktadır. Tabu listesi oluşturulurken her döngüdeki en iyi çözüm listeye alınmakta, listenin dolduğu durumda listedeki ilk kayıtlar (başlangıçtaki çözümler) listeden atılıp, son döngüler ile elde edilen çözümler listeye alınmaktadır.

Burada tabu araştırması ile ilgili bazı kriterler ön plana çıkmaktadır. Bunlar başlangıç çözümü, komşuluk yapıları, tabu listesinin düzenlenmesi, aspirasyon kriteri, durdurma kriteri, yoğunlaşma ve çeşitlendirme taktikleridir [24].

Tabu listesinin yeni elde edilen çözümler ile liste uzunluğu artacak ve çözüm aramada büyük bir etki yaparak kısa çözümlerin daireselliği sayesinde nesnel bir çözüm arama tekniği olduğunu gösterecektir. Burada istenilen durum çözüm adaylarının mevcut bu döngüsü ile tabu listesinin geliştirilmesidir. Tabu listesine dahil edilen yeni çözümler oluşturulurken eğer elde edilen çözüm tabu listesindeki çözümlerden daha iyi ise elde edilen çözüm tabu listesine eklenebilir. Bunun tersi durumunda, tabu listesine eklenmeyecek ve doğal olarak belirli bir bellek kaplamayacaktır. En iyi çözüm bulunana kadar bu işlemler mevcut döngü ile devam edecektir [26].

Tabu algoritması bazı değişiklikler uygulanarak SKP çözümü için uygun hale getirilip, performansı değerlendirilebilir. Bu konu için üzerinde çalışılabilecek bir metasezgisel yöntem olarak gözönünde tutulmalıdır.

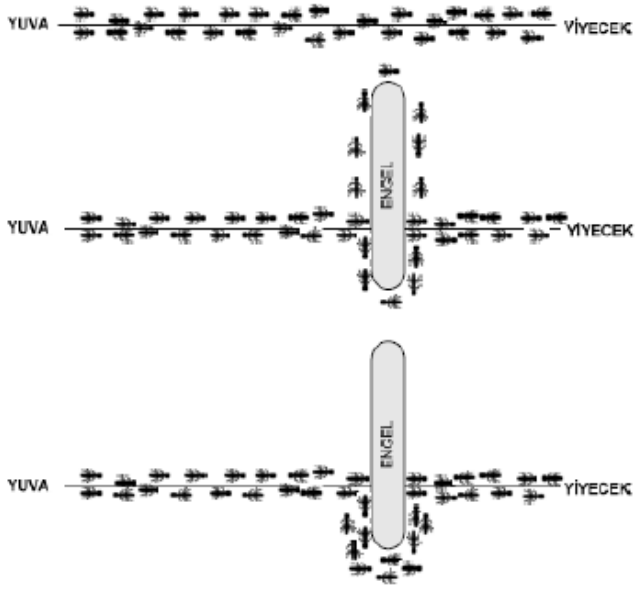
3.4.2 Karınca Kolonisi Algoritması

Karınca kolonileri metasezgiseli, doğal karıncaların yuvaları ile besin kaynakları arasında izledikleri yolların izlenmesi sonucu ortaya çıkan bilimsel gerçekler üzerine doğmuştur. Gerçek karıncalar ile ilgili deneyler Goss ve arkadaşları tarafından 1989 yılında laboratuar ortamında yetiştirilmiş karınca kolonileri üzerinde yapılmıştır. Bu çalışmalarda elde edilen sonuçlar; çoğu karınca türü neredeyse kördür, karıncalar yuvalarından yiyecek kaynağına veya tersi yönde hareket ederlerken geçtikleri yollara feromen adı verilen bir tür kimyasal madde bırakmaktadırlar. Karıncalar bir yol seçmeleri gerektiği zaman bu seçimi alternatif yollar üzerine bırakılmış olan feromen madde yoğunluğuna göre belirlemektedirler ve karıncaların bu hareketleri merkezi bir kontrol ile sağlanmamaktadır şeklinde özetlenmiştir. Bir karınca bir yiyecek kaynağı bulduğu zaman kaynağın kalitesini veya miktarını değerlendirir ve bir miktar yiyecek alarak yuvasına geri döner. Bu geri dönüş sırasında, bulduğu yiyecek kaynağının kalitesi veya miktarıyla doğru orantılı olacak şekilde kullandığı yola feromen

maddesi bırakır. Böylece diğer karıncalar bu yolun sonundaki yiyecek kaynağının kalitesi veya miktarı konusunda bilgi sahibi olurlar. Yuvaya yakın kaynaklara ulaşmak daha kolay olacağı için bu bölgelerde feromen maddesinin yoğunluğu daha fazla olacaktır [23].

Doğal karıncaların bu davranış kalıplarının, özellikle en kısa yol problemleri olmak üzere, pek çok kombinatoriyel optimizasyon probleminde kullanılabileceği ilk kez 1992 yılında Marco Dorigo ve arkadaşları tarafından ortaya atılmıştır . Karıncaların doğal hareketleri ile kombinatoriyel optimizasyon problemlerinin uyuşan karakteristik özellikleri; gerçek karıncaların arama alanı ile kombinatoriyel problemlerin mümkün sonuçlar kümesi, bir kaynaktaki yiyecek miktarı ile amaç fonksiyonu, feromen madde ile hafızadır. Karınca kolonileri metasezgiselinden türetilmiş ve çeşitli problemlerin çözümünde kullanılan çok sayıda algoritma vardır. Bu algoritmalar formülasyon olarak birbirinden ayrılmakta fakat hepsi karınca kolonileri metasezgiselinin ortak özelliklerini kullanmaktadır [23].

Şekil 3.10‘ da zaman geçtikçe tüm karıncaların en kısa olan yolu kullandıkları görülmektedir. Bunu yaparken önceki geçişlerden yollarda kalan feromon izlerinden faydalanmaktadırlar. Temel kural, feromon miktarının yoğun olduğu yolun tercih edilme olasılığının yüksek olmasıdır.



Şekil 3.10: Gerçek karıncaların en kısa yolu bulma aşamaları [26]

Karınca koloni algoritması bazı değişiklikler ile SKP çözümü için uygulanabilir hale getirilebilir. Performansı diğer metasezgisel yöntemlerle kıyaslanabilir. Bu nedenle Karınca koloni algoritma SKP çözümü için alternatif bir yöntem olarak değerlendirilebilir.

3.4.3 Benzetilmiş Tavlama Algoritması

Tavlama Benzetimi; kombinatoryal optimizasyon problemlerinde kullanılan olasılıklı bir algoritmadır. Bu genel amaçlı arama algoritmasında genel iyileştirmeler gerçekleştirilerek, yerel optimum tuzağına düşmeden (gerektiğinde düşük kaliteli çözümler kabul edilerek) başarılı sonuçlar elde edilir. Algoritmaya "Tavlama Benzetimi" isminin verilmesi bir rastlantı değildir. "Tavlama" sözcüğü katıların fiziksel tavlama sürecine benzerliğinden ileri gelmektedir.

Tavlama işlemi parçaların kararlı yapıda olmalarını sağlamak için gerçekleştirilir. Maddelerin kristal yapısındaki bozuklukların düzeltilmesini sağlar. Böylece minimum enerjili durum sağlanmış olur. Tavlama süreci iki aşamada gerçekleşir. Öncelikle sıcaklık katının eriyebileceği değere kadar arttırılır. İkinci aşamada ise sıcaklık uygun şekilde (yeteri kadar yavaş) azaltılır. Böylece minimum enerji durumu elde edilir. Bu durum sıcaklığın bulunduğu ve soğutmanın da yeteri kadar yavaş yapıldığı durumlarda gerçekleşir [13].

Matematiksel analizi yapılamayan modellerin durumunda, eşitlikleri bulmak için benzetimlerden yararlanılır. Bilgisayarın ilk yıllarında bile çeşitli derecelerde sistemlerin benzetiminin yapılabileceği görülmüştür. Örneğin bu hareketli parçalardan oluşan bir sistem olabilir. Hareketli parçalar bir kutudaki küreler olsun. Küreler uzaklık ve azalan bir kuvvetin etkisinde hareket ederler. Sistemin enerjisi parçalar arasındaki uzaklıkla hesaplanabilir. Benzetim algoritmasının her adımında rastgele seçilen bir parça, mevcut durumdaki parçayla yer değiştirilir. Eğer yeni durum enerji durumundakinden daha az ise yer değiştirme kabul edilir. Bu temel adım tanımlandığı kadar tekrarlanır. Metropolis adımı denilen adımın tekrarlanmasıyla oluşan prosedüre Metropolis Döngüsü adı verilir [13].

Yapılan deneylerde düşük sıcaklıkların yer durumunu garanti etmediği görülmüştür. Maddeyi yer durumuna taşıyacak olan teknik "tavlama" olarak adlandırılır. Maddenin erime durumundan başlayarak sıcaklık yavaş yavaş düşürülür. Sıcaklık çok hızlı düşürüldüğünde maddenin kristal yapısında kusurlar görülecektir.

Benzetilmiş tavlama algoritması en iyileme problemlerinin çözümü için son yıllarda sıkça başvurulan bir arama yöntemidir. Bu yöntemde amaç, problem için uygun çözümler ararken zayıf bir çözümde takılıp kalmayı engellemektedir.

Arama süreci çoğunlukla rasgele seçilen bir başlangıç çözümüyle başlar ve arama uzayında seçilen başlangıç çözümüne yakın başka bir çözüm üretilir. Bu iki çözüm maliyetleri arasındaki değişim miktarı (ΔC) hesaplanır. Aşağıdaki formülde gösterilecek olursa;

C_i = i. Ötelemedeki çözüm

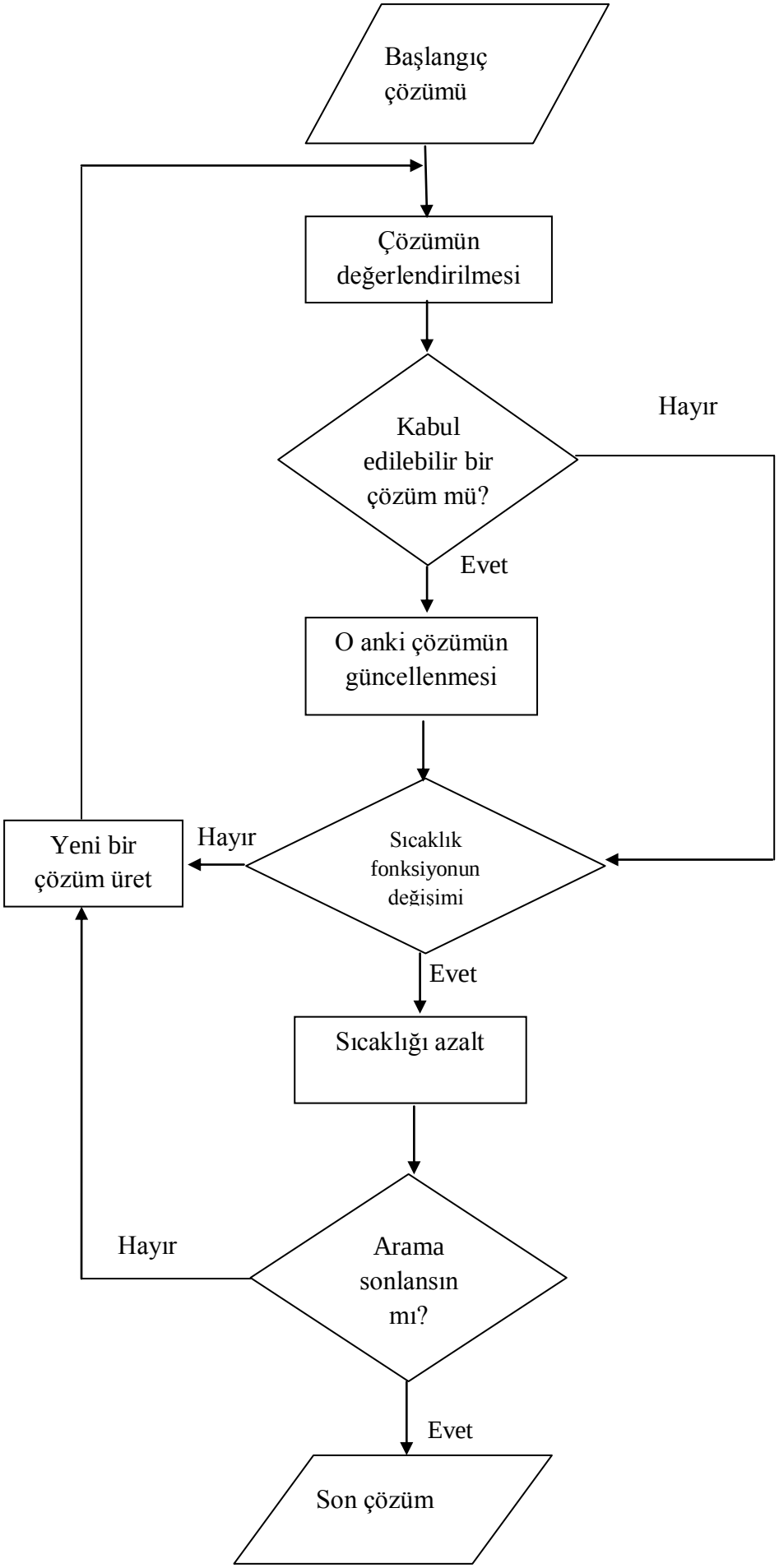
C_{i-1} = O anki çözüm

$$\Delta C = C_i - C_{i-1}$$

Genel bir yerel arama süreci için maliyette bir azalma gerçekleşiyorsa, o anki çözüm yeni üretilmiş diğer çözümle değiştirilir. Aksi durumda, o anki çözüm aşağıdaki denklemle ifade edilen bir formüle göre belirli bir olasılıkla saklı tutulur [20].

$$exp(-\Delta C / T) > R$$

Formülde ΔC çözümlerin maliyetleri arasındaki değişim miktarıdır. R [0,1] arasında üretilmiş düzgün dağılımlı rasgele bir sayıdır. T sıcaklık olarak bilinen kontrol parametresidir. Başlangıçta göreceli olarak yüksek bir sıcaklıktır. Bundan dolayı yeni çözümlerin kabul edilme şansı oldukça yüksektir. Daha sonra tavlama süreci boyunca önceden belirlenmiş bir soğutma planına göre yavaş yavaş azaltılır. Bu azaltmayla, algoritma yeni çözümleri kabul ederken çok daha seçici olur. Tavlama süreci, o anki çözümün yakınlarındaki diğer çözümlerde başka bir değişime rastlanıncaya kadar tekrarlanır. Bu durumda yerel arama algoritması, yerel en küçük bir çözümle sonlandırılır. Kombinasyonel en iyileme problemlerinin çözümündeki bu tavlama sürecini gösteren benzetilmiş bir tavlama algoritmasının akış diyagramı Şekil 3.11' de gösterilmiştir [27].



Şekil 3.11: Benzetilmiş tavlama algoritmasının akış diyagramı [28]

Tavlama benzetimi algoritmasının adımları gösterilecek olursa;

Adım 1. Uygun 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 sayacı sıfırla: n = 0 ve devam et

Adım 3. i' nin bir komşusu olan j durumunu rassal 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.

Tavlama Planına Ait Parametreler

Sıcaklık parametresinin azalan değerlerinin sonlu bir sırası için, sonlu uzunluklu homojen Markov zincirleri üretilerek oluşturulan bir tavlama benzetimi algoritmasının uygulaması, sonlu zaman alacaktır. Yakınsamanın sağlanabilmesi için bu algoritmada kullanılan parametreler kümesi uygun bir şekilde belirlenmek zorundadır. Tavlama benzetimi algoritmasında kullanılan parametrelerin ve değerlerinin belirlenmesi tavlama veya soğutma planı olarak tanımlanmaktadır. Tavlama planı ile aşağıdaki parametreler belirlenmektedir [24].

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 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 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 komşu yapısı boyutla orantısal alınabilmektedir. Bu tavlama planı ile, sıcaklık parametresinin her değerinde elde edilen çözüm, belli sayıda ardıl sıcaklık değişimleri boyunca aynı kalırsa tavlama benzetimi algoritması durdurulmaktadır. Buna göre elde edilen son durum, fiziksel tavlamadaki donma durumuna karşılık gelmektedir [24].

Benzetilmiş tavlama kesme paketleme algoritmaları kullanılması durumuna ait performans değerlendirmesi SÖKE (2006) makalesinde sunmuştur [13].

3.4.4 Açgözlü Rassallaştırılmış Uyarlamalı Arama

Açgözlü rassallaştırılmış uyarlamalı arama yordamı (GRASP) kombinatoryal en iyileme problemleri için kullanılan metasezgisel bir metot olup, her bir tekrarda probleme ilişkin uygun bir çözümün hesaplandığı, arama uzayının tekrarlı rassal örneklenmesine dayalıdır. Her bir GRASP tekrarlaması (iterasyonu) iki ana aşamadan oluşmaktadır: Rassallaştırılmış sezgisel temeldeki oluşturma aşaması ve ilk aşamada oluşturulan başlangıç çözümü iyileştiren yerel arama aşamasıdır. Tekrarlama süreci belirtilen bir tekrar sayısına ulaşıncaya kadar sürdürülür ve o zamana kadar elde edilen en iyi çözüm son çözüm olarak değerlendirilir. GRASP ilk kez Feo ve Bard tarafından tanıtılmıştır [29].

GRASP algoritmasının temel adımları aşağıda gösterilecek olursa;

Adım 1. Girdi veriyi oku.

Adım 2. j←1, i←1

Adım 3. i ≤ EnbTekrar olduğu sürece, adım 4'e git, aksi durumda adım 10'a git

Adım 4. Açgözlü Rassallastırılmış Oluşturma

Adım 5. Yerel arama

Adım 6. Eğer i=j*TekrarSayısı ise Rota Yenileme gerçeklestir ve j' yi 1 arttır.

Adım 7. En iyi çözümü güncelle.

Adım 8. i' yi 1 arttır.

Adım 9. Adım 3' e git.

Adım 10. Mevcut en iyi çözümü göster ve DUR.

3.4.4.1 Oluşturma Aşaması

Çözümler, uygunluğu daima sağlayacak şekilde, açgözlü rassallaştırılmış yordam yardımıyla oluşturulmaktadır. Aday işler kümesi, henüz sıralanmamış ve olası bir sırada bulunabilecek tüm işleri kapsamaktadır. Burada, başlangıçta sıralanacak olası işlerin toplam sayısının toplam talebe eşit olduğunu not etmek gerekmektedir. Bir çözüm oluşturmak için ilk iş aday işler içinden rassal olarak seçilir ve bu iş sıraya konur. Bu tek işlik sıra geçerli sıra olarak kabul edilir. Sıranın sonraki işleri açgözlü yordam yardımıyla seçilmektedir. Bu yordama göre, aday işlerin her biri için geçerli sıra ile oluşturulabilecek farklı sıralar, uygunluğu da dikkate alarak, ileriye kaydırma metodu yardımıyla türetilmektedir. Bu metot rassal başlangıç sıra oluşturmaya kıyasla daha fazla süre gerektirmekte ancak daha iyi başlangıç çözümler oluşturmaya imkan sağlamaktadır. Kısaca metot şu şekilde çalışmaktadır: Geçerli sıranın "ABC" olduğunu ve aday işin de D olduğunu varsayalım. D işi ile ileriye kaydırma metodu yardımıyla, oluşturulabilecek alternatif sıralar şunlardır; "DABC", "ADBC", "ABDC" ve "ABCD" . Burada D işinin, geçerli sıra dikkate alındığında, soldan sağa (ileriye) doğru sırasıyla kaydırıldığını not etmek gerekmektedir. Tüm aday işler için uygun alternatif sıralar türetildikten sonra, bu aday işler ve ilgili uygun sıralarından "Sınırlandırılmış Aday Listesi (Restricted Candidate List-RCL)" oluşturulmaktadır. Amaç, kullanım oranını en küçüklemek olduğundan, tüm alternatif sıralardan en küçük kullanım oranına sahip sıra ile ilişkili

iş aslında sıralanacak bir sonraki en iyi iştir, ancak bu şekilde seçim yapısı tam bir açgözlü tip sezgiseli doğurmaktadır. Bunun yerine, çalışmada, en iyi kullanım oranlarına sahip sıralarla ilişkili işlerden sınırlandırılmış aday listesi oluşturulmaktadır. *Uenk* ve *Uenb'* nin açgözlü yordam yardımıyla hesaplanan en küçük ve en büyük kullanım oranları olduğu varsayılsın. Sınırlandırılmış aday listesi içinde yer alacak iş sayısı, olmak üzere bir parametresi yardımıyla sınırlandırılabilmektedir: değerine bağlı olarak, kullanım oranları, *Uenk* + (*Uenb* – *Uenk*) Uenk değerine eşit ya da küçük olan sıralara ilişkin tüm aday işler sınırlandırılmış aday listesi içinde yer almaktadır. değeri 1' e eşit olduğunda tüm aday işler sınırlandırılmış aday listesine dahil edilir ki bu rassallaştırılmış oluşumla sonuçlanmaktadır. Diğer yandan, değerinin 0 olması halinde ise, sınırlandırılmış aday listesine içinde en küçük kullanım oranlı sıra ile ilişkili iş bulunur. Bu durum da tam açgözlü oluşumla sonuçlanmaktadır. Sınırlandırılmış aday listesi oluşturulduktan sonra, sıralanacak sonraki iş, kullanım oranlarının dikkate alınması ile sınırlandırılmış aday listesi içinden rassal olarak seçilmektedir. Seçim süreci, aynı zamanda, seçilen işle ilişkili sırayı da belirlediğinden, bu sıra geçerli iş sırası olarak kabul edilir ve aday işler kümesi güncellenir. Bu yordam aday işler kümesinde hiçbir iş kalmayıncaya yani uygun bir çözümün oluşturulmasına kadar devam ettirilmektedir. Olasılık dağılımına göre α 'nın alabileceği olası değerler; 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0'dır [23].

Oluşturma Aşamasının Adımları

0: İlk işi rassal olarak seç ve sıraya koy. Bu sırayı geçerli sıra olarak kabul et.

1: Aday işler kümesini güncelle.

2: Aday işler (henüz sıralanmamış olanlar) için olası uygun sıraları ileriye kaydırma metodu ile türet.

3: Her bir aday iş için türetilen uygun sıraların kullanım oranlarını hesapla

4: Sınırlandırılmış aday listesini türet

5: Sınırlandırılmış aday listesinden rassal olarak bir iş ve ilişkili sırasını seç. Bu sırayı geçerli sıra olarak kabul et.

6: Eğer bir çözüm tamamlandı ise, Adım 7' ye git, aksi halde (henüz tüm işler sıralanmadı ise) Adım 1' e git

7: Oluşturulan çözümü geri döndür ve durdur.

3.4.4.2 Yerel Arama Aşaması

Arama rassallaştırılmış oluşturmayla, küçük komşuluklar için bile, sıklıkla en iyi çözümler türetilememektedir. Yerel arama aşaması, oluşturulan başlangıç çözümde genellikle iyileşme sağlamaktadır. Bir yerel arama algoritması komşulukta bulunan daha iyi bir çözümü geçerli çözümle yer değiştirecek şekilde tekrarlamalı olarak çalışmaktadır. Yerel arama aşamasının adımları sıralanacak olursa; [23]

Adım 1. Geçerli çözümün kullanım oranı değerini KulORN‘ a ata. Yerel bir en iyi elde edilene kadar bu çözüm üzerinde *ikili değişim komşuluğu* araması uygula. Yerel en iyiyi geçerli çözüm olarak kabul et.

Adım 2. Yerel bir en iyi elde edilene kadar geçerli çözüm üzerinde *geriye kaydırma komşuluğu* araması uygula. Yerel en iyiyi geçerli çözüm olarak kabul et.

Adım 3. Yerel bir en iyi elde edilene kadar geçerli çözüm üzerinde *ileriye kaydırma komşuluğu* araması uygula. Yerel en iyiyi geçerli çözüm olarak kabul et.

Adım 4. Geçerli çözümün kullanım oranı değeri KulORN‘ dan daha iyi ise Adım 1‘ e git, aksi halde dur.

3.4.5 Genetik Algoritma

Genetik algoritma, doğadaki evrim yöntemlerini kullanan bir arama yöntemidir. Genetik algoritma tekniği, Michigan Üniversitesi’ nde yer alan John Holland tarafından 1970’ li yıllarda ortaya çıkmış ve 1975’ te Holland Doğal ve Yapay Sistemlerin Uygulanması adlı kitabında incelemelere yer vererek yayınlamıştır. Mekanik öğrenme konusunda çalışan Holland, Darwin’ in evrim teoreminden etkilenerek canlılarda yaşanan genetik süreci bilgisayar ortamında gerçekleştirmeyi düşündü [7].

GA hem problem çözmek hem de modelleme için kullanılmaktadır. Günümüzde genetik algoritmaların uygulama alanları genişlemektedir. Bunlardan bazıları : Atölye Çizelgeleme, Yapay Sinir Ağları Tasarımı, Görüntü Kontrolü, Elektronik Devre Tasarımı, Optimizasyon, Uzman Sistemler, Paketleme Problemleri,

Makine ve Robot Öğrenmesi, Gezgin Satıcı Problemi, Ekonomik Model Çıkarma v.b sayılabilir [30].

Genetik algoritmalar, klasik optimizasyon algoritmalarından dört temel noktada ayrılır [18]:

- GA parametrelerin kendileri ile değil, parametre takımının kodlanmış bir haliyle uğraşırlar.
- GA aramaya tek bir noktada değil, bir nokta ailesinden başlarlar. Dolayısıyla yerel bir optimuma takılmadan çalışabilirler.
- GA amaç fonksiyonunun (objective function) türevlerini ve bir takım ek bilgileri değil, doğrudan amaç fonksiyonunun kendisini kullanırlar.
- GA' da deterministik değil rastlantısal geçiş kurallar kullanılır.

GA' da bir çözüm uzayındaki her noktayı, kromozom ile kodlar. Her noktanın bir uygunluk değeri vardır. Çözüm uzayındaki noktalar kümesi bir araya gelerek popülasyonu oluşturur. Her kuşakta, çaprazlama ve mutasyon gibi genetik operatörler kullanılarak yeni bir popülasyon oluşturulur. Çözüm uzayından rassal olarak seçilen noktalar üzerinde operatörler yardımıyla daha iyi noktalara ulaşmak amaçtır. Bu arama iyi sonuç elde edilemeyene kadar devam etmektedir. GA' lar ile problemlerin çözülmesinde hedeflenen sonucu üretecek özelliklerin kalıtım yolu ile başlangıç çözümlerinden elde edilen yeni çözümlere onlardan da daha sonraki çözümlere geçtiği kabul edilmektedir. Genel olarak GA' da izlenen adımlar aşağıdaki şekilde sıralanabilir: [18]

1. Uygun çözümleri simgeleyen N adet kromozomdan oluşan başlangıç popülâsyonunun oluşturulması
2. Her x kromozomu için f(x) uygunluk değerinin hesaplanması
3. Topluluktan uygunluk değerlerini dikkate alarak (uygunluk değeri daha iyi olanların seçilme olasılığı yüksek olacak şekilde) iki kromozomun seçilmesi
4. Belirli bir çaprazlama olasılığıyla ebeveynlerden gelen kromozomların çaprazlanarak yeni bireylerin oluşturulması ya da çaprazlama yapılmazsa ebeveynlerden gelen kromozomların aynen bir sonraki nesle kopyalanması
5. Yeni bireylerin belirli bir olasılığa göre mutasyona uğratılması
6. Oluşturulan bireyin yeni topluluğa eklenmesi

7. Önceki topluluğun, yeni toplulukla değiştirilmesi
8. Sonlandırma koşulu sağlandıysa mevcut topluluktaki en iyi çözümün belirlenmesi, sağlanmadıysa 2. adıma dönülmesi

Bir genetik algoritma tasarımında ilk adım, olurlu çözümlerin kromozom yapılarına karşı gelen karakter dizileriyle nasıl ifade edilebileceğinin düşünülmesi olmaktadır. Malzeme kesme probleminde de kesme planını modelleyecek şablon tasarlandıktan sonra, başlangıç popülasyonunu oluşturacak kromozomlar, bu kalıba uyacak şekilde rassal olarak türetilirler. Bunların ne kadar başarılı olduğunu belirlemek için bir de başarı ölçütünün (uyum fonksiyonu) tanımlanması gerekmektedir. Malzeme kesme problemlerinde, fire oranı bir başarı göstergesi olarak kullanılabilir.

Bu çözümlere genetik operatörler uygulandıkça, uyum gücü daha yüksek kuşaklar elde edilmeye başlanacaktır. Bu iyileşme önce hızlı olmakta, sonra da yavaşlayıp durmaktadır. Daha fazla iyileşme olmayacağı anlaşıldıktan sonra, algoritmanın durdurulması gerekir. Bu amaçla da önceden belirlenmiş bir durma ölçütü kullanılır.

Bir kesme planının doğal gösterimi, her bir parçanın ana malzeme üzerine yerleştirildiği koordinatlara dayanmaktadır. Bu gösterimin avantajı kesme planının tekrar düzenlenmesinin kolay olmasıdır. Ancak koordinatlarda küçük değişiklikler yapılması durumunda, yeniden düzenlenen kesme planında parçaların üst üste gelmesi durumu ortaya çıkabilmektedir. Doğal gösterimin bu özelliği ise genetik algoritmalar için uygun değildir. Bu nedenle daha kullanışlı bir veri yapısına ihtiyaç duyulmaktadır [21].

Kaynaklarda yer alan çalışmalarda genellikle kesme planları küçük parçaların ana malzeme üzerine yerleştirileceği sıraya karşı gelen permütasyonlarla gösterilmektedir. Ancak bu durumda her bir sıralamanın ana malzeme üzerindeki yerleşiminin belirlenmesi için ikinci bir yordama ihtiyaç duyulur. Bu gösterimin avantajı, genetik algoritmanın çaprazlama ve mutasyon operatörleri tarafından sıralamanın değiştirilmesi ile yeni permütasyonların çok kolay bir şekilde elde edilebilmesidir [21].

Genetik yaklaşımda kromozom yapısının belirlenmesinden sonra verilecek karar, başlangıç popülasyonunun büyüklüğünün ne olacağı ve bunun ne şekilde oluşturulacağıdır. Türetme, genellikle olurlu çözümlerin rastgele üretilmesi şeklinde olmaktadır. Başlangıç popülasyonunun büyüklüğü problemin son aşamaya yakınsamasına kuvvetli bir biçimde etki etmektedir. Küçük bir popülasyon, problemin arama uzayını azaltırken bunun tersine büyük bir popülasyon ise hesaplama maliyetini arttırmaktadır. Bu nedenle, yöntemin başarısı popülasyon büyüklüğünün uygunluğuna bağlı olmaktadır [31].

Uyum fonksiyonu, popülasyonda bulunan bireylerin kalitesinin belirlenmesi amacıyla kullanılmaktadır. En yüksek uyum değerine sahip bireylerin daha yüksek olasılıkla üreme işlemine alınması ve yeni neslin oluşturulmasında bu bireylerin kullanılması genetik algoritmaların en iyi çözüme yakınsamasında büyük rol oynamaktadır. Bir malzeme kesme probleminde kullanılabilecek uyum fonksiyonu, "etkin bir biçimde kullanılan alanın toplam alana bölünmesi" olarak tanımlanabilir (malzemeden yararlanma oranı).

Bir kromozomun seçilme olasılığının (selection probability) uyum değeri ile orantılı olarak belirlenmesi durumunda bazı istenmeyen durumlarla karşılaşılabilmektedir. Örneğin GA' nın ilk nesillerinde üstün özellikteki kromozomlar seçim işleminde çok fazla baskın olabilmekte (selection pressure) bu ise erken yakınsamaya (premature convergence) sebep olmaktadır. Bunun yanı sıra GA' nın son safhalarında ise kromozomların uyum değerlerinin birbirine yakın olması nedeniyle üstün olanların diğerleri ile rekabet etmesi daha güç olmaktadır. Seçim olasılığının belirlenmesinde karşılaşılan bu sorunların önlenmesi amacıyla geliştirilen ölçeklendirme (scaling) ve sıralama (ranking) tekniklerinden bazıları doğrusal ölçeklendirme, dinamik doğrusal ölçeklendirme ve sigma budama olarak adlandırılmaktadır [31].

Rassal örnekleme yöntemlerinden en iyi bilineni oransal seçim (proportional selection) veya rulet seçimi (roulette wheel selection) olarak adlandırılmaktadır. Bu yöntemde çark üzerindeki merkez açılarının büyüklükleri bireylerin uygunluk değerlerini temsil etmektedir. Bir sonraki nesilde yer alacak bireyin seçimi için çark rastgele döndürülür. Çark üzerinde büyük temsil edilen yüksek uyum değerli bireyler seçilmek için daha çok şansa sahiptir. Deterministik yöntemlerden bazıları budama

(truncation) seçimi, elitist seçimi ve kuşak değişimi (generational replacement) olarak tanımlanmaktadır. Kuşak değişiminde yeni nesil sadece çocuk kromozomlardan oluşmaktadır. Karma örnekleme yönteminin çok iyi bilinen bir örneği ise rassal turnuva seçimi işlemidir. Bu yöntemde rassal olarak seçilen belirli sayıdaki bireyler arasından en yüksek uyum değerine sahip olanlar yeni nesle aktarılmaktadır [18].

Genetik algoritmalar en iyi çözümün elde edilebileceğini garanti etmemektedir. Bu nedenle genetik algoritmaların tasarım aşamasında, arama işlemini sona erdirmek için bir "durma ölçütü" nün belirlenmesi gerekmektedir. Durma ölçütü belirli bir çözüm değeri olabileceği gibi algoritmanın belirli sayıda nesil sonunda sona erdirilmesi olarak da tanımlanabilmektedir.

3.4.5.1 Genetik Algoritma Operatörleri

Kullanılan genetik operatörler, varolan popülasyon üzerine uygulanan işlemlerdir. Bu işlemlerin amacı daha iyi özelliğe sahip yeni nesiller üretmek ve arama algoritmasının alanını genişletmektir. Farklı uygulamalarda farklı operatörler kullanılmakla birlikte genelde genetik algoritmada üç standart operatör kullanılır. Bu operatörler: [31]

- Yeniden Üretim (Re-production)
- Çaprazlama (Crossover)
- Mutasyon (Mutation)

Yeniden Üretim

Bu aşama küme elemanlarının ya da verilerin kromozoma dönüştürülüp seçildiği bölümdür. Bir kromozom temsil ettiği çözüm hakkında bir şekilde bilgi içermelidir. En çok kullanılan kodlama ikili karakter dizisidir. Bu yöntemle kromozom şu şekilde görülmektedir:

Kromozom 1 : 1101100100110110
Kromozom 2 : 1101111000011110

Her kromozom ikili karakter dizisi şeklinde temsil edilmektedir. Karakter dizisinde her bir çözümün bir özelliğini temsil eder. Bir başka olasılık tüm karakter dizisinin bir sayıyı temsil etmesidir. Elbette, birçok başka kodlama yöntemi vardır. Kodlama daha çok çözülen probleme bağlıdır. Örneğin bazı problemler için tamsayı veya gerçek sayı şeklinde kodlamak gerekirken, bazı problemlerde permütasyon şeklinde kodlamaya ihtiyaç vardır [28].

Çaprazlama

Üretim (ya da yeniden üretim) sonra, çaprazlama işlemiyle devam edilir. Çaprazlama, atalardaki seçili genler üzerinde işlem yapar ve yeni çocuklar oluşturur. Bunun en basit şekli, rasgele bir kesme noktası (çaprazlama noktası) seçip, bu noktadan önceki her şeyi ilk bireyden, sonraki her şeyi ikinci bireyden alıp birleştirerek çocuğu oluşturmaktır. Çaprazlama aşağıdaki şekilde gösterilebilir: (| kesme noktasıdır): [10].

Kromozom 1 : 11011 | 00100110110
Kromozom 2 : 11011 | 11000011110
Çocuk 1 : 11011 | 11000011110
Çocuk 2 : 11011 | 00100110110

Çaprazlamanın birçok yolu mevcuttur, örneğin birden fazla kesme noktası seçilebilir. Çaprazlama daha da karmaşık olabilir ve tamamen kromozomların kodlanmasına bağlıdır. Özel problemler için yapılmış özel çaprazlamalar genetik algoritmanın başarısını arttırabilir.

Mutasyon

Çaprazlama işlemi gerçekleştirildikten sonra, mutasyon işlemi yapılır. Mutasyonun amacı, toplumdaki tüm çözümlerin çözülen problemlerin yerel uygun değerine dönüşmesinin önüne geçmektir. Mutasyon işlemi çaprazlama sonucu oluşan çocuğu rastgele değiştirmektedir. İkili kodlamada rastgele seçilmis bir kaç biti 1’ i 0’ a, 0’ ı 1’ e şeklinde değiştirmek bir mutasyondur [28].

Asıl Çocuk 1 : 1101111000011110
Mutasyon Geçirmis Çocuk 1 : 1100111000011110
Asıl Çocuk 2 : 1101100100110110
Mutasyon Geçirmis Çocuk 2 : 1101101100110110

Mutasyon tekniği (çaprazlama tekniği de) kromozomların kodlamasına çoğunlukla bağlıdır. Örneğin permütasyon şeklinde kodlamadaki mutasyon, rastgele seçilen iki genin yer değiştirmesi olarak gerçekleştirilir. Tamsayı gösterimleri için kullanılan mutasyon işlemcileri

- Yer değiştirme mutasyonu (Swap mutation)
- Üçlü yer değiştirme mutasyonu (Three swap mutation)
- Kaydırma mutasyonu (Shift mutation)
- Ters mutasyon (Inversion mutation)
- Bitişik mutasyon (Adjacent mutation)

olarak tanımlanır [20].

Tablo 3.4' de stok kesme problemlerinde kullanılabilecek sezgisel ve metasezgisel algoritmaların özellikleri bir arada değerlendirilmiştir.

Tablo 3.4: Metasezgisel ve sezgisel metodların karşılaştırmaları [23]

Yöntem	Avantaj	Dezavantaj
Lineer Programlama Prosedürü	Hızlı, sadece optimal çözüme ulaştıracak yerleşme planları oluşturulur.	Kesirli sonuçlar üretebilir ve tek amacı kesme kaybı minimizasyonudur.
Ardışık Sezgisel Prosedür	Hızlı, tamsayılı çözümler üretir. Birden çok amacın göz önünde bulundurulmasını sağlar.	Özellikle zor problemlerde artan kesme kayıplarına yol açan çözümler üretir.
Tabu Araştırması	Açgözlü rassallaştırılmış uyarlamalı arama yordamı ile karşılaştırıldığında daha kaliteli sonuçlar üretir.	Açgözlü rassallaştırılmış uyarlamalı arama yordamından daha yavaştır.
Açgözlü Rassallaştırılmış Uyarlamalı Arama Yordamı	Tabu Araştırmasına göre çok hızlı olup kabul edilebilir sonuçlar üretir.	Tabu araştırmasına göre daha kesin olmayan sonuçlar üretir.
Genetik Algoritma	Göreceli olarak basittir.	Karınca kolonisi algoritmasına göre daha yavaştır.
Karınca Kolonisi Algoritması	Problemlere etkin ve kesin çözümler sunar.	Göreceli olarak komplikedir ve kısıtların artmasıyla birlikte verimsiz bir metot haline gelebilir.
Benzetilmiş Tavlama Algoritması	Genetik algoritmaya göre daha hızlıdır.	Genetik algoritmaya göre daha verimsiz sonuçlar elde eder.

4. UYGULAMA

Bu bölümde önceki bölümlerde anlatılan stok kesme problemlerine ait çözüm yaklaşımı iki alt bölümde incelenmiştir.

Birinci bölümde iki boyutlu düzgün dikdörtgen şekillerden oluşan stok kesme problemi sezgisel yöntemle çözülecektir. Bu kapsamda Bölüm 3.1.1' de anlatılan "BLF Algoritması" ile Bölüm 3.2.2' de anlatılan "İlk Uygun Azalan Sezgisel Yöntem" birlikte kullanılarak bir program geliştirilmiştir. Geliştirilen programın mevcut test verileri ile karşılaştırılıp performansı değerlendirilmiştir.

İkinci bölümde ise; iki boyutlu düzgün şekillerden oluşan stok kesme problemi metasezgisel yöntemle çözülecektir. Bu kapsamda Bölüm 3.1.1' de anlatılan "BLF Algoritması" ile Bölüm 3.4.3' de anlatılan "Genetik Algoritma" birlikte kullanılarak bir program geliştirilmiştir. Geliştirilen program mevcut test verileri ve ticari bir stok kesme programı olan Diamino ile karşılaştırılıp performansı değerlendirilmiştir.

4.1 Sezgisel Yöntem ile Stok Kesme Problemlerine Geliştirilen Çözüm Yaklaşımı

Sezgisel yöntem ile stok kesme problemlerine geliştirilen çözüm yaklaşımı üç aşamada incelenmiştir. Birinci aşamada verilerin seçim algoritmasından bahsedilmiştir. İkinci aşamada yerleşim algoritmasından bahsedilmiştir. Üçüncü aşamada ise seçme ve yerleştirme algoritmaları birleştirilerek geliştirilen yazılım mevcut test verileri ile karşılaştırılıp performansı değerlendirilmiştir.

4.1.1 Seçim Algoritması

Seçim Algoritması olarak Bölüm 3.2.2' de anlatılan "İlk Uygun Azalan Sezgisel Yöntem" kullanılmıştır. Bu yöntemde stok kesim malzemeleri alanlarına göre büyükten küçüğe doğru sıralanır.

4.1.2 Yerleştirme Algoritması

Yerleştirme Algoritması olarak Bölüm 3.1.1' de anlatılan BLF Algoritması kullanılacaktır. Bu yöntemde stok kesim malzemeleri permütasyon sırasına göre aşağı sol kısımdan başlayarak, öncelikli yerleştirme arasında kalan boşluklar doldurularak yerleştirme yapılır. Geliştirilen sezgisel yazılımda başlangıç noktası olarak sol üst köşe alınmıştır. Ayrıca parçaların döndürülemediği kısıtı eklenmiştir.

4.1.3 Sezgisel Yöntem ile Geliştirilen Yazılım ve Performans Değerlendirmesi

Sezgisel Yöntem ile geliştirilen yazılımda "BLF" ve "İlk Uygun Azalan Sezgisel Yöntem" kullanılacaktır. Daha sonra ise mevcut test verileri ile performansı değerlendirilecektir. Bu yazılımın geliştirilmesinde VB6 (Visual Basic 6.0) kullanılmıştır. Geliştirilen yazılımda kullanılan algoritma;

Adım 1: Yüksekliği L ve genişliği W olan büyük dikdörtgen ve genişliği w_i ve yüksekliği l_i ve âdeti k_i olan küçük dikdörtgenleri belirle.
Adım 2: Küçük dikdörtgenleri alanlarına göre büyükten küçüğe sırala.
Adım 3 : i = 1
Adım 4 : Alanı büyük ilk stok malzemesini ilk sol üst boş köşeye al ve i. dikdörtgenin oraya yerleşip yerleşmediğini kontrol et yerleşiyor ise alana yerleştir.
Adım 5 : Eğer i. kutunun yerleşeceği boş alan yok ise i = i + 1 .
Adım 6 : Eğer i ≤ kutu sayısı ise adım 4' e git değil ise adım 7' e git.
Adım 7 : Bitir.

Verimlilik = (Kullanılan alan / Ana parçanın alanı) x 100 **(4.1)**

olarak hesaplanmıştır. Geliştirilen yazılımın mevcut test verileri ile performansı örneklerle değerlendirilirse;

Örnek 1.

Büyük parçanın boyutları : 100 x 70

Tablo 4.1: Örnek 1' e ait parçaların boyutları [24]

Parça No	**En**	**Boy**	**Adet**
1	20	40	1
2	40	10	1
3	30	10	1
4	10	10	1
5	20	20	1
6	10	30	1
7	30	10	1
8	20	10	1
9	20	20	1
10	30	50	1
11	20	30	2
12	30	20	1
13	50	10	1

<table>
<tr><td rowspan="3">20 x 40</td><td colspan="2">40 x 10</td><td colspan="2">30 x 10</td><td>10 x 10</td></tr>
<tr><td rowspan="2">20 x 20</td><td rowspan="3">10 x30</td><td colspan="2">30 x 10</td><td>20 x 10</td></tr>
<tr><td rowspan="2">20 x 20</td><td rowspan="3">30 x 50</td></tr>
<tr><td rowspan="2">20 x 30</td><td rowspan="2">20 x 30</td><td colspan="2" rowspan="2">30 x 20</td></tr>
<tr><td colspan="2">50 x 10</td></tr>
</table>

Şekil 4.1: Örnek1' e ait optimum yerleşim [24]

Örnek 1 geliştirilen sezgisel yazılım ile çözülürse;

Şekil 4.2: Geliştirilen sezgisel yazılım ile Örnek 1' in çözümü

Şekil 4.2’ de görüldüğü gibi stok kesme problemi 4 saniyede çözülmüştür. 10 x 30 boyutundaki parça yerleştirilememiş ve verimlilik % 95.714’ tür.

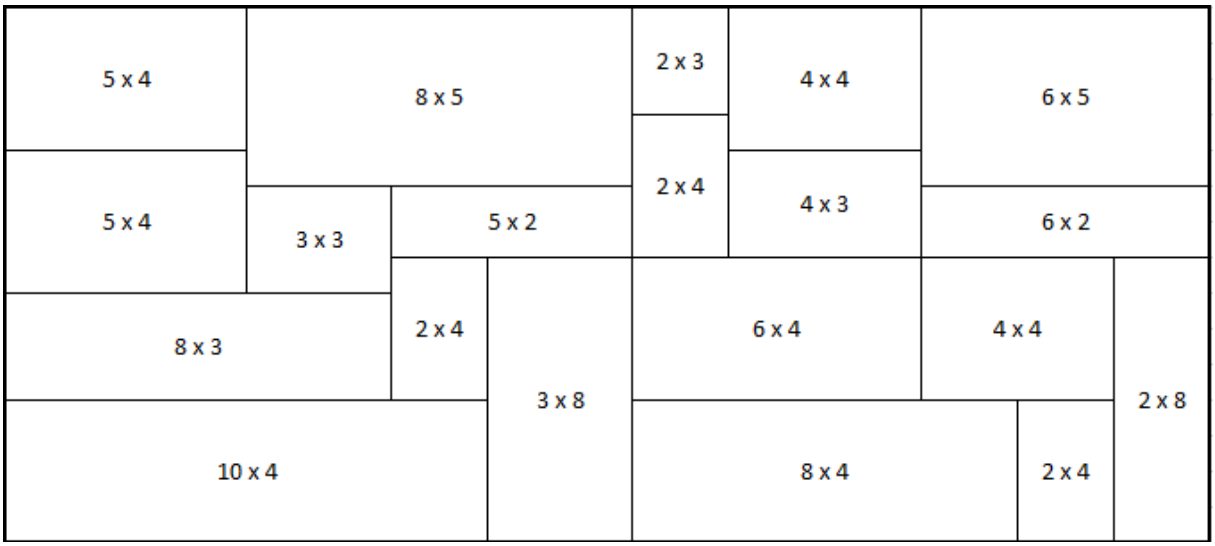
Örnek 2.

Büyük parçanın boyutları : 25 x 15

Tablo 4.2: Örnek 2’ ye ait parçaların boyutları [32]

Parça No	**En**	**Boy**	**Adet**
1	5	4	1
2	8	5	1
3	2	3	1
4	4	4	1
5	6	5	1
6	5	4	1
7	3	3	1
8	5	2	1
9	2	4	1
10	4	3	1
11	6	2	1
12	8	3	1
13	2	4	1
14	3	8	1
15	6	4	1

16	4	4	1
17	2	8	1
18	10	4	1
19	8	4	1
20	2	4	1



Şekil 4.3: Örnek2' ye ait optimum yerleşim [32]

Örnek 2 geliştirilen sezgisel yazılım ile çözülürse;

Şekil 4.4: Geliştirilen sezgisel yazılım ile Örnek 2' nin çözümü

Şekil 4.4' de görüldüğü gibi stok kesme problemi 0 ile 1 saniye arasında çözülmüştür. 2 x 4 boyutundaki 2 adet parça yerleştirilememiş ve verimlilik % 93.6' dır.

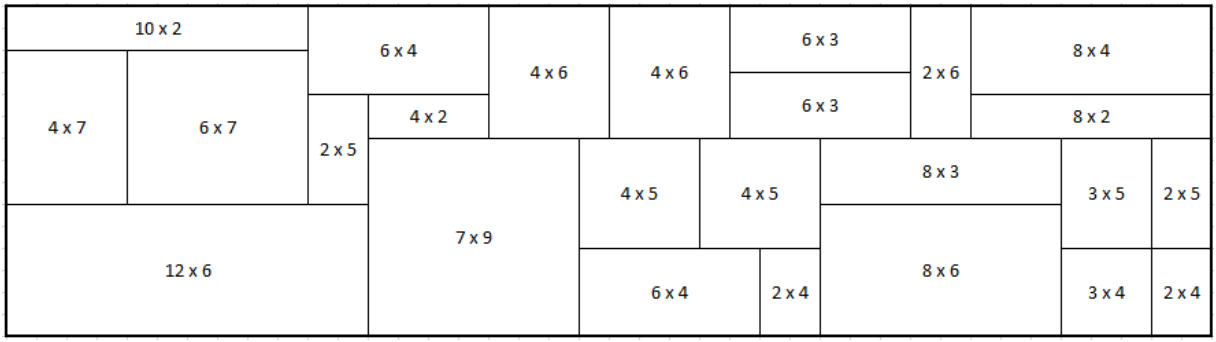
Örnek 3.

Büyük parçanın boyutları : 40 x 15

Tablo 4.3: Örnek 3' e ait parçaların boyutları [32]

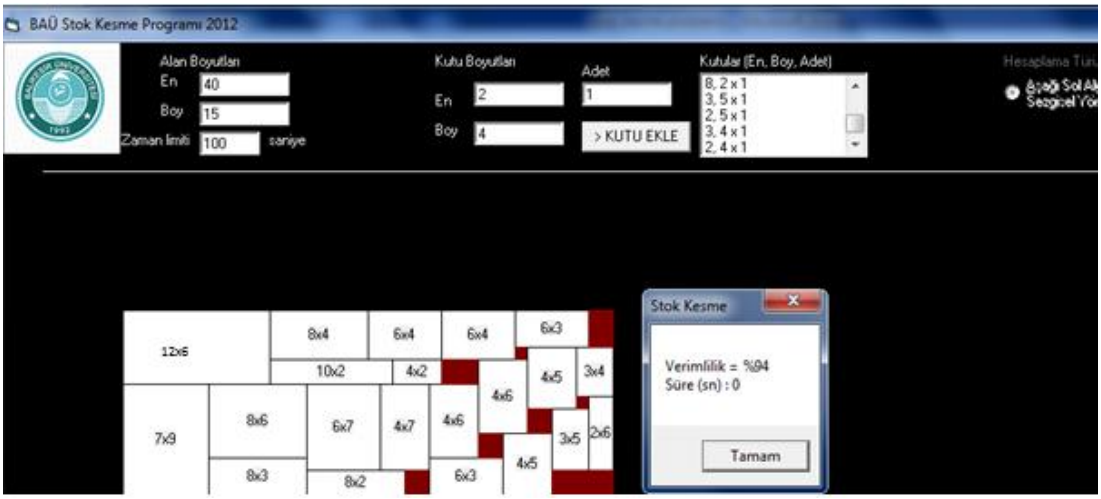
Parça No	**En**	**Boy**	**Adet**
1	12	6	1
2	4	7	1
3	6	7	1
4	10	2	1

5	2	5	1
6	6	4	1
7	4	2	1
8	4	6	1
9	7	9	1
10	4	5	1
11	6	4	1
12	4	6	1
13	6	3	1
14	4	5	1
15	2	4	1
16	8	4	1
17	8	6	1
18	8	3	1
19	6	3	1
20	2	6	1
21	8	2	1
22	3	5	1
23	2	5	1
24	3	4	1
25	2	4	1



Şekil 4.5: Örnek3' e ait optimum yerleşim [32]

Örnek 3 geliştirilen sezgisel yazılım ile çözülürse;



Şekil 4.6: Geliştirilen sezgisel yazılım ile Örnek 3'ün çözümü

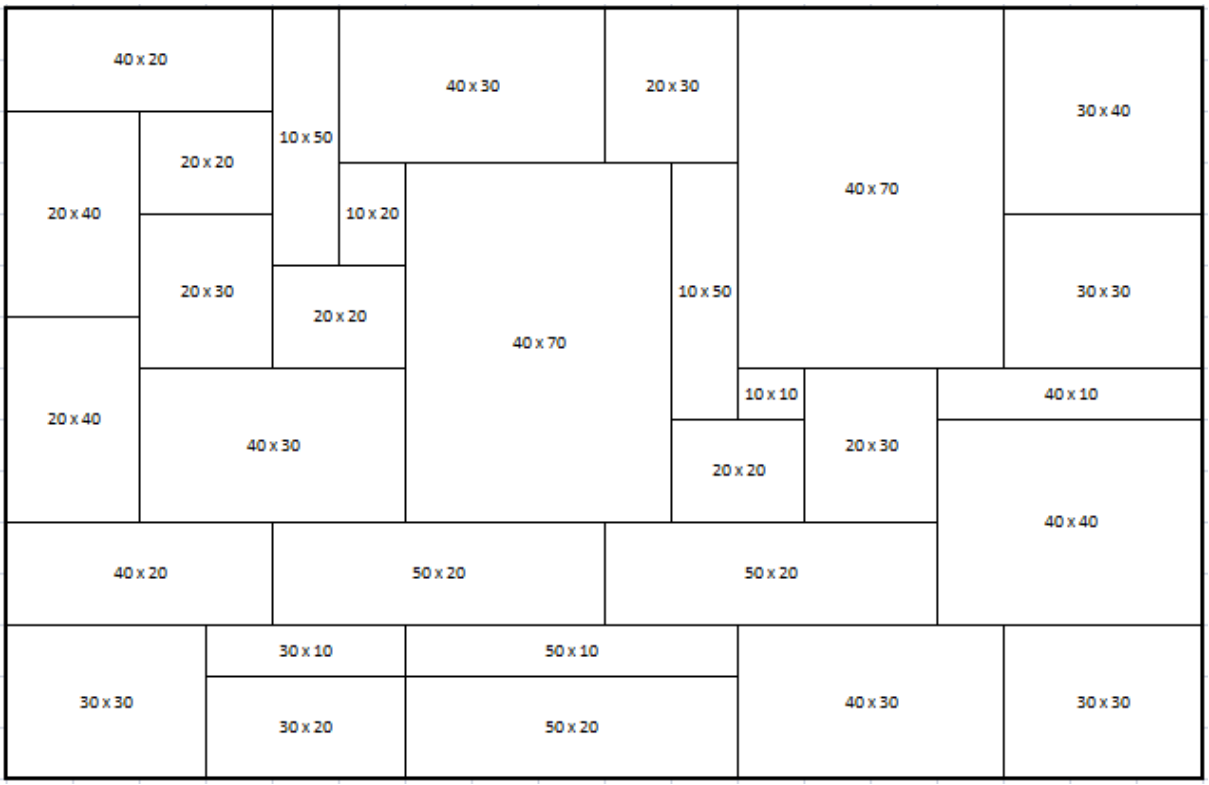
Şekil 4.6' da görüldüğü gibi stok kesme problemi 0 ile 1 saniye arasında çözülmüştür. 2 x 5' den 2 adet ve 2 x 4' den 2 adet parça yerleştirilememiş ve verimlilik % 94' tür.

Örnek 4.

Büyük parçanın boyutları : 180 x 150

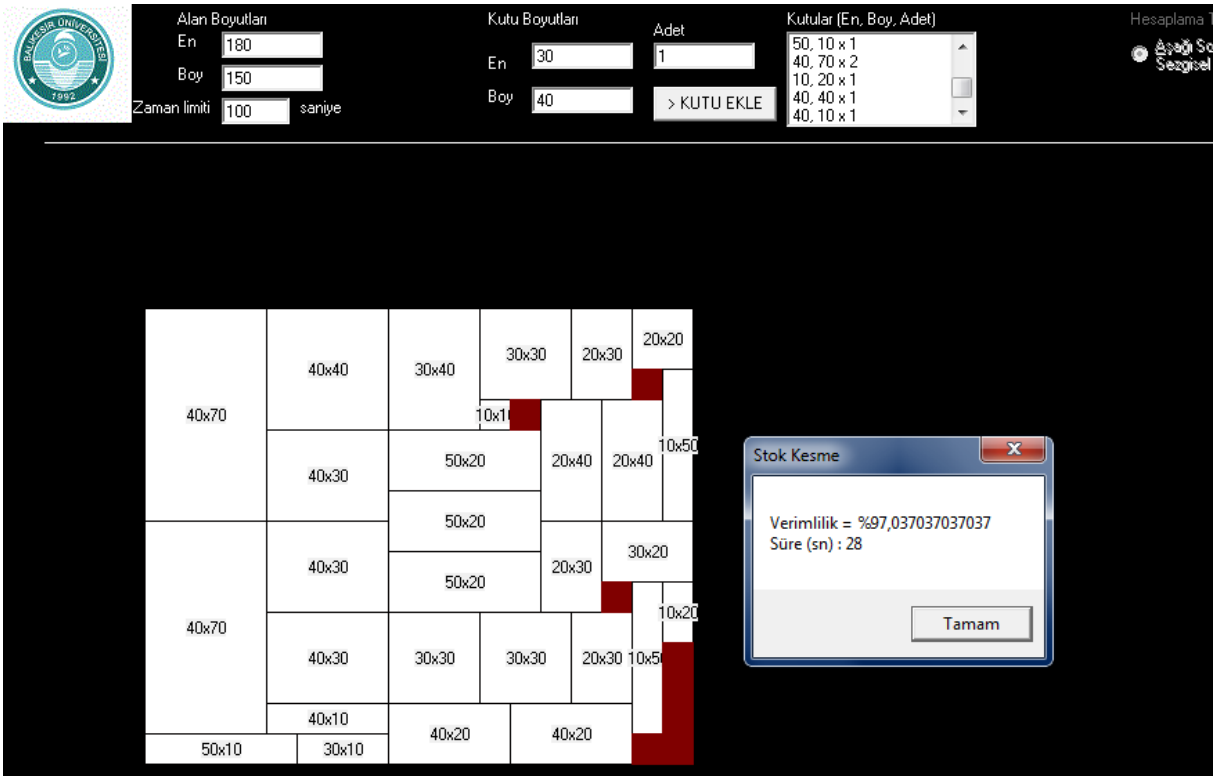
Tablo 4.4 Örnek 4’ e ait parçaların boyutları [32]

Parça No	**En**	**Boy**	**Adet**
1	40	20	2
2	20	40	2
3	10	50	2
4	20	20	3
5	20	30	3
6	30	30	3
7	40	30	3
8	30	20	1
9	10	10	1
10	30	10	1
11	50	20	3
12	50	10	1
13	40	70	2
14	10	20	1
15	40	40	1
16	40	10	1
17	30	40	1



Şekil 4.7: Örnek4' e ait optimum yerleşim [32]

Örnek 4 geliştirilen sezgisel yazılım ile çözülürse;



Şekil 4.8: Geliştirilen sezgisel yazılım ile Örnek 4' ün çözümü

Şekil 4.8’ de görüldüğü gibi stok kesme problemi 28 saniyede çözülmüştür. 20 x 20’ den 2 adet yerleştirilememiş ve verimlilik % 97,03’ dür.

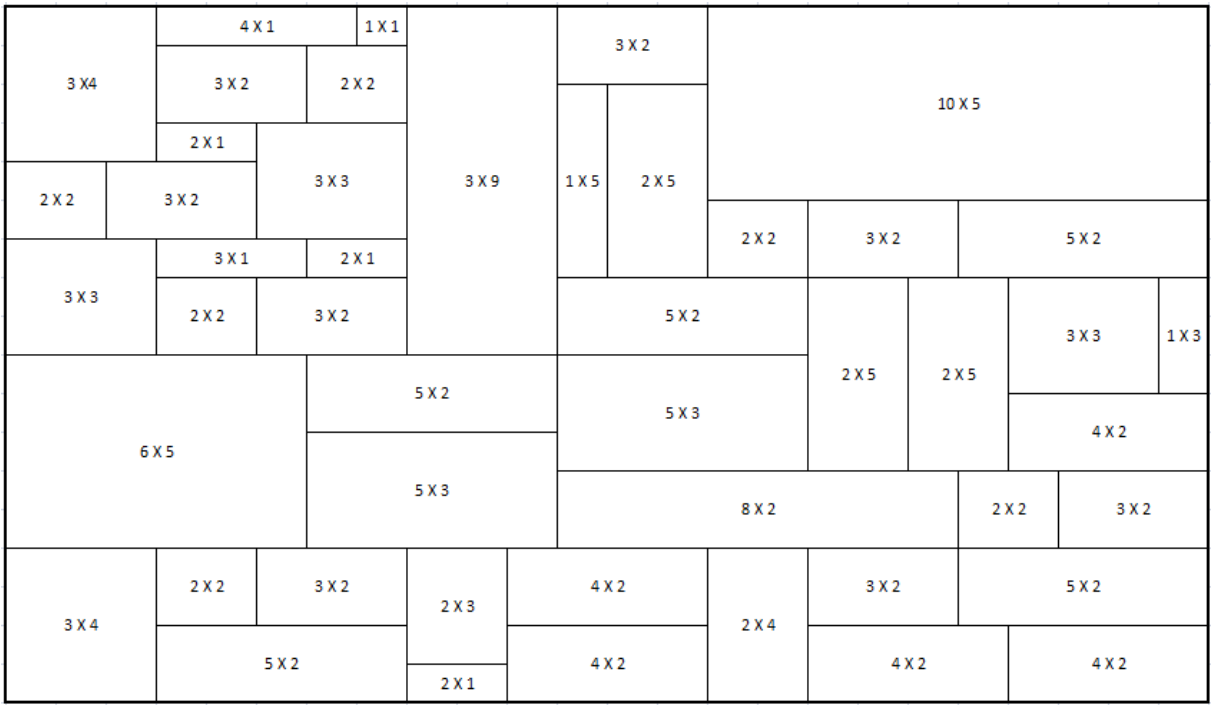
Örnek 5.

Büyük parçanın boyutları : 24 x 18 [Bu örnek yazar tarafından test amacıyla hazırlanmıştır.]

Tablo 4.5: Örnek 5’ e ait parçaların boyutları

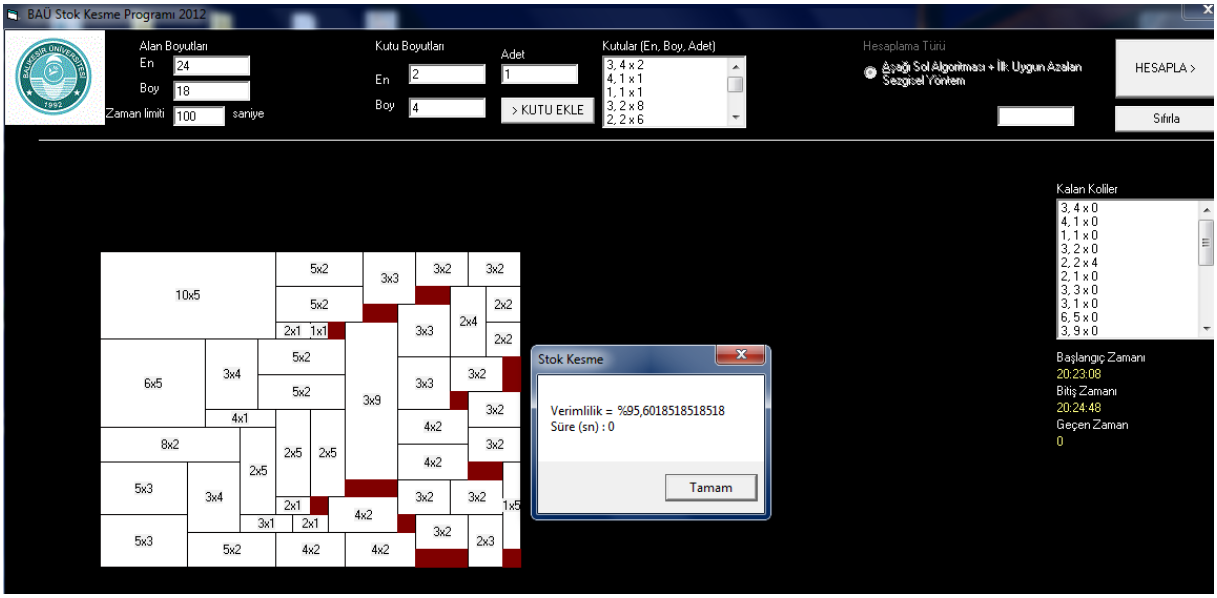
Parça No	**En**	**Boy**	**Adet**
1	3	4	2
2	4	1	1
3	1	1	1
4	3	2	8
5	2	2	6
6	2	1	3
7	3	3	3
8	3	1	1
9	6	5	1
10	3	9	1
11	1	5	1
12	2	5	1
13	5	2	5
14	10	5	1
15	5	3	2

16	2	5	2
17	8	2	1
18	1	3	1
19	4	2	5
20	2	3	1
21	2	4	1



Şekil 4.9: Örnek5' e ait optimum yerleşim

Örnek 5 geliştirilen sezgisel yazılım ile çözülürse;



Şekil 4.10: Geliştirilen sezgisel yazılım ile Örnek 5' in çözümü

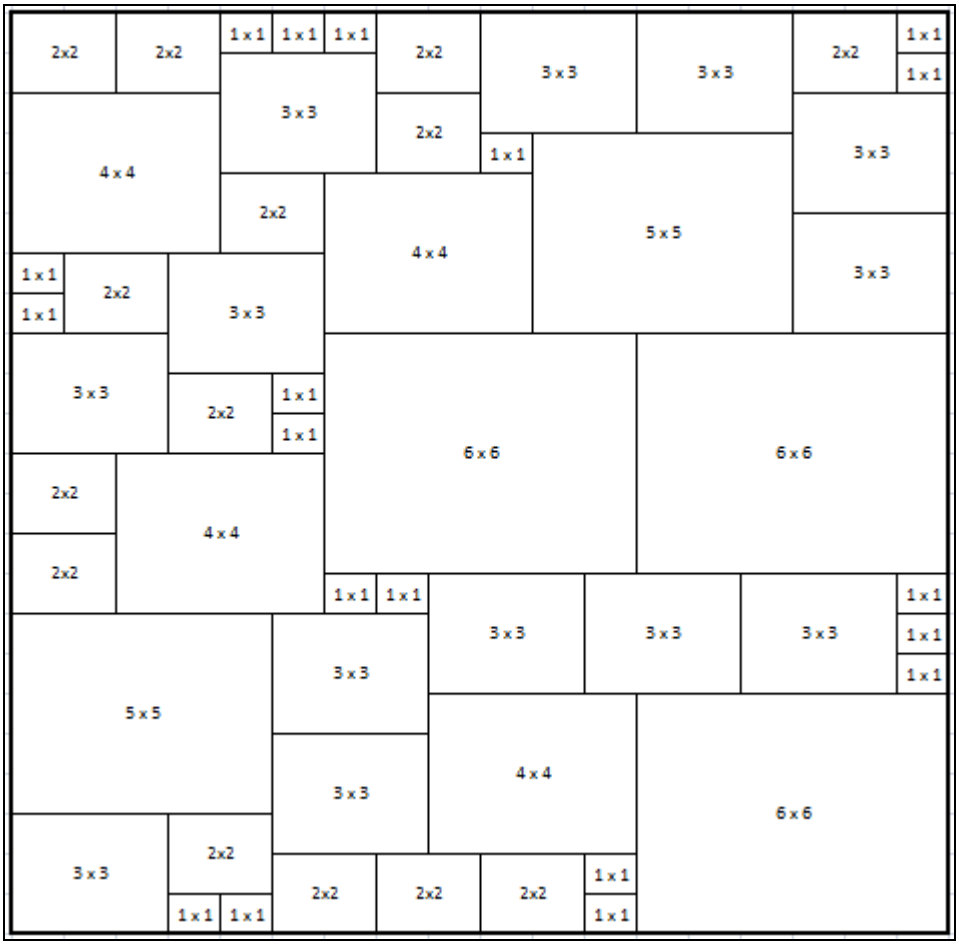
Şekil 4.10' da görüldüğü gibi stok kesme problemi 0 ile 1 saniye arasında çözülmüştür. 2 x 2' den 4 adet ve 1 x 3' den 1 adet yerleştirilememiştir.Sonuç olarak verimlilik % 95,6' dır.

Örnek 6.

Büyük parçanın boyutları : 18 x 23 [Bu örnek yazar tarafından test amacıyla hazırlanmıştır.]

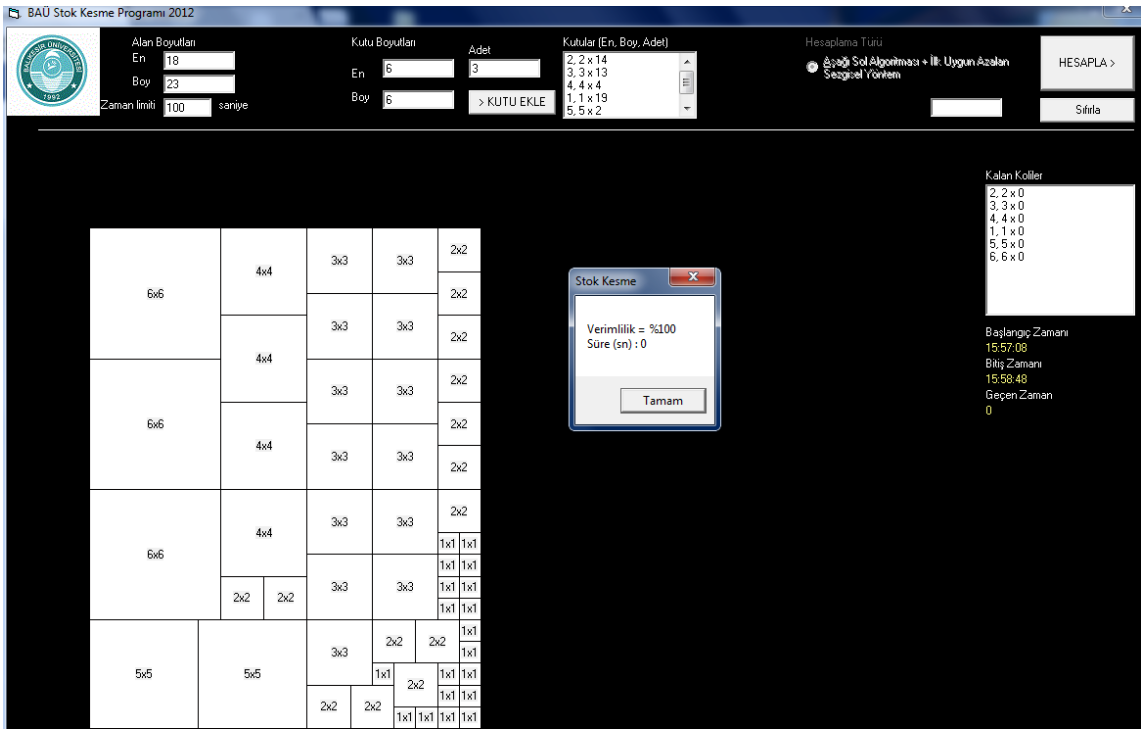
Tablo 4.6: Örnek 6' ya ait parçaların boyutları

Parça No	En	Boy	Adet
1	2	2	14
2	3	3	13
3	4	4	4
4	1	1	19
5	5	5	2
6	6	6	3



Şekil 4.11: Örnek 6' ya ait optimum yerleşim

Örnek 6 geliştirilen sezgisel yazılım ile çözülürse;



Şekil 4.12: Geliştirilen sezgisel yazılım ile Örnek 6' nın çözümü

Şekil 4.12' de görüldüğü gibi stok kesme problemi 0 ile 1 saniye arasında çözülmüştür. Sezgisel yazılım ile kare şekillerden oluşan stok kesme probleminde optimum çözüme ulaşılmıştır.

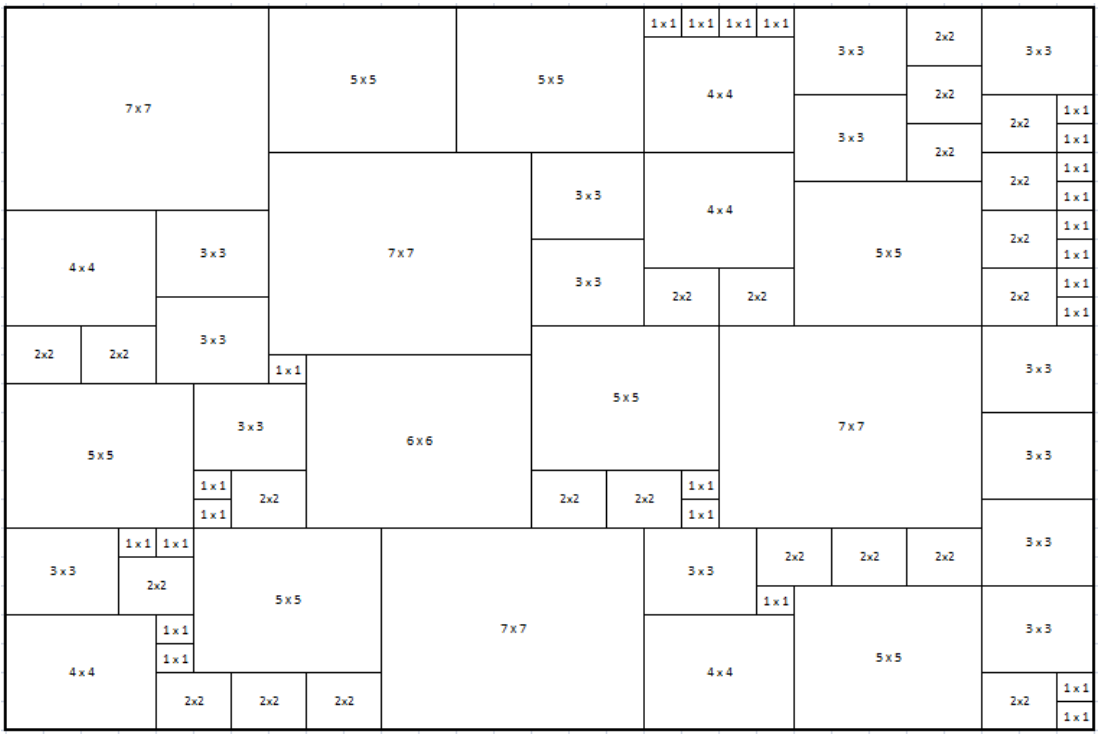
Örnek 7.

Büyük parçanın boyutları : 60 x 90 [Bu örnek yazar tarafından test amacıyla hazırlanmıştır]

Tablo 4.7: Örnek 7' ye ait parçaların boyutları

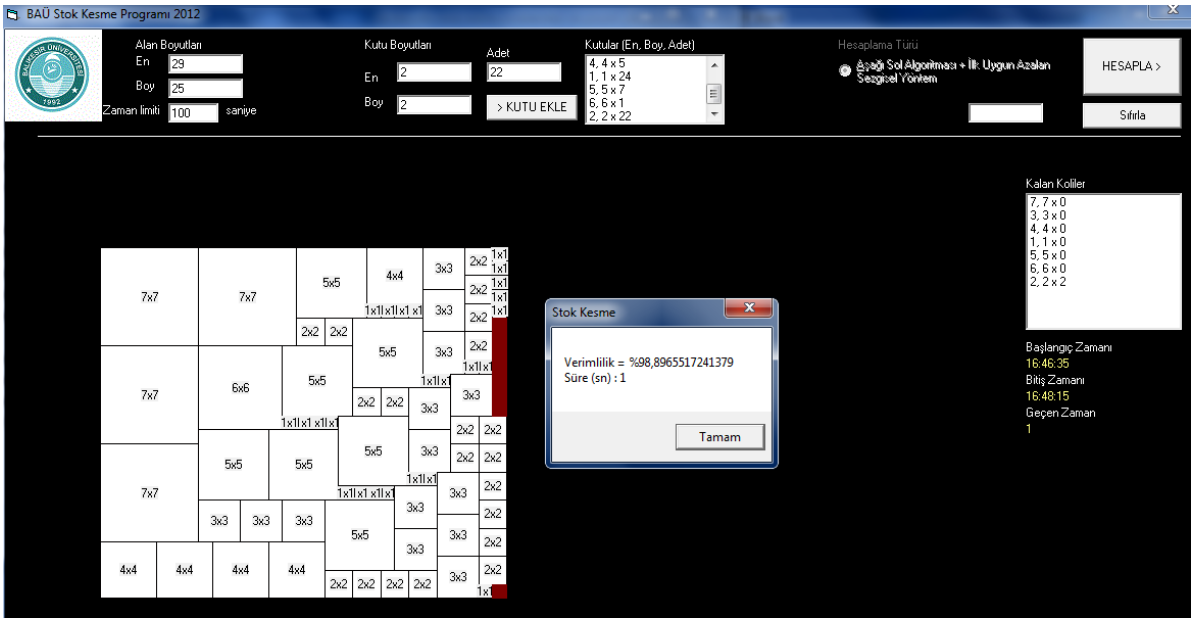
Parça No	En	Boy	Adet
1	7	7	4
2	3	3	14
3	4	4	5
4	1	1	24

5	5	5	7
6	6	6	1
7	2	2	22



Şekil 4.13: Örnek 7' ye ait mevcut optimum yerleşim

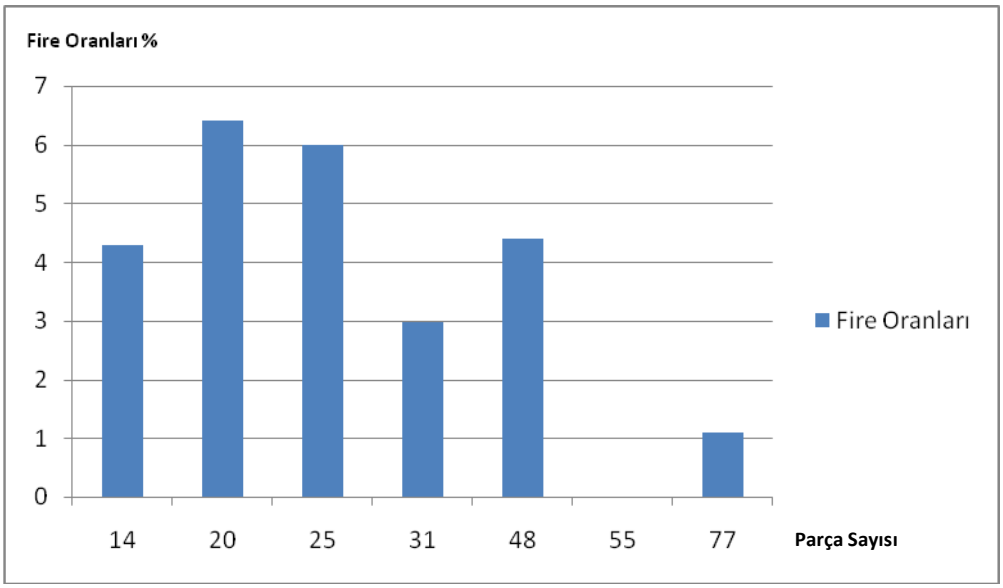
Örnek 7 geliştirilen sezgisel yazılım ile çözülürse;



Şekil 4.14: Geliştirilen yazılım ile Örnek 7' nin çözümü

Şekil 4.14’ te görüldüğü gibi stok kesme problemi 1 saniyede çözülmüştür. Sezgisel yazılım ile kare şekillerden oluşan stok kesme probleminde optimum çözüme ulaşılamamış, 20 x 20’ den 2 adet parça yerleştirilememiş ve verimlilik %98,896’ dır.

Yukarıda optimumu verilen mevcut örnekler incelendiğinde, bu örneklerin optimum çözümüne geliştirilen sezgisel yazılım ile Örnek6’ da ulaşılmıştır. Örneklerin çözümü 0 ile 27 saniye arasında olmuştur. Bu da oldukça hızlı çözümdür. Kare şekillerden oluşan örneklerde fire oranı daha düşüktür. Fire oranı % 0 ile % 7 arasında değişmektedir. Bu da oldukça iyi bir orandır. Çoğu sektörde bu fire oranı kabul edilebilir bir değerdir. Şekil 4.15’ te geliştirilen sezgisel yazılım ile parça sayısı ve fire oranları karşılaştırılmıştır. Şekilde görüldüğü gibi kare şekillerde fire oranı daha düşüktür. Parçalar arasında benzerlik azaldıkça fire oranı artmaktadır.



Şekil 4.15: Geliştirilen sezgisel yazılım ile parça sayısı ve fire oranı karşılaştırılması

4.2 Metasezgisel Yöntem ile Stok Kesme Problemlerine Geliştirilen Çözüm Yaklaşımı

Metasezgisel yöntem ile stok kesme problemlerine geliştirilen çözüm yaklaşımı beş aşamada incelenecektir. Birinci aşamada tekstil sektöründe popüler metasezgisel ticari bir program olan Lectra grubuna ait Diamino programından

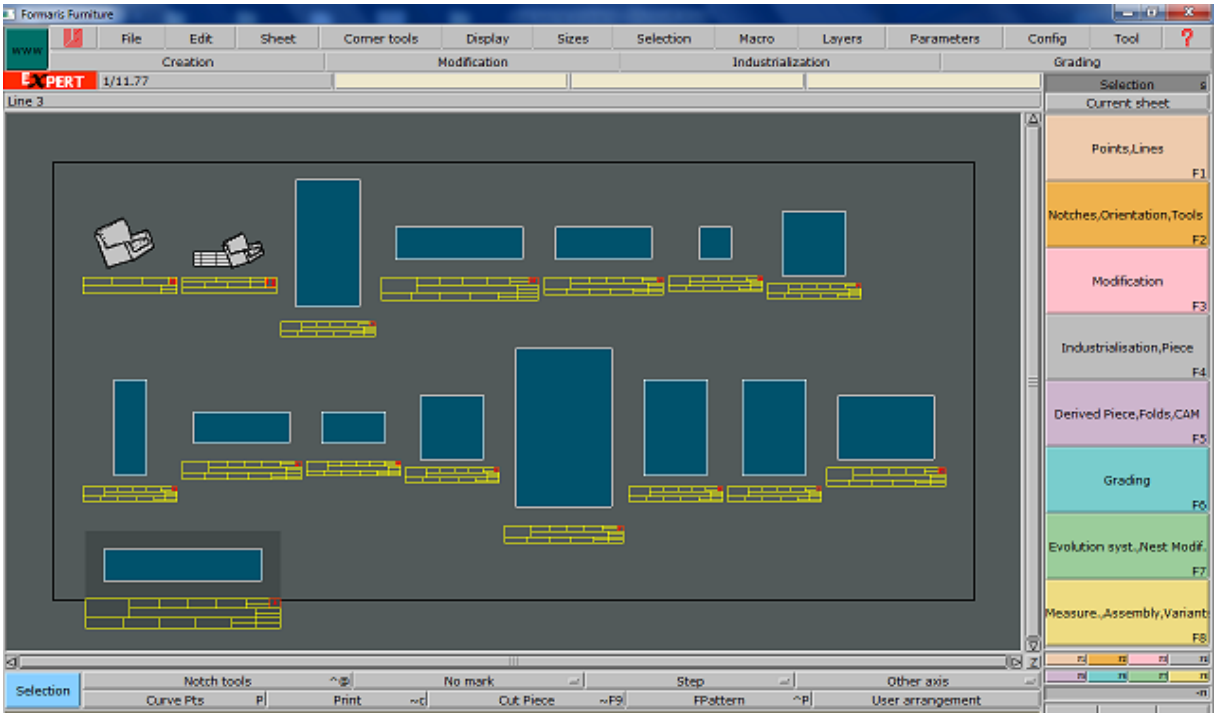
bahsedilecek ve optimumu verilen mevcut yedi örnek Diamino programı ile çözülecektir. İkinci aşamada geliştirilecek olan metasezgisel yazılımda kullanılan seçim algoritmasından bahsedilecektir. Üçüncü aşamada geliştirilecek olan metasezgisel yazılımda kullanılacak yerleşim algoritmasından bahsedilecektir. Dördüncü aşamada ise seçme ve yerleştirme algoritmaları birleştirilerek geliştirilen metasezgisel yazılım mevcut test verileri ile karşılaştırılıp performansı değerlendirilecektir. Beşinci aşamada ise geliştirilen metasezgisel yazılım ile Diamino programı karşılaştırılıp performans değerlendirmesi yapılacaktır.

4.2.1 Lectra Firmasına Ait Diamino Metasezgisel Programı

Lectra firması Fransa merkezli ve 1973 yılında kurulmuştur. Yaklaşık 1350 çalışanı vardır. Lectranın entegre teknoloji çözümleri-yazılım, CAD / CAM donanım dünyanın önde gelen firmalarındandır ve tekstil, deri, endüstriyel kumaşlar, ya da kompozit kullanan firmalara hizmet vermektedir. Moda, otomotiv ve mobilya yanı sıra havacılık, deniz, rüzgar türbünü gibi alanlarda Lectra' nın entegre ürünleri kullanılmaktadır [33].

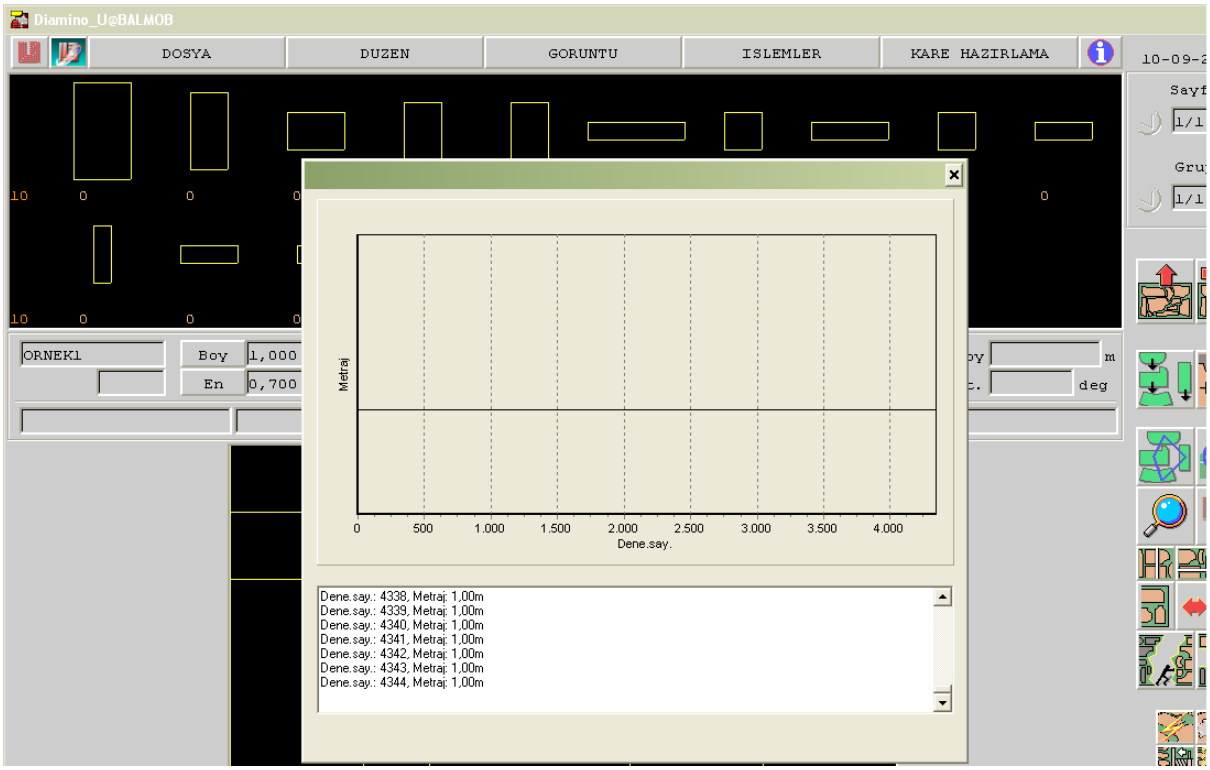
Kanepe ve koltuk imalatında Lectra' nın entegre sistemleri yaygın olarak kullanılmaktadır. Kumaş kesiminde Lectra CNC makinesi rakiplerine göre oldukça hızlıdır. En önemli avantajları, konveyör hareket ederken kesime devam etmesi ve kendi CAD ve CAM programlarını kullanmasıdır. Lectra çizimde Formaris programını optimizasyon ve pastal yerleştirmede Diamino programını kullanmaktadır [33].

Optimizasyonu verilen mevcut yedi örnek Diamino programı ile çözülmüştür.

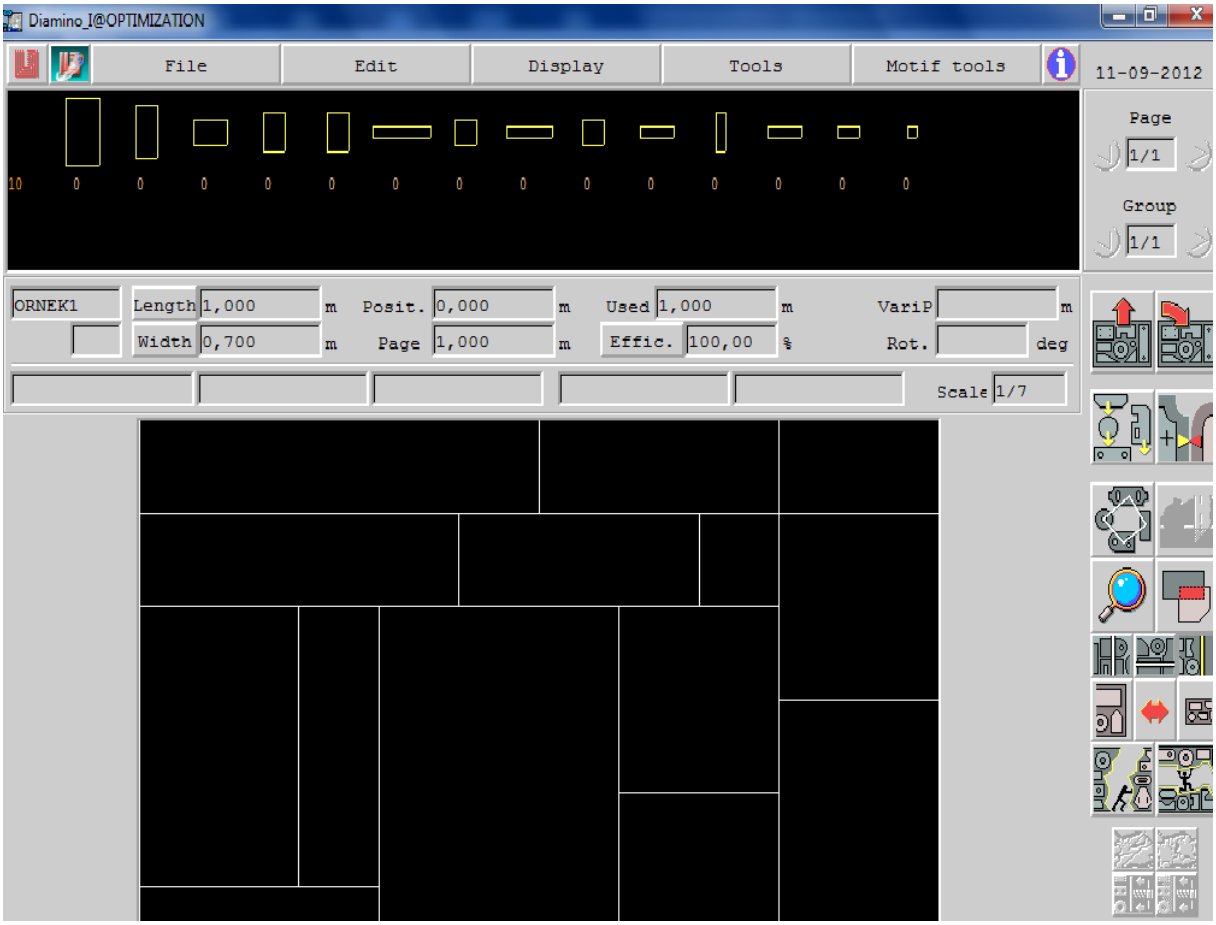


Şekil 4.16: Formaris programı ile Örnek 1' in stok malzemeleri

Örnek 1 Diamino programı ile çözülürse;;



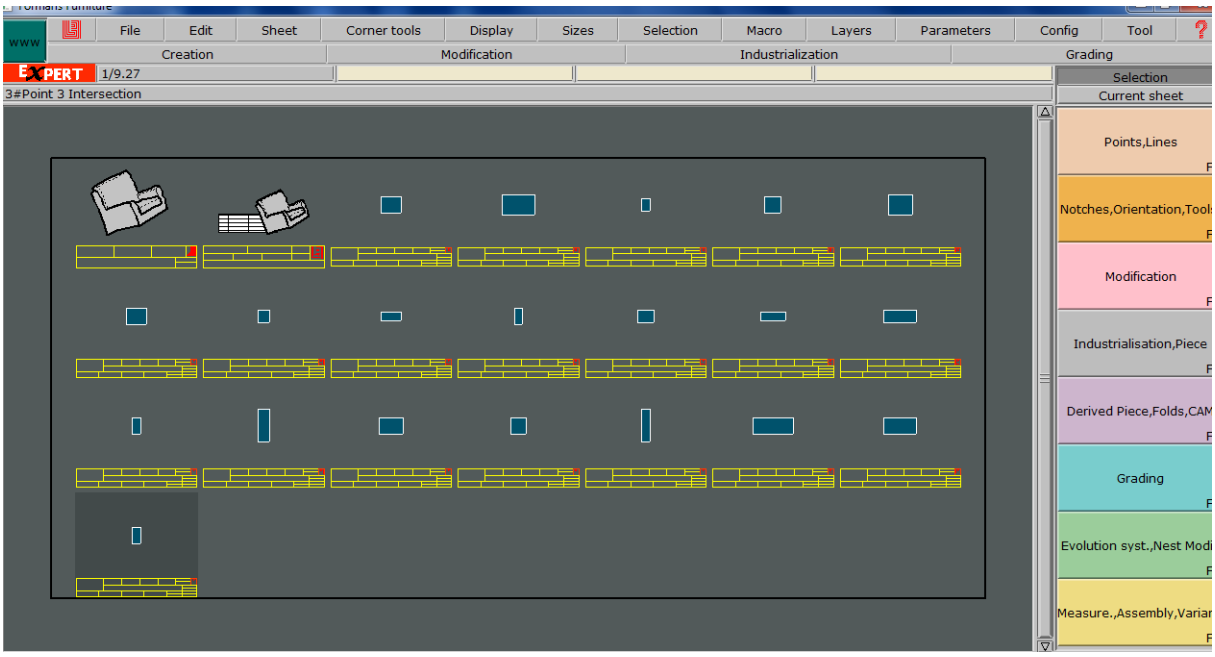
Şekil 4.17: Diamino programı ile Örnek 1' in iterasyonu



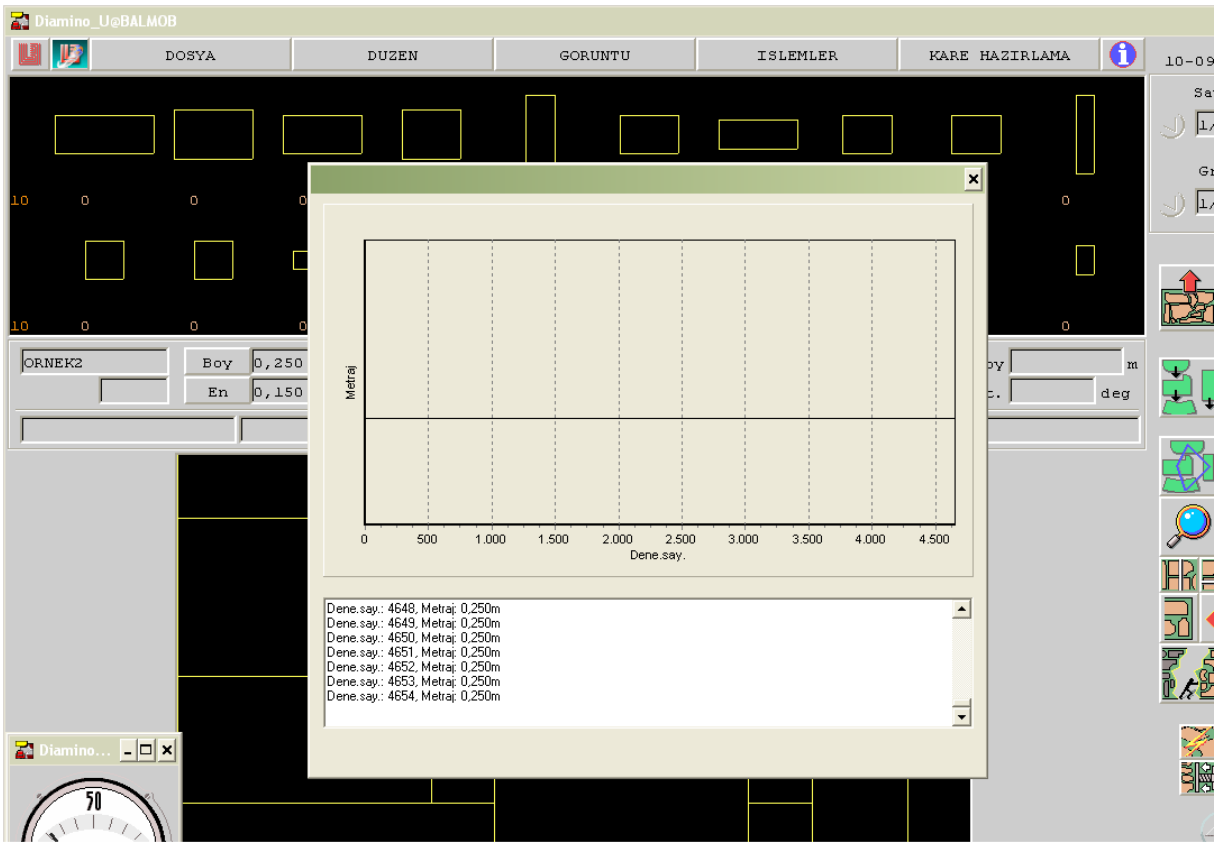
Şekil 4.18: Diamino programı ile Örnek 1' in çözümü

Şekil 4.17 ve 4.18' de görüldüğü gibi 4344 iterasyonda Diamino programı yerleştirmeyi tamamlamış ve % 100 verimlilikle yerleştirme yapmıştır. Yani optimum çözüme Diamino programı bu örnekte ulaşmıştır. Hesaplama süresi 72 saniye sürmüştür.

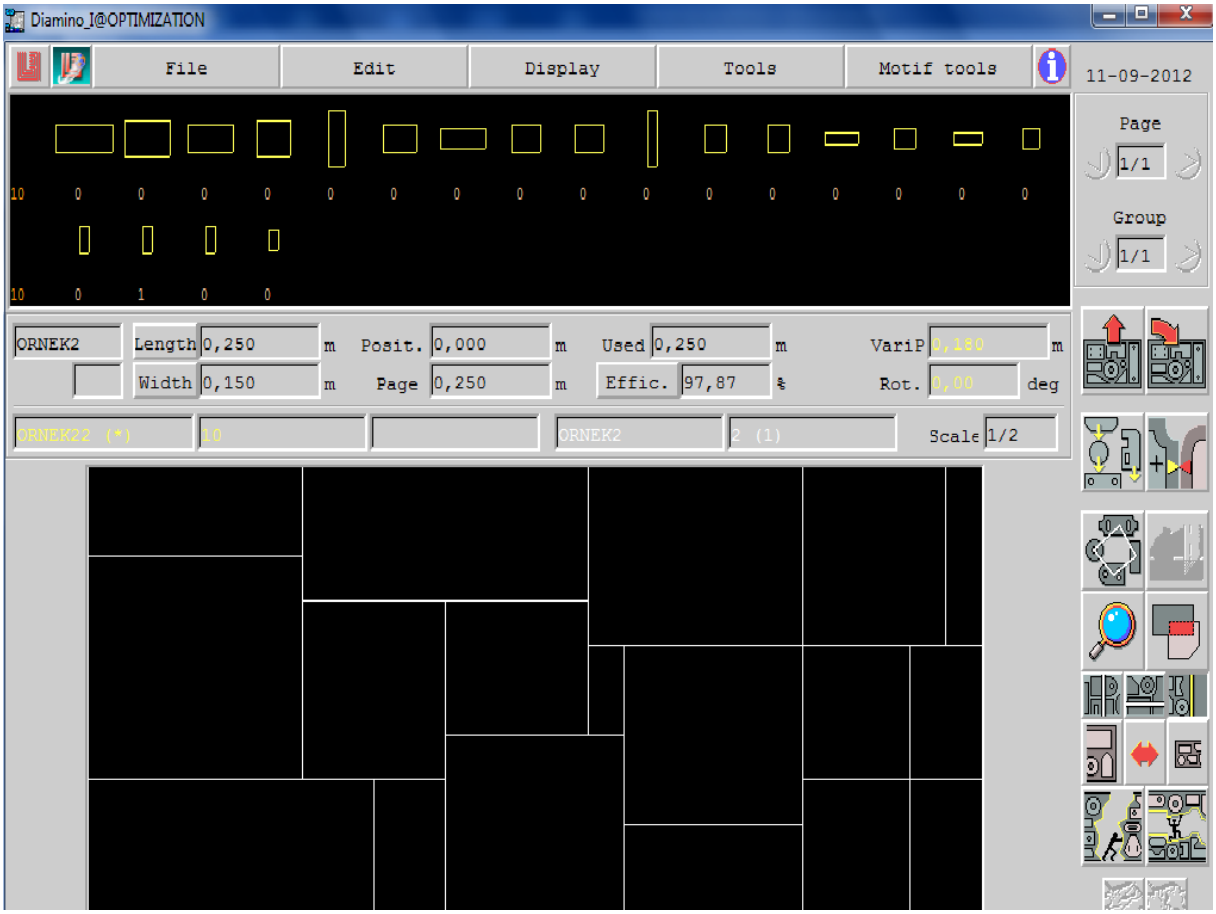
Örnek 2 Diamino programı ile çözülürse;



Şekil 4.19: Formaris programı ile Örnek 2' nin stok malzemeleri



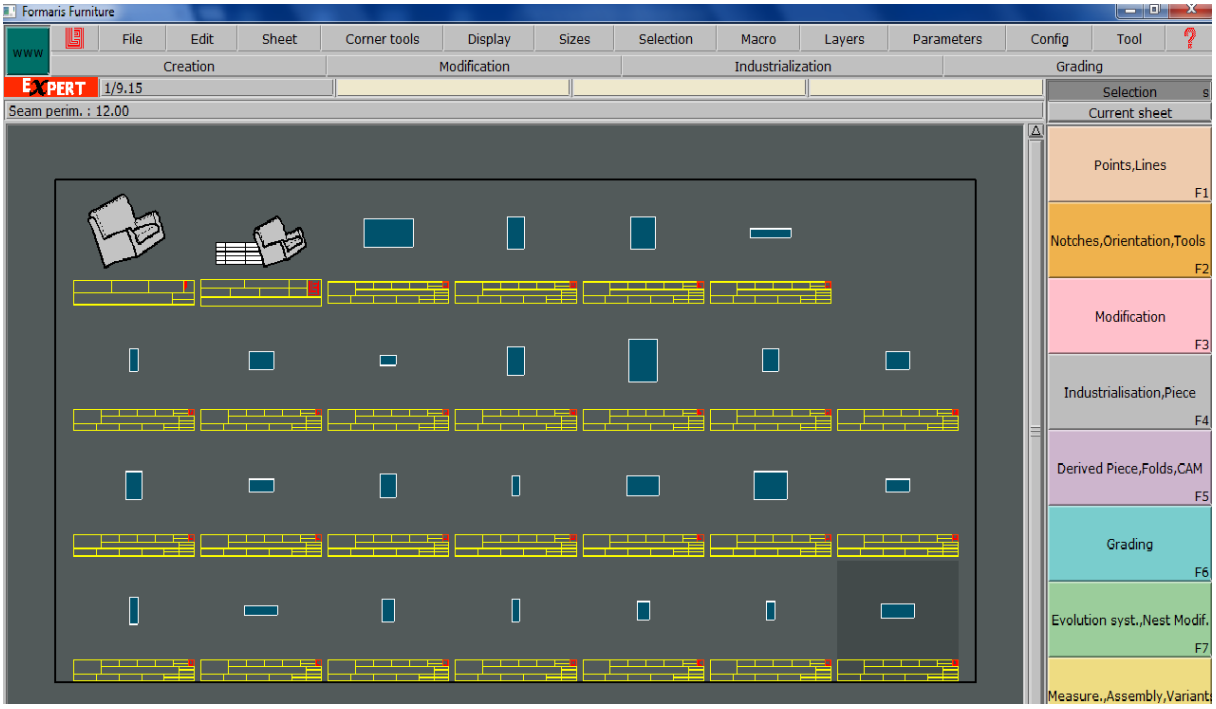
Şekil 4.20: Diamino programı ile Örnek 2' nin iterasyonu



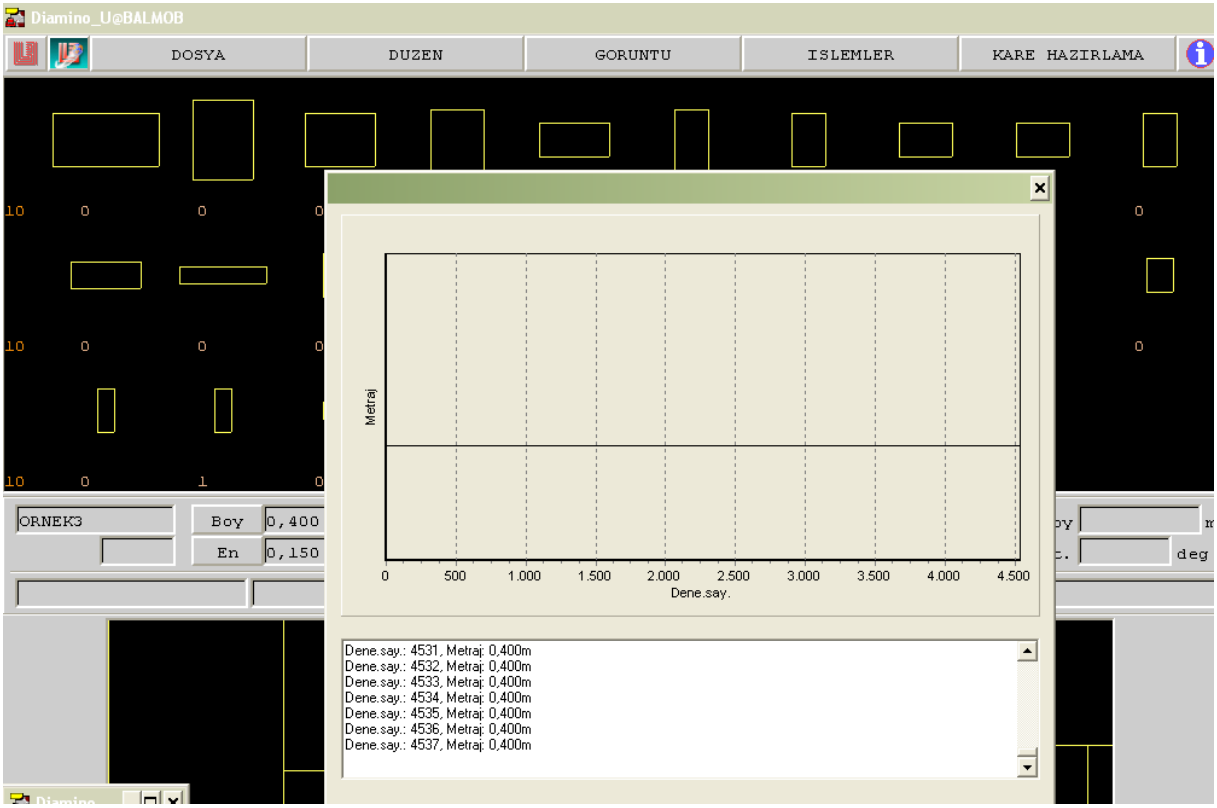
Şekil 4.21: Diamino programı ile Örnek 2' nin çözümü

Şekil 4.20 ve 4.21' de görüldüğü gibi 4654 iterasyonda Diamino programı yerleştirmeyi tamamlamış ve % 97,87 verimlilikle yerleştirme yapmıştır. Optimum çözüme Diamino programı bu örnekte ulaşamamıştır. Hesaplama süresi 39 saniye sürmüştür.

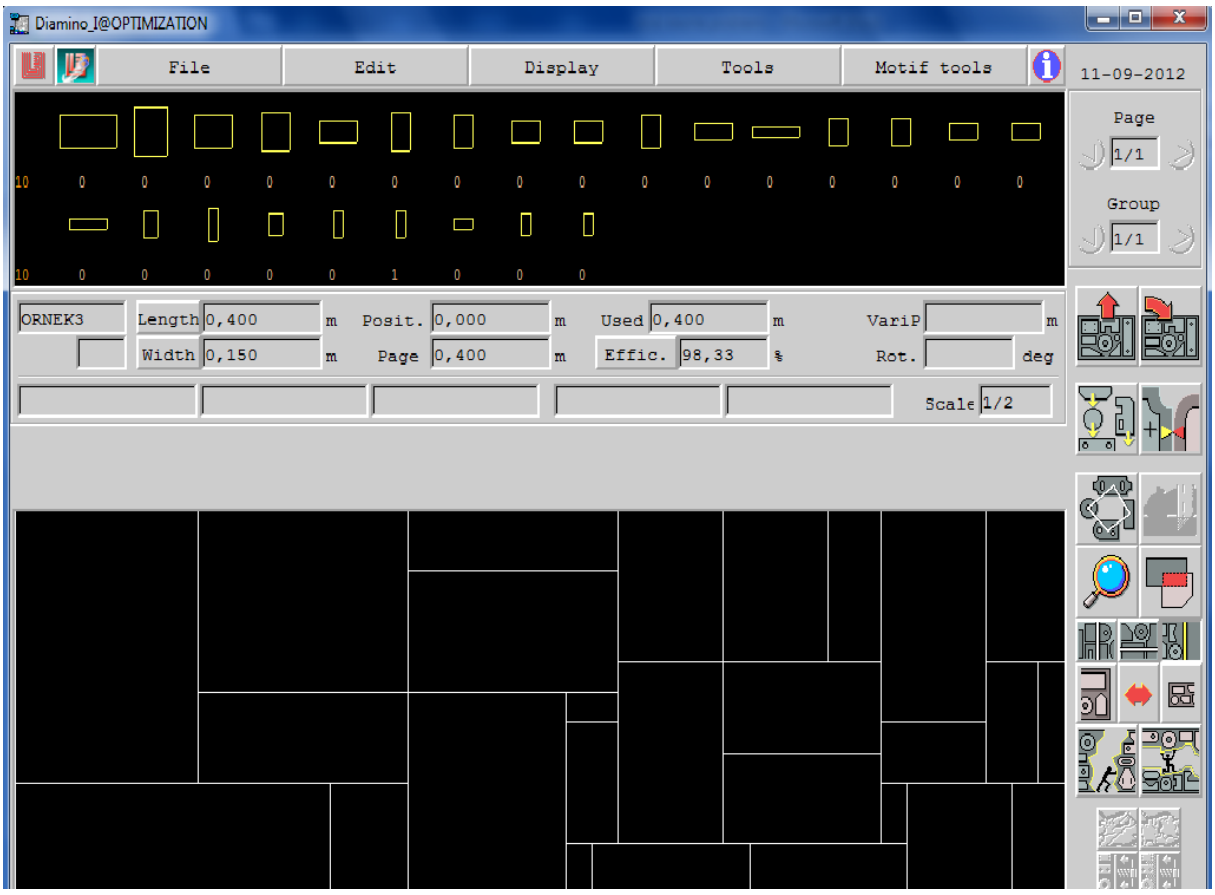
Örnek 3 Diamino programı ile çözülürse;



Şekil 4.22: Formaris programı ile Örnek 3' ün stok malzemeleri



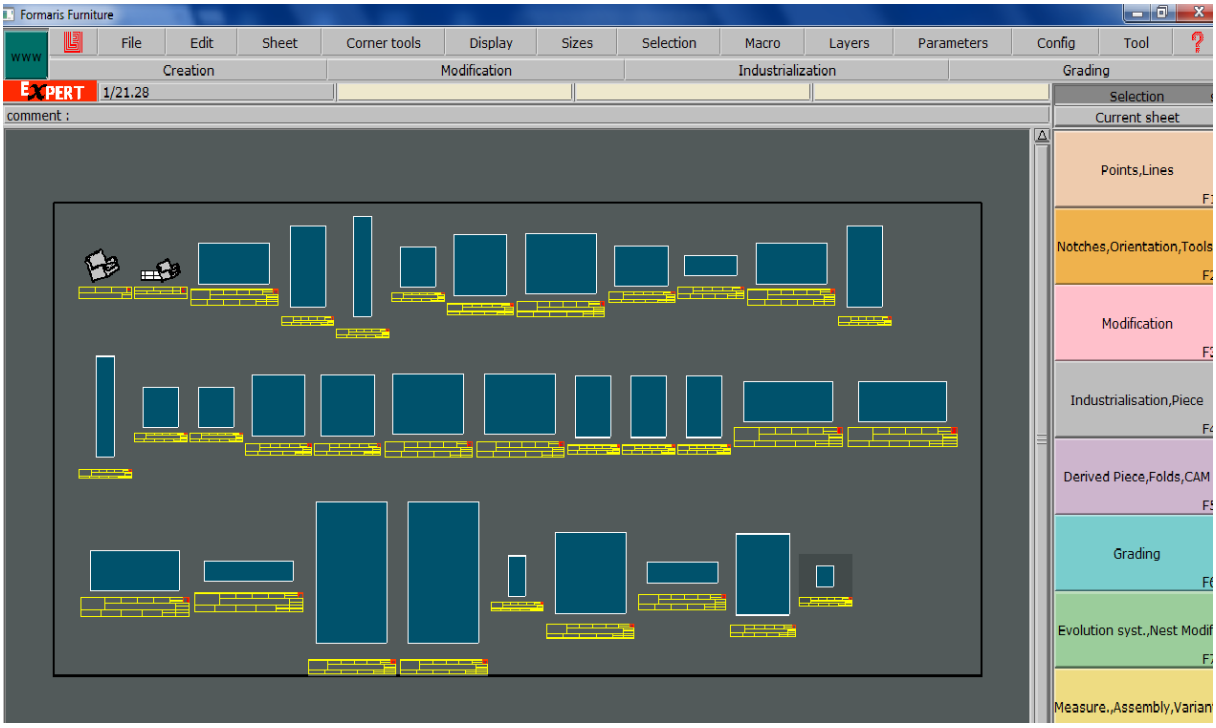
Şekil 4.23: Diamino programı ile Örnek 3' ün iterasyonu



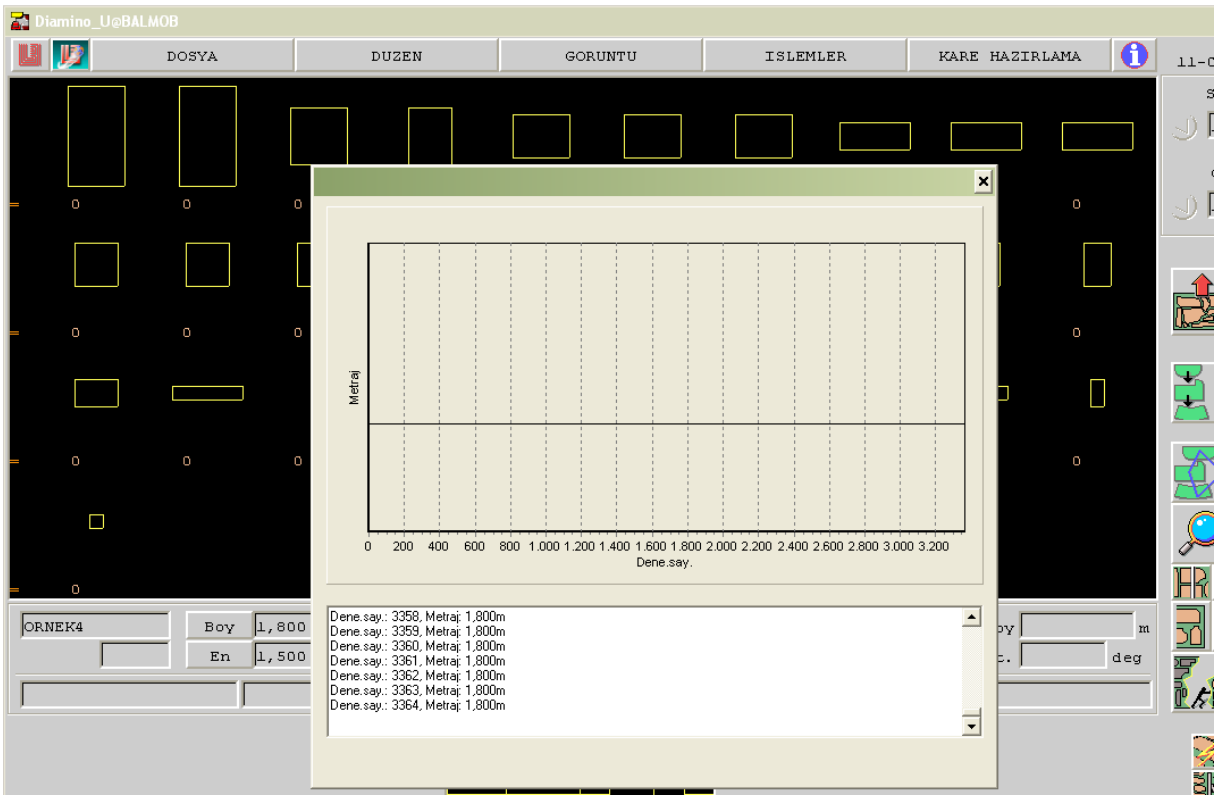
Şekil 4.24: Diamino programı ile Örnek 3' ün çözümü

Şekil 4.23 ve 4.24' de görüldüğü gibi 4537 iterasyonda Diamino programı yerleştirmeyi tamamlamış ve % 98,33 verimlilikle yerleştirme yapmıştır. Optimum çözüme Diamino programı bu örnekte ulaşamamıştır. Hesaplama süresi 78 saniye sürmüştür.

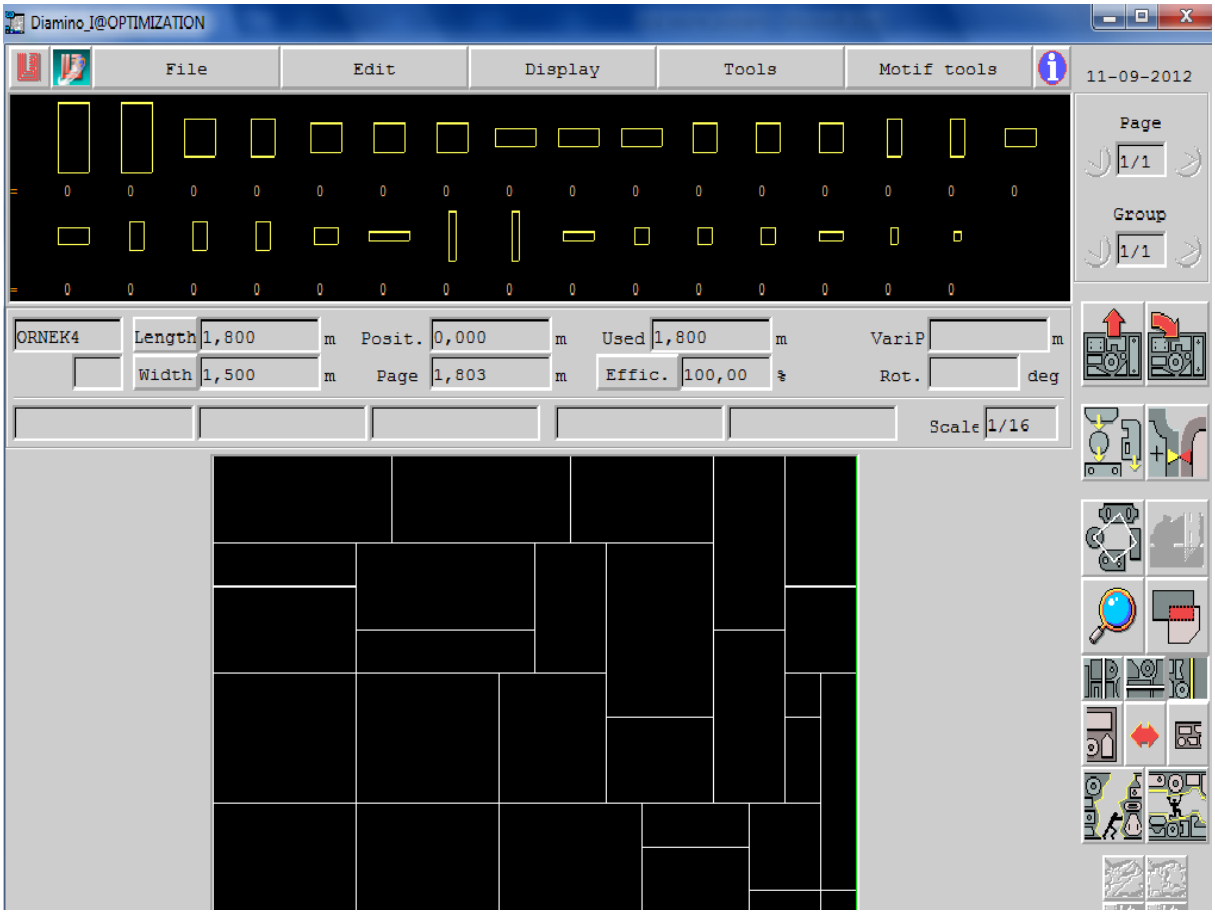
Örnek 4 Diamino programı ile çözülürse;



Şekil 4.25: Formaris programı ile Örnek 4' ün stok malzemeleri



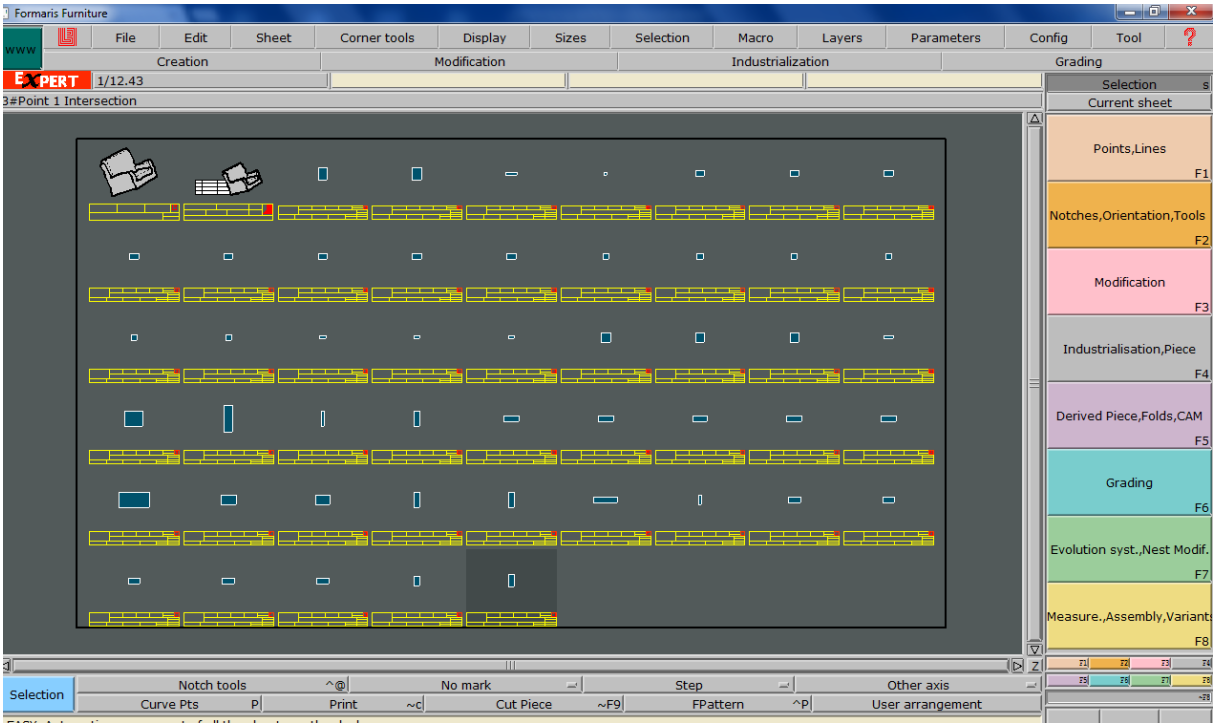
Şekil 4.26: Diamino programı ile Örnek 4' ün iterasyonu



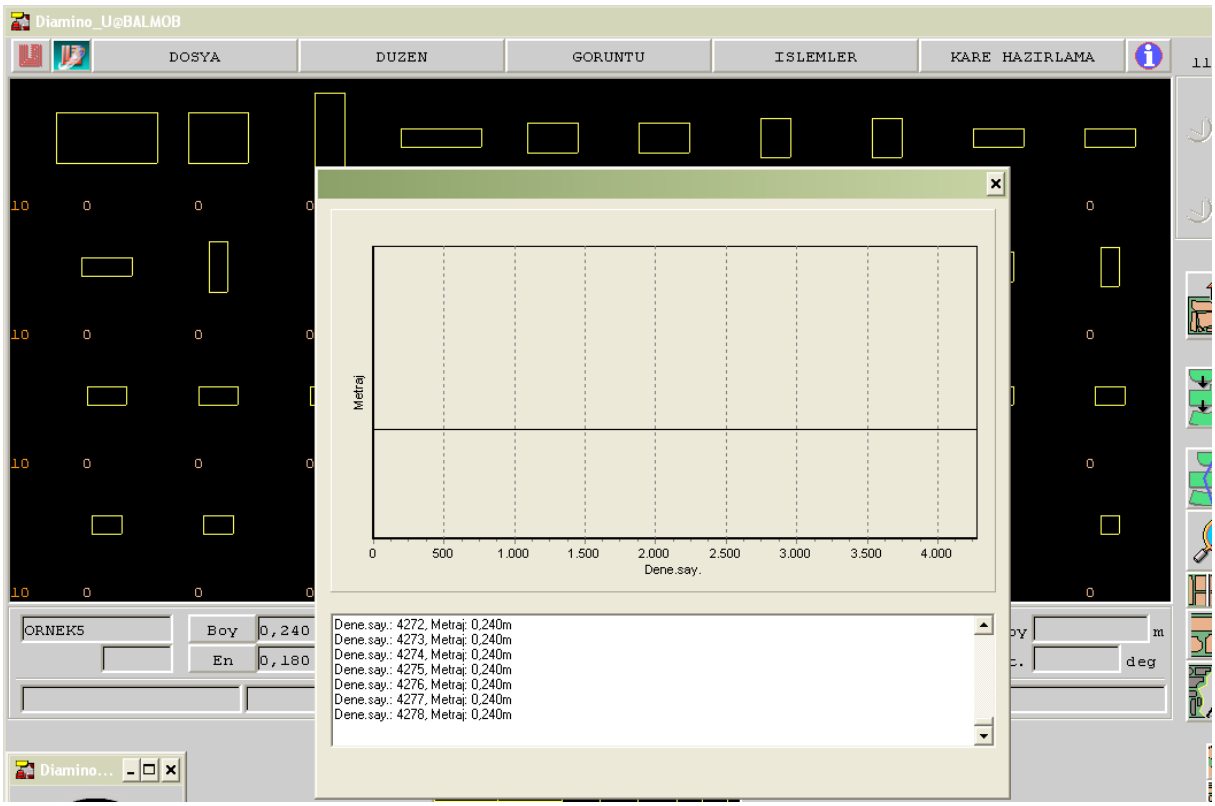
Şekil 4.27: Diamino programı ile Örnek 4' ün çözümü

Şekil 4.26 ve 4.27' de görüldüğü gibi 3364 iterasyonda Diamino programı yerleştirmeyi tamamlamış ve % 100 verimlilikle yerleştirme yapmıştır. Yani optimum çözüme Diamino programı bu örnekte ulaşmıştır. Hesaplama süresi 43 saniye sürmüştür.

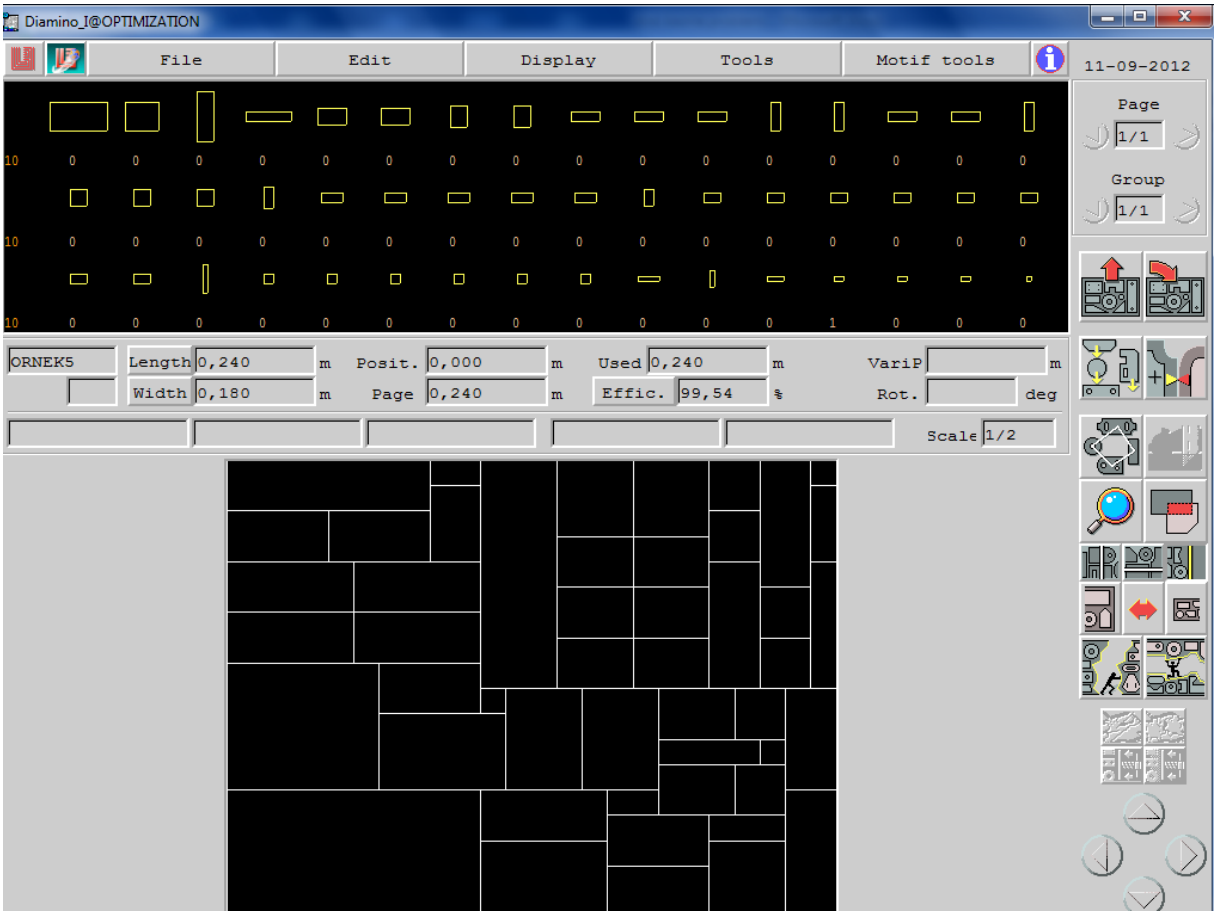
Örnek 5 Diamino programı ile çözülürse;



Şekil 4.28: Formaris programı ile Örnek 5' in stok malzemeleri



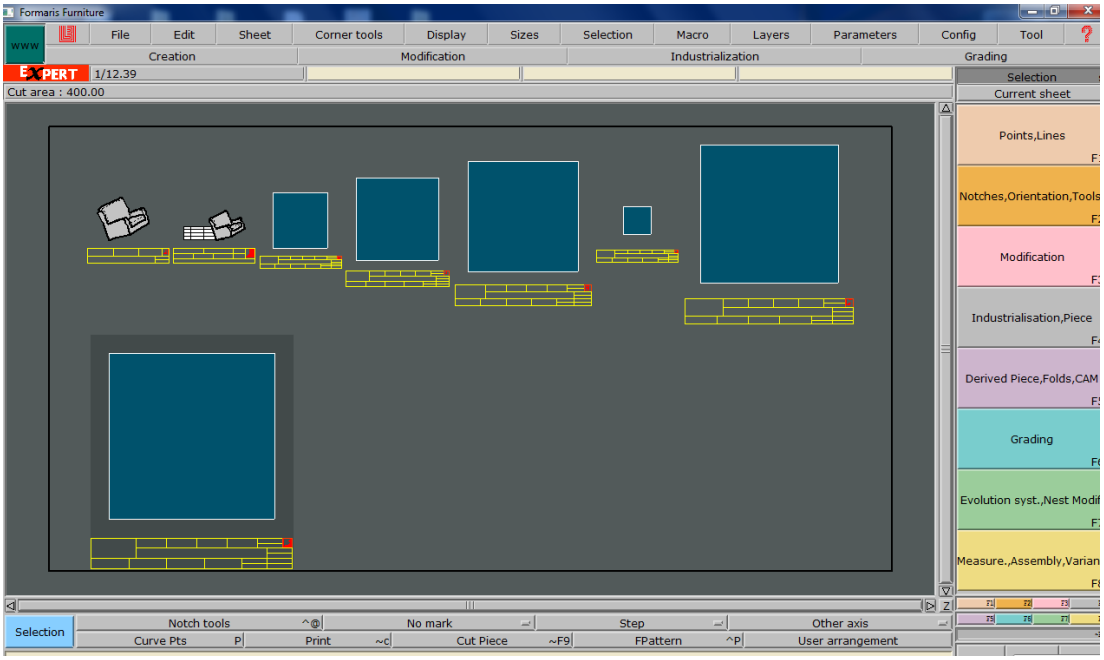
Şekil 4.29: Diamino programı ile Örnek 5' in iterasyonu



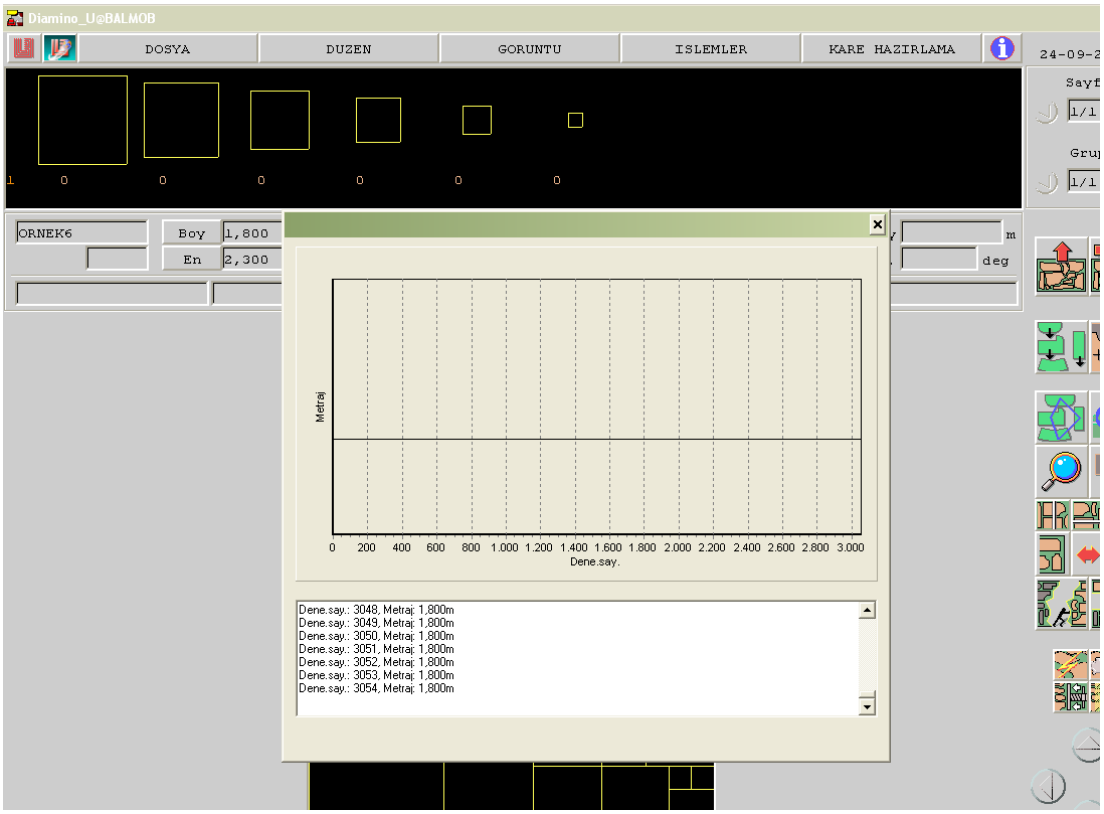
Şekil 4.30: Diamino programı ile Örnek 5' in çözümü

Şekil 4.29 ve 4.30' da görüldüğü gibi 4278 iterasyonda Diamino programı yerleştirmeyi tamamlamış ve % 99,54 verimlilikle yerleştirme yapmıştır. Optimum çözüme Diamino programı bu örnekte ulaşamamıştır. Hesaplama süresi 63 saniye sürmüştür.

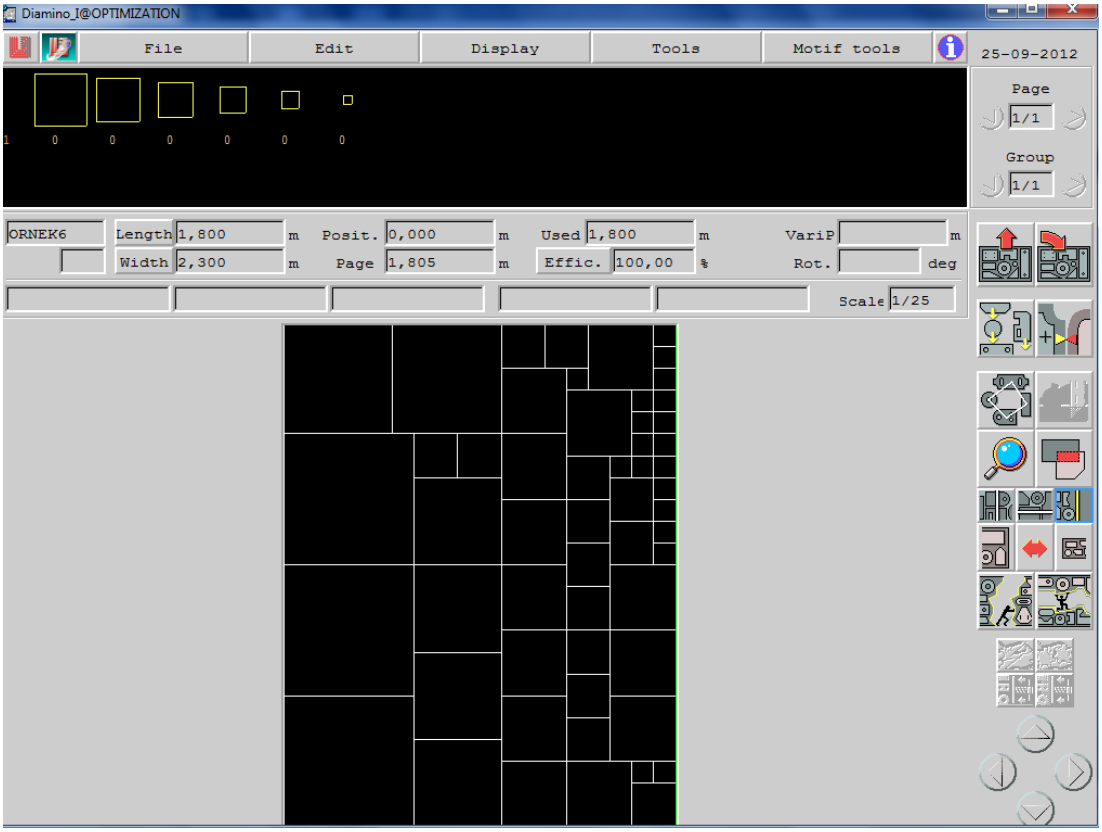
Örnek 6 Diamino programı ile çözülürse;



Şekil 4.31: Formaris programı ile Örnek 6' nın stok malzemeleri



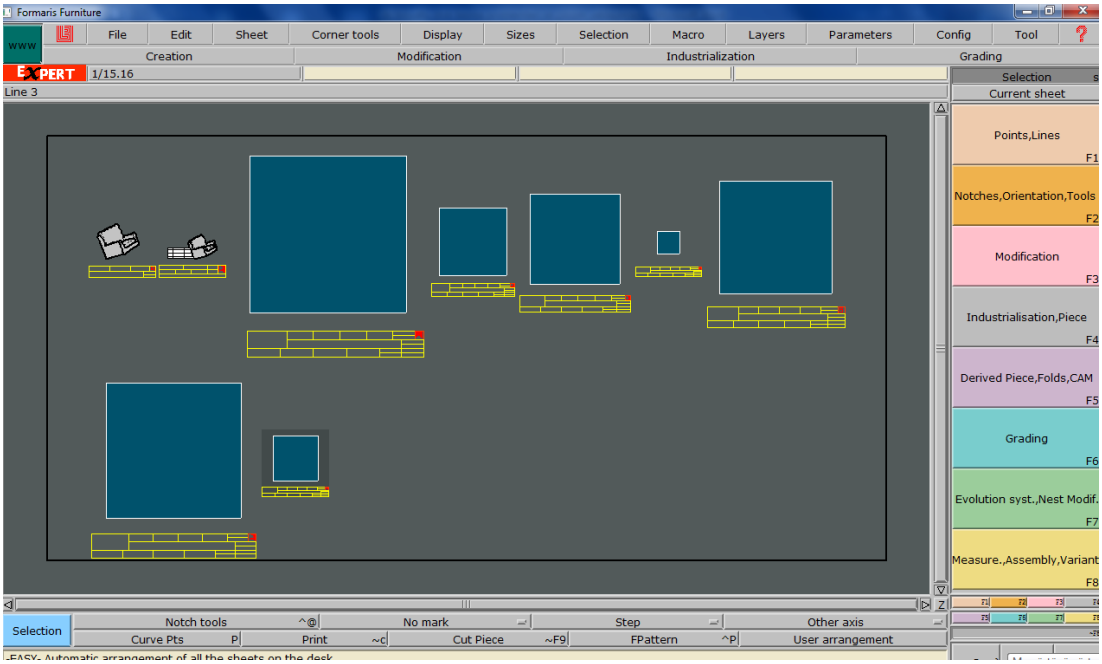
Şekil 4.32: Diamino programı ile Örnek 6' nın iterasyonu



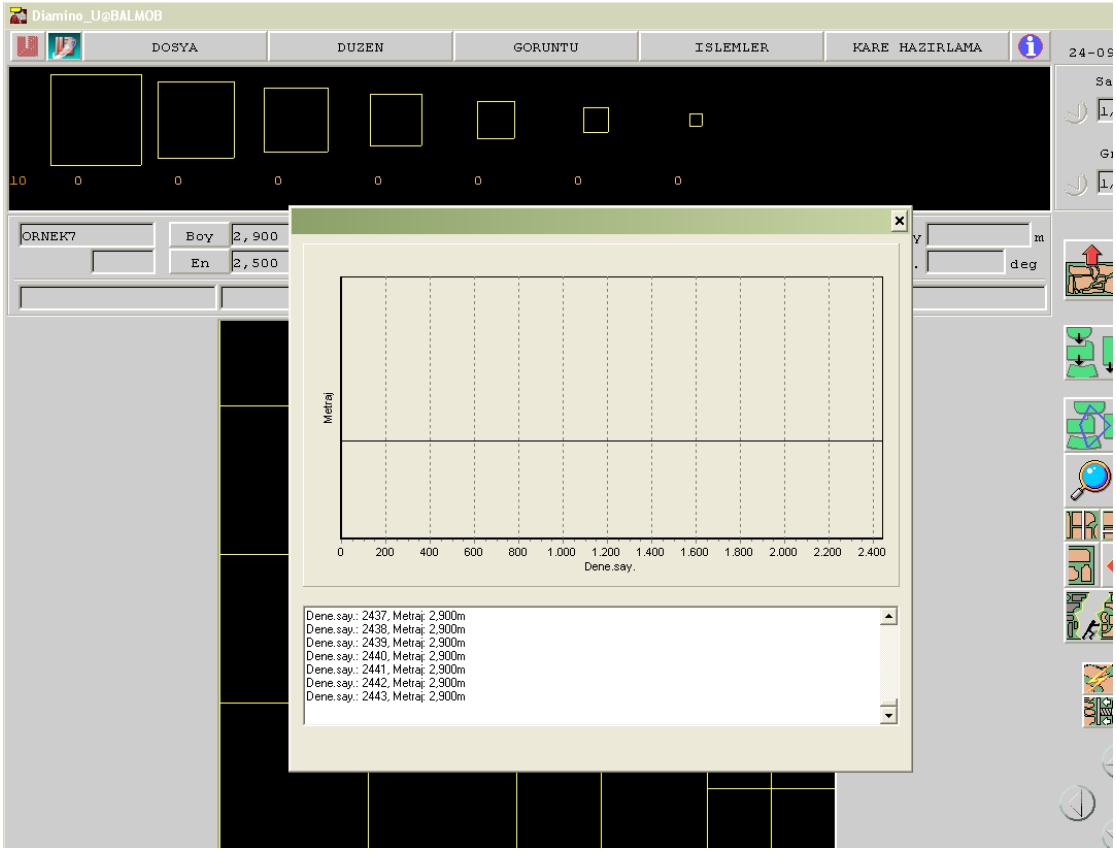
Şekil 4.33: Diamino programı ile Örnek 6' nın çözümü

Şekil 4.32 ve 4.33' de görüldüğü gibi 3054 iterasyonda Diamino programı yerleştirmeyi tamamlamış ve % 100 verimlilikle yerleştirme yapmıştır. Optimum çözüme Diamino programı bu örnekte ulaşmıştır. Hesaplama süresi 73 saniye sürmüştür.

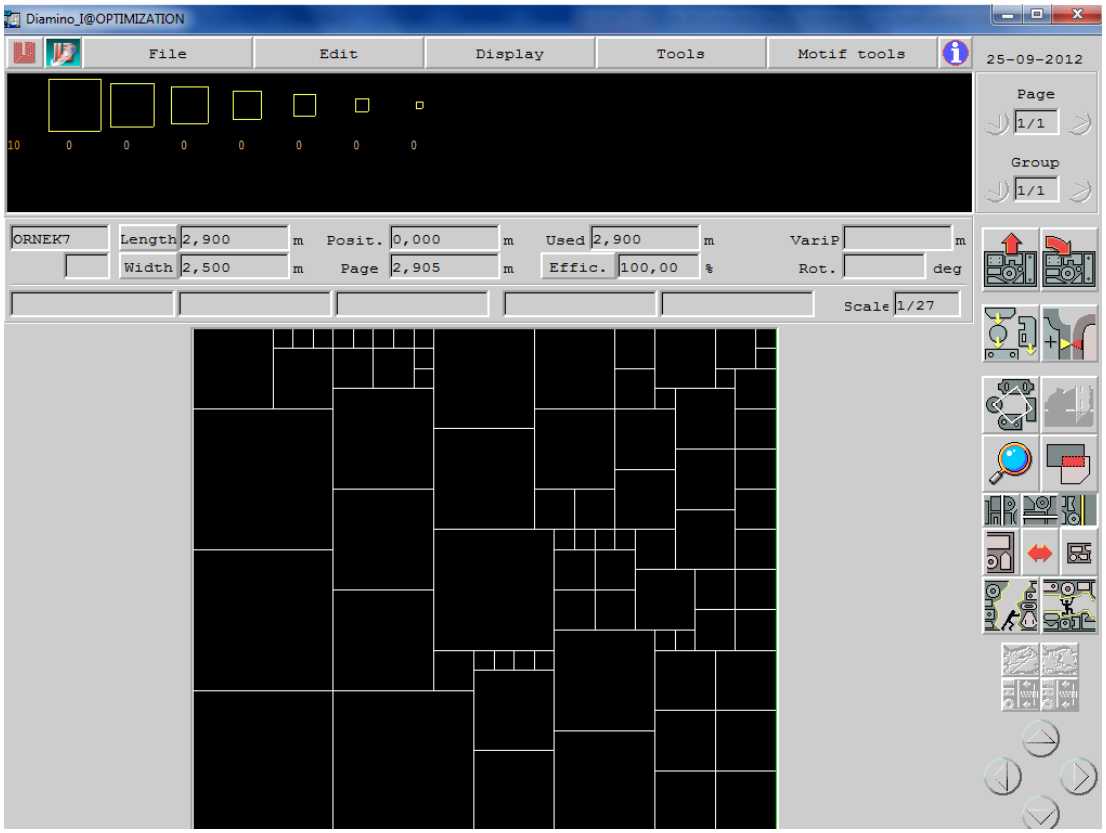
Örnek 7 Diamino programı ile çözülürse;



Şekil 4.34: Formaris programı ile Örnek 7' nin stok malzemeleri



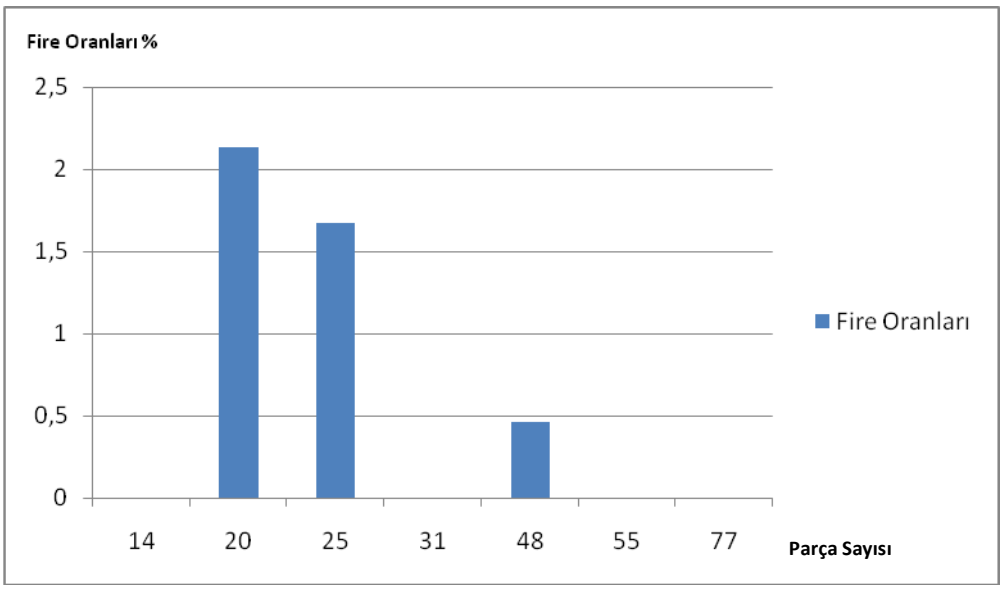
Şekil 4.35: Diamino programı ile Örnek 7' nin iterasyonu



Şekil 4.36: Diamino programı ile Örnek 7' nin çözümü

Şekil 4.35 ve 4.36' da görüldüğü gibi 2443 iterasyonda Diamino programı yerleştirmeyi tamamlamış ve % 100 verimlilikle yerleştirme yapmıştır. Optimum çözüme Diamino programı bu örnekte ulaşmıştır. Hesaplama süresi 45 saniye sürmüştür.

Sonuç olarak Diamino programı ile mevcut yedi örnekten dördünde optimum çözüme ulaşılmıştır. Geliştirilen sezgisel yazılımına göre daha yavaş olmasına rağmen daha iyi sonuçlar vermiştir. Diamino programı ile bu yedi örnekteki fire oranları % 0 ile % 2,5 arasında değişmektedir. Bu da birçok sektörde geçerli bir sonuçtur. Bu sonuçlara ulaşmak için 2400 ile 4700 arasında değişen iterasyon uygulanmıştır. Ayrıca kare şekillerden oluşan SKP' de Diamino programı daha başarılı olmuştur. Şekil 4.37' de Diamino programının test verileri ile performansı değerlendirilmiştir. Şekilde görüldüğü gibi Diamino program kare şekillerden ve benzer ölçülerden oluşan örneklerde fire oranı daha düşüktür. Diamino programı ile kare şekillerden oluşan örneklerde fire oranı 0'a indirilmiştir.



Şekil 4.37: Diamino programı ile test verilerinin performans değerlendirmesi

4.2.2 Geliştirilen Metasezgisel Yazılıma Ait Seçim Algoritması

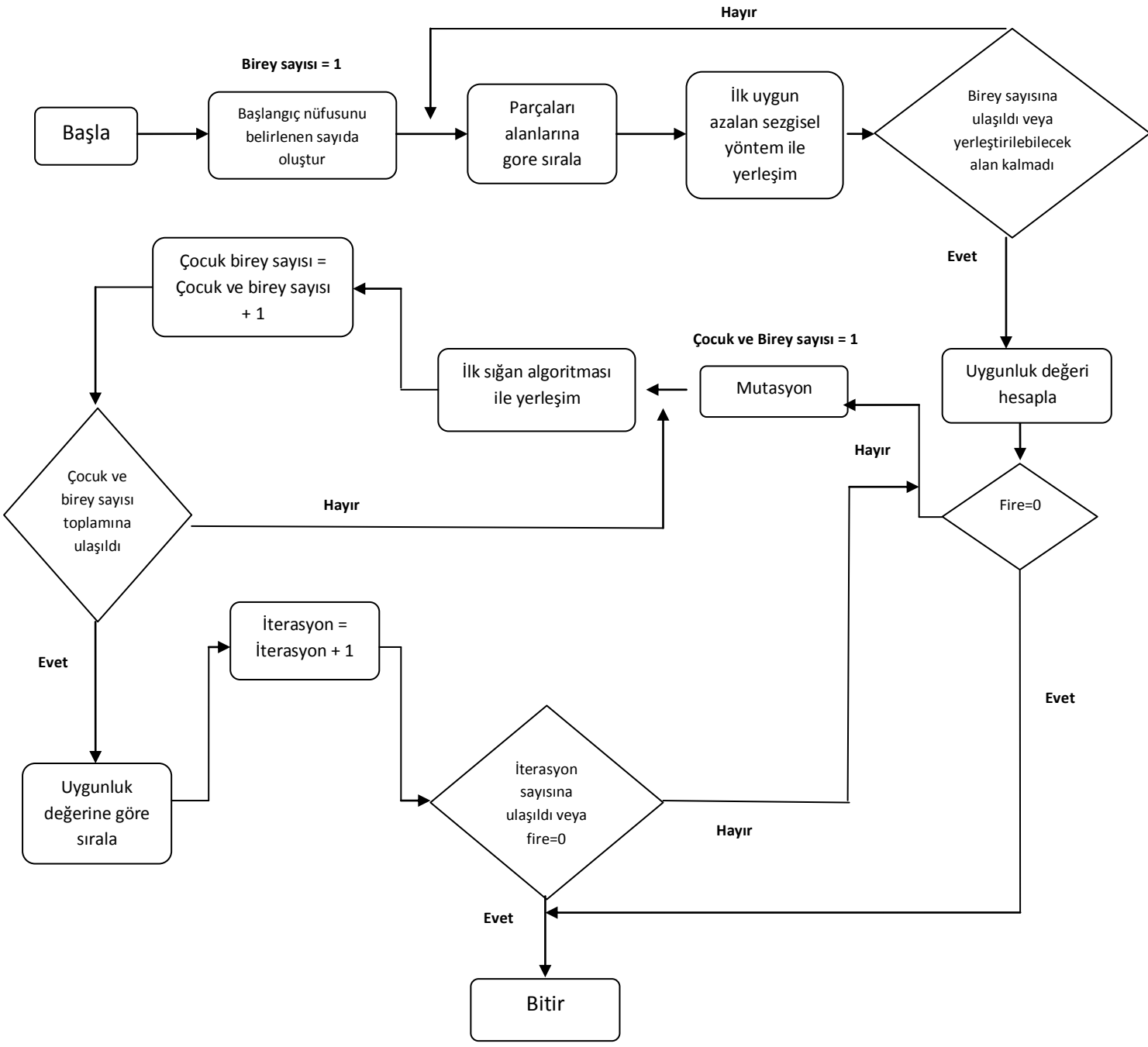
Seçim Algoritması olarak Bölüm 3.4.4' te anlatılan genetik algoritma kullanılmıştır. Genetik algoritma kullanılarak veriler hızlı ve rasgele biçimde üretilir. Bu sayede belirli çözümlere takılmamış, başarılı sonuçlar elde edilmiş olur.

4.2.3 Geliştirilen Metasezgisel Yazılıma Ait Yerleştirme Algoritması

Yerleştirme Algoritması olarak Bölüm 3.1.1' de anlatılan BLF Algoritması kullanılacaktır. Bu yöntemde stok kesim malzemeleri permütasyon sırasına göre yukarı sol kısımdan başlayarak yerleştirilir. Yerleştirme yapılırken arada kalan boşluklar hesaplanır ve o bölgeye uygun parça eşleştirilerek yerleştirilmeye devam edilir. Permütasyon sırası alan büyüklüğü ve genetik algoritma kullanılarak belirlenmiştir. Geliştirilen metasezgisel yazılımda başlangıç noktası olarak sol üst köşe alınmıştır.

4.2.4 Geliştirilen Metasezgisel Yazılım

Geliştirilen metasezgisel yazılımda Genetik Algoritma ve BLF Algoritması kullanılmıştır. Yazılım VB6 ile yazılmıştır. Geliştirilen metasezgisel yazılımda veriler İlk uygun azalan sezgisel yöntem, yeniden üretim ve mutasyonla oluşturulmuştur. Geliştirilen metasezgisel yazılımın veri akış diyagramını Şekil 4.38' de gösterilecek olursa;



Şekil 4.38: Geliştirilen metasezgisel yazılımın veri akış diyagramı

Şekil 4.38' deki veri akış diyagramı detaylı olarak açıklanırsa;

- Öncelikle başlangıç çözümü olarak, parçalar alanlarına göre büyükten küçüğe doğru sıralanır. Permütasyon sırası alanlarının büyüklüğü ile doğru orantılıdır. BLF algoritması kullanılarak yerleştirme başlanır ve birey sayısı tamamlanıncaya kadar ya da yerleştirilebilecek alan kalmayana kadar yerleştirmeye devam edilir, uygunluk ve fire değeri hesaplanır.
- Uygunluk değeri hesaplanmasında öncelikle fire değeri bulunur.

Fire = (Ana parçanın alanı – Kullanılan alan) / Ana parçanın alanı **(4.2)**

Uygunluk fonksiyonu = 1 / (ε + Fire değeri) **(4.3)**

Uygunluk fonksiyonunun bölme hatası vermesini engellemek için bir ε değeri kullanılmıştır. Bu ε değeri 2.2204×10^{-16} gibi çok küçük bir değerdir [28].

- Yeni yerleşim planları oluşturmak için nüfus üzerinde mutasyon genetik işlemcisi kullanılır. Mutasyon oranı 0.01 alınmıştır.
- Yerleştirme algoritması olarak BLF algoritması kullanılır. Her yeni birey için fire değeri hesaplanır.
- Durdurma kriteri olarak iterasyon sayısı ya da fire değerinin 0 olması dikkate alınır. Durdurma kriteri sağlandığında uygunluk değeri en yüksek ya da fire değeri en düşük çocuk seçilir.

Geliştirilen metasezgisel yazılım ile Örnek1, Örnek2, Örnek3, Örnek4, Örnek6 ve Örnek7 çözülürse;

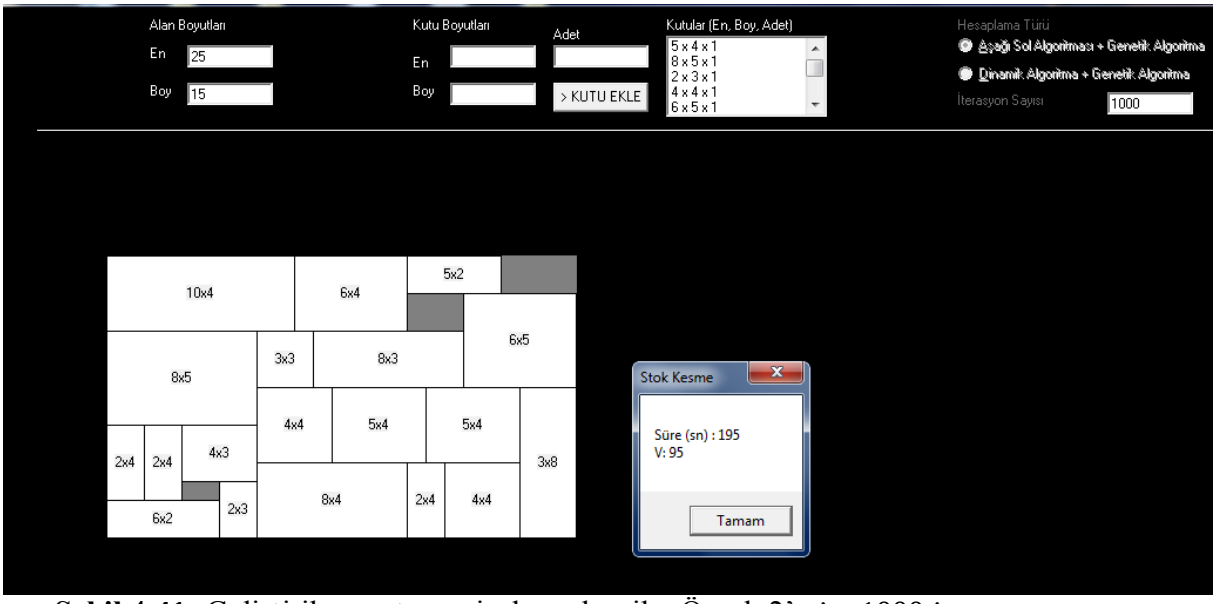
Örnek 1' in geliştirilen metasezgisel yazılım ile çözümü;

Şekil 4.39: Geliştirilen metasezgisel yazılım ile Örnek 1' in 1 iterasyon sonucunda çözümü

Şekil 4.40: Geliştirilen metasezgisel yazılım ile Örnek 1' in 100 iterasyon sonucunda çözümü

Şekil 4.40' da görüldüğü gibi 100 iterasyon sonucunda optimum çözüme ulaşılmıştır.

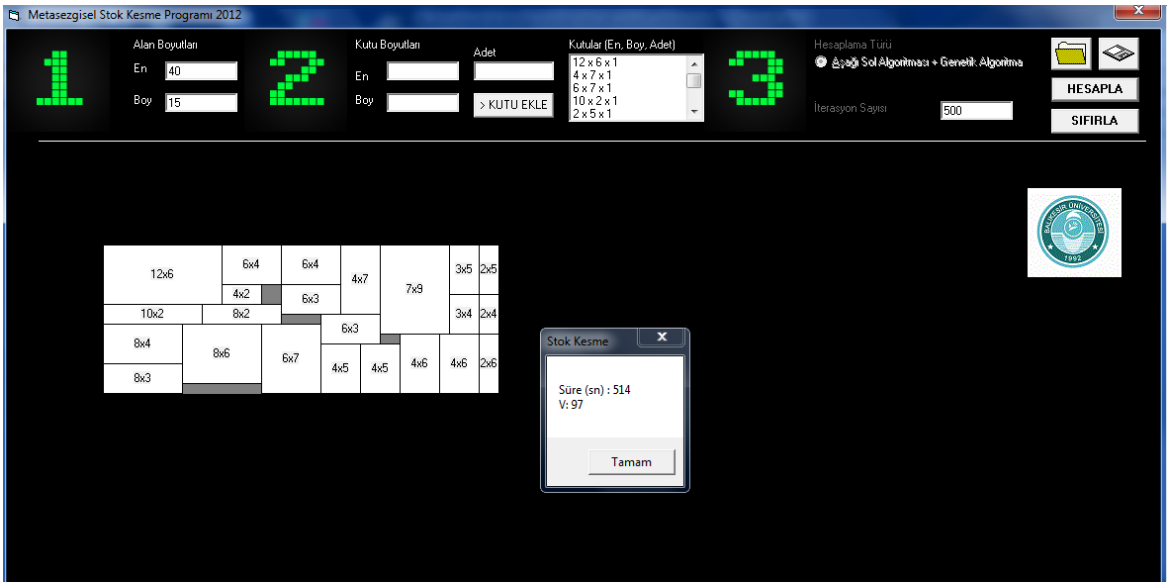
Örnek 2' nin geliştirilen metasezgisel yazılım ile çözümü;



Şekil 4.41: Geliştirilen metasezgisel yazılım ile Örnek 2' nin 1000 iterasyon sonucunda çözümü

Şekil 4.41' de görüldüğü gibi 1000 iterasyon sonucunda optimum çözüme ulaşılamamıştır. Hesaplama süresi ise 195 saniyede gerçekleşmiştir. Verimlilik % 95 seviyesinde gerçekleşmiştir.

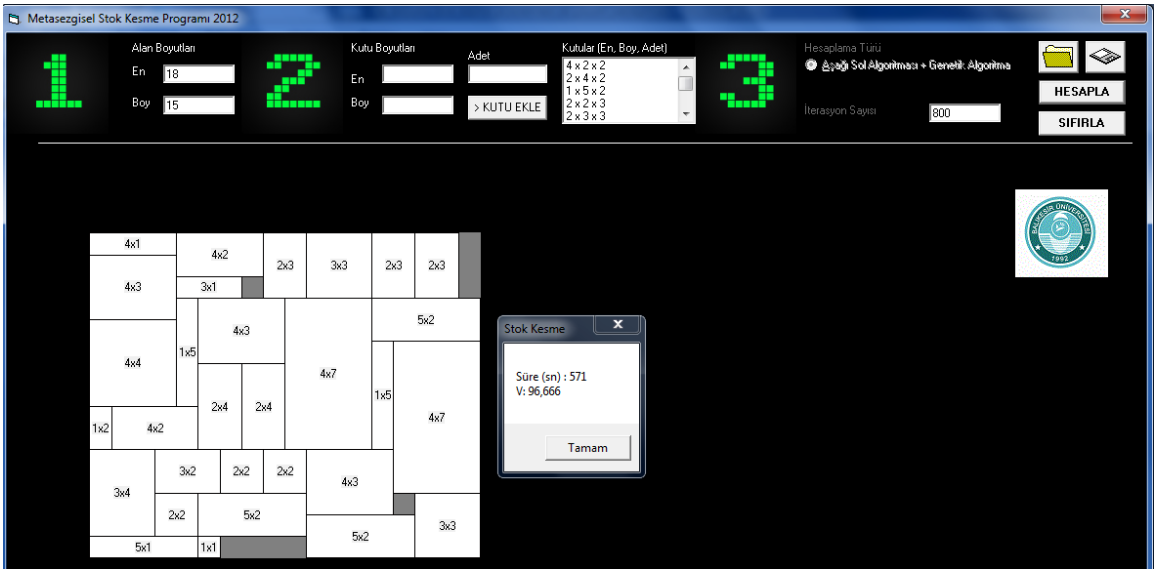
Örnek 3' ün geliştirilen metasezgisel yazılım ile çözümü;



Şekil 4.42: Geliştirilen metasezgisel yazılım ile Örnek 3'ün 500 iterasyon sonucunda çözümü

Şekil 4.42’ de görüldüğü gibi 500 iterasyon sonucunda optimum çözüme ulaşılamamıştır. Hesaplama süresi ise 514 saniyede gerçekleşmiştir. Verimlilik % 97 seviyesinde gerçekleşmiştir.

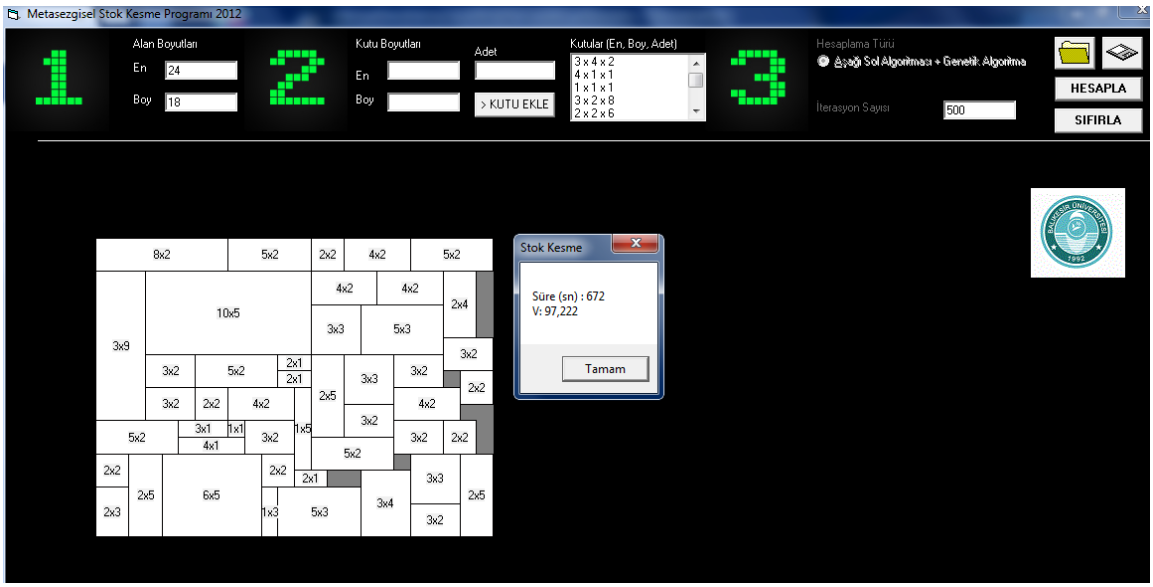
Örnek 4’ ün geliştirilen metasezgisel yazılım ile çözümü;



Şekil 4.43: Geliştirilen metasezgisel yazılım ile Örnek 4’ ün 800 iterasyon sonucunda çözümü

Şekil 4.43’ de görüldüğü gibi 500 iterasyon sonucunda optimum çözüme ulaşılamamıştır. Hesaplama süresi ise 571 saniyede gerçekleşmiştir. Verimlilik % 96,66 seviyesinde gerçekleşmiştir.

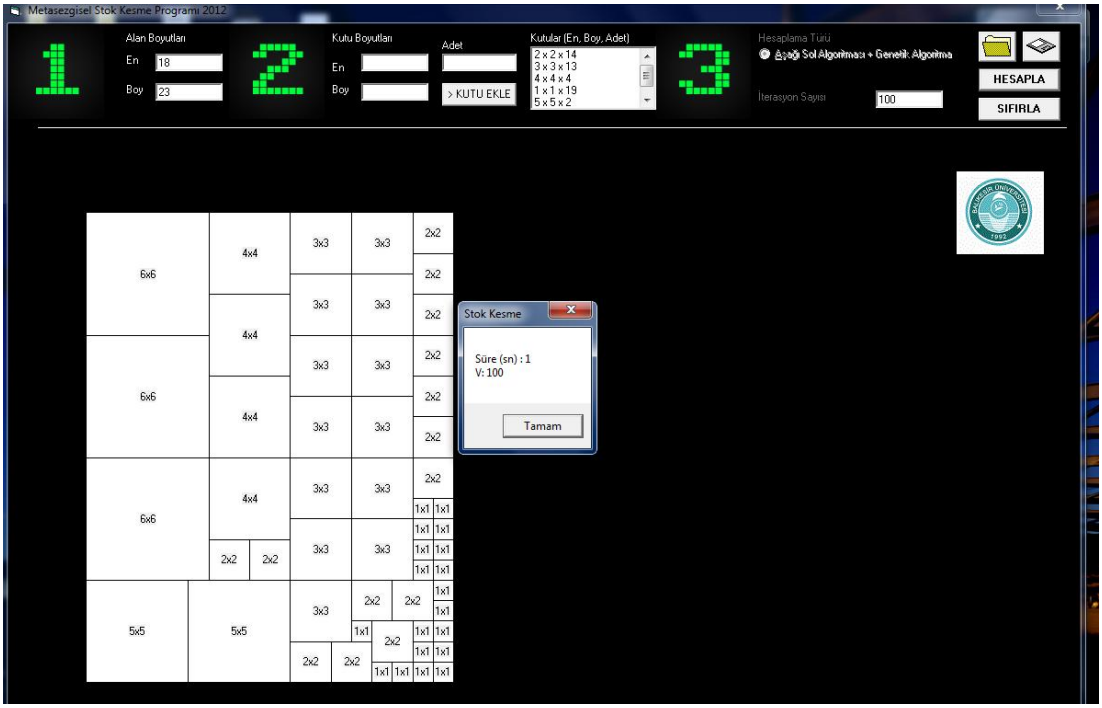
Örnek 5’ in geliştirilen metasezgisel yazılım ile çözümü;



Şekil 4.44: Geliştirilen metasezgisel yazılım ile Örnek 5' in 500 iterasyon sonucunda çözümü

Şekil 4.44' de görüldüğü gibi 500 iterasyon sonucunda optimum çözüme ulaşılamamıştır. Hesaplama süresi ise 672 saniyede gerçekleşmiştir. Verimlilik % 97,222 seviyesinde gerçekleşmiştir.

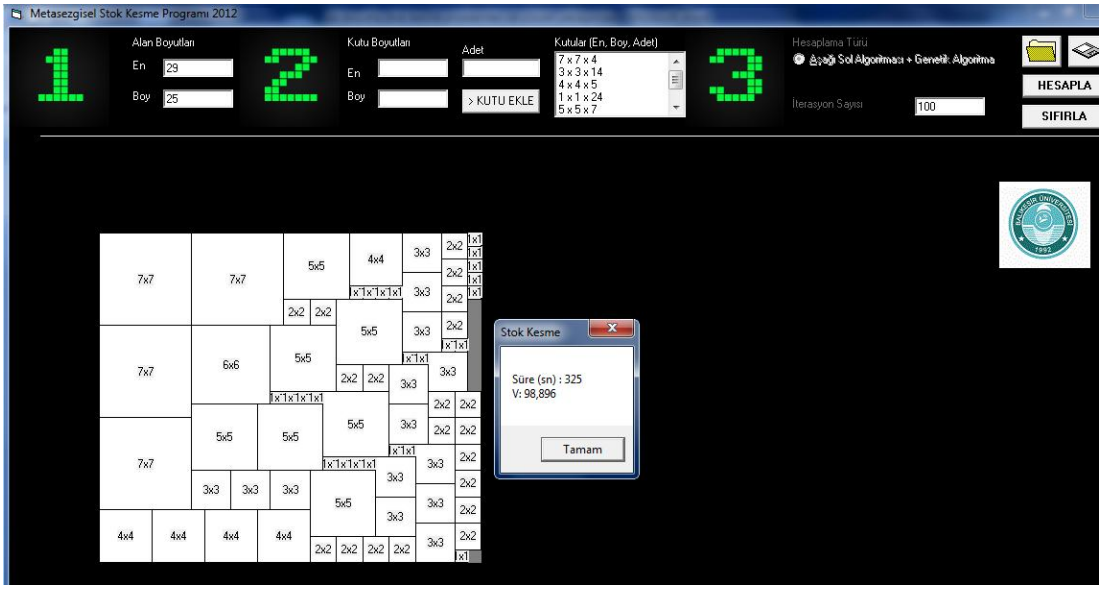
Örnek 6' nın geliştirilen metasezgisel yazılım ile çözümü;



Şekil 4.45: Geliştirilen metasezgisel yazılım ile Örnek 6' nın 100 iterasyon sonucunda çözümü

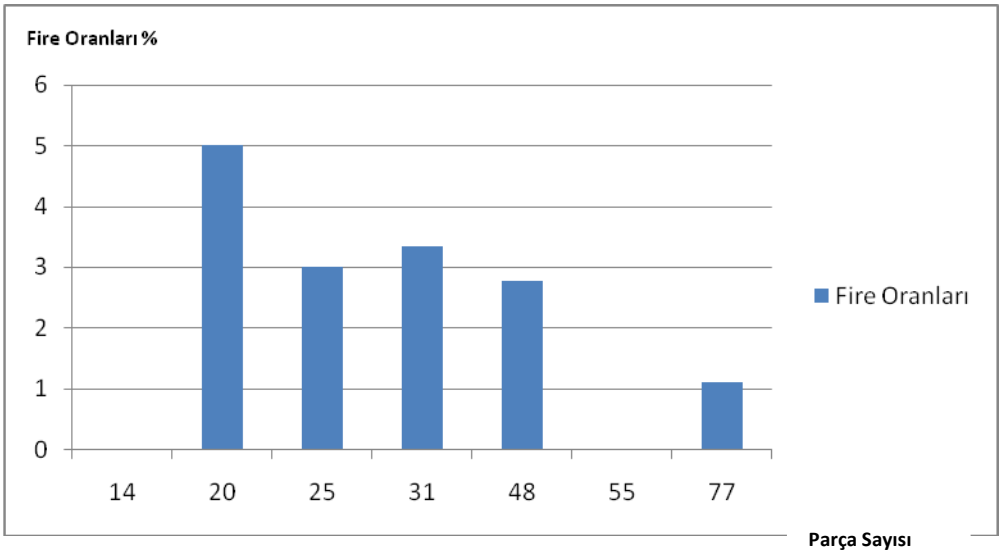
Şekil 4.45’ de görüldüğü gibi 100 iterasyon sonucunda optimum çözüme ulaşılmıştır. Hesaplama süresi ise 1 saniyede gerçekleşmiştir.

Örnek 7’ nin geliştirilen metasezgisel yazılım ile çözümü;



Şekil 4.46: Geliştirilen metasezgisel yazılım ile Örnek 7’ nin 100 iterasyon sonucunda çözümü

Şekil 4.46’ da görüldüğü gibi 100 iterasyon sonucunda optimum çözüme ulaşılamamıştır. Hesaplama süresi ise 325 saniyede gerçekleşmiştir. Verimlilik % 98,896 seviyesinde gerçekleşmiştir. Şekil 4.47’ de geliştirilen metasezgisel algoritma ile mevcut test verilerinin performansı değerlendirilmiştir. İterasyon sayısı örneklerde fire oranını düşürmüştür. Kare şekillerden oluşan örneklerde geliştirilen sezgisel yöntem ile benzer sonuçlar elde edilmiştir. Geliştirilen metasezgisel yöntem, geliştirilen sezgisel yönteme göre dikdörtgen şekillerde daha başarılı sonuçlar vermiştir.



Şekil 4.47: Geliştirilen metasezgisel yazılımın mevcut test verileri ile performansının değerlendirilmesi

5. SONUÇ VE GELECEK ARAŞTIRMA

Rekabetin her geçen gün arttığı günümüzde fiyat noktasında avantaj sağlayan firmalar rakiplerine göre bir adım öndedir. Fiyat avantajını sağlamak için en önemli husus fire oranlarıdır. Bu noktada stok kesme problemlerine olan ilgi her geçen gün artmaktadır. Fakat optimal sonucu polinomial zamanda veren sezgisel ve metasezgisel algoritma henüz geliştirilememiştir.

Bu çalışmada iki boyutlu dikdörtgen yapıda elemanlar için stok kesme problemlerine çözüm yaklaşımları incelenmiştir. Sezgisel ve metasezgisel çözüm yazılımı geliştirilmiştir. Bu yazılımın performansı optimumu verilen mevcut test verileri kullanılarak değerlendirilmiş ve başka bir stok kesme yazılımı olan ve ticari olarak kabul görmüş Lectra firmasına ait Diamino programı ile karşılaştırılmıştır. Lectra firması 40 yıllık bir deneyime sahip olan firma olduğu düşünüldüğünde geliştirilen yazılım ile oldukça başarılı sonuçlar elde edilmiştir.

Örnek 1 geliştirilen sezgisel yazılım ile çözüldüğünde, 4 saniyede çözülmüş ve verimlilik % 95.714' tür. Diamino programı ile çözüldüğünde, 4344 iterasyonda 72 saniyede çözülmüş ve % 100 verimlilikle yerleştirme yapılmıştır. Yani optimum çözüme Diamino programı bu örnekte ulaşmıştır. Geliştirilen metasezgisel yazılım ile çözüldüğünde, 100 iterasyonda 1 saniyede çözülmüş ve % 100 verimlilikle yerleştirme yapılmıştır. Yani optimum çözüme geliştirilen metasezgisel yazılım ile ulaşılmıştır. 14 parçalık bu stok kesme probleminde geliştirilen metasezgisel yazılım oldukça iyi sonuç vermiştir.

Örnek 2 geliştirilen sezgisel yazılım ile çözüldüğünde, 0-1 saniyede çözülmüş ve verimlilik % 93.6' dır. Diamino programı ile çözüldüğünde, 4654 iterasyonda 39 saniyede çözülmüş ve % 97.87 verimlilikle yerleştirme yapılmıştır. Yani optimum çözüme Diamino programı bu örnekte ulaşamamıştır. Geliştirilen metasezgisel yazılım ile çözüldüğünde, 1000 iterasyonda 195 saniyede çözülmüş ve % 95 verimlilikle yerleştirme yapılmıştır. Yani optimum çözüme geliştirilen metasezgisel yazılım ile ulaşılamamıştır. 20 parçalık bu stok kesme probleminde geliştirilen metasezgisel yazılım Diamino yazılımına yakın sonuç vermiştir.

Örnek 3 geliştirilen sezgisel yazılım ile çözüldüğünde, 0-1 saniyede çözülmüş ve verimlilik % 94' dür. Diamino programı ile çözüldüğünde, 4537 iterasyonda 78 saniyede çözülmüş ve % 98.33 verimlilikle yerleştirme yapılmıştır. Yani optimum çözüme Diamino programı bu örnekte ulaşamamıştır. Geliştirilen metasezgisel yazılım ile çözüldüğünde, 500 iterasyonda 514 saniyede çözülmüş ve % 97 verimlilikle yerleştirme yapılmıştır. 25 parçalık bu stok kesme probleminde geliştirilen metasezgisel yazılım Diamino yazılımına yakın sonuç vermiştir.

Örnek 4 geliştirilen sezgisel yazılım ile çözüldüğünde, 28 saniyede çözülmüş ve verimlilik % 97.03' dür. Diamino programı ile çözüldüğünde, 3364 iterasyonda 43 saniyede çözülmüş ve %100 verimlilikle yerleştirme yapılmıştır. Yani optimum çözüme Diamino programı bu örnekte ulaşmıştır. Geliştirilen metasezgisel yazılım ile çözüldüğünde, 800 iterasyonda 571 saniyede çözülmüş ve %96,666 verimlilikle yerleştirme yapılmıştır. Yani optimum çözüme geliştirilen metasezgisel yazılım ile ulaşılamamıştır. 31 parçalık bu stok kesme probleminde geliştirilen Diamino programı ile başarılı sonuç elde edilmiştir.

Örnek 5 geliştirilen sezgisel yazılım ile çözüldüğünde, 0-1 saniyede çözülmüş ve verimlilik % 95.6' dır. Diamino programı ile çözüldüğünde, 4278 iterasyonda 63 saniyede çözülmüş ve %99.54 verimlilikle yerleştirme yapılmıştır. Yani optimum çözüme Diamino programı bu örnekte ulaşamamıştır. Geliştirilen metasezgisel yazılım ile çözüldüğünde, 500 iterasyonda 672 saniyede çözülmüş ve % 97,222 verimlilikle yerleştirme yapılmıştır. Yani optimum çözüme geliştirilen metasezgisel yazılım ile ulaşılamamıştır. 48 parçalık bu stok kesme probleminde geliştirilen metasezgisel yazılım Diamino yazılımına yakın sonuç vermiştir.

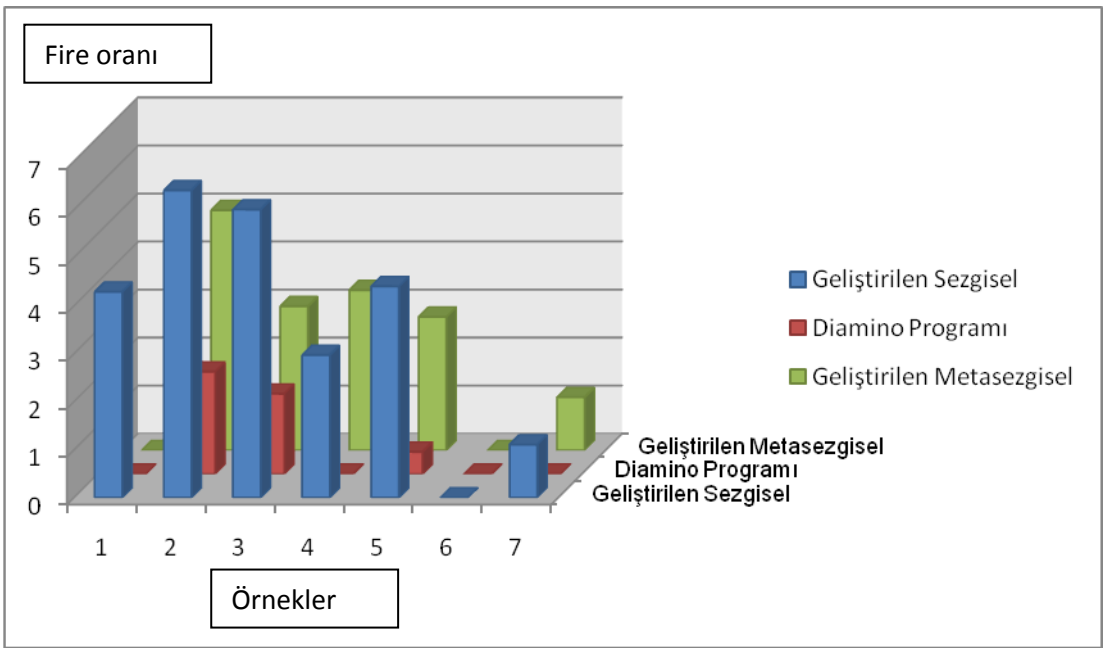
Örnek 6 geliştirilen sezgisel yazılım ile çözüldüğünde, 0-1 saniyede çözülmüş ve verimlilik % 100' dür. Diamino programı ile çözüldüğünde, 3054 iterasyonda 73 saniyede çözülmüş ve %100 verimlilikle yerleştirme yapılmıştır. Geliştirilen metasezgisel yazılım ile çözüldüğünde, 100 iterasyonda 1 saniyede çözülmüş ve % 100 verimlilikle yerleştirme yapılmıştır. 55 parçalık bu stok kesme probleminde geliştirilen sezgisel yazılım Diamino yazılımı ve geliştirilen metasezgisel yazılım ile optimum çözüme ulaşılmıştır.

Örnek 7 geliştirilen sezgisel yazılım ile çözüldüğünde, 1 saniyede çözülmüş ve verimlilik % 98,896' dır. Diamino programı ile çözüldüğünde, 2443 iterasyonda 45 saniyede çözülmüş ve % 100 verimlilikle yerleştirme yapılmıştır. Yani optimum çözüme Diamino programı bu örnekte ulaşmıştır. Geliştirilen metasezgisel yazılım ile çözüldüğünde, 100 iterasyonda 325 saniyede çözülmüş ve % 98,896 verimlilikle yerleştirme yapılmıştır. Yani optimum çözüme geliştirilen metasezgisel yazılım ile ulaşılamamıştır. 77 parçalık bu stok kesme probleminde geliştirilen sezgisel ve metasezgisel yazılım optimum çözüme ulaşamamıştır.

Örneklerin sonuçları Tablo 5.1' de incelendiğinde;

Tablo 5.1: Test örneklerinin karşılaştırmalı incelenmesi

Örnek No	**Parça Tipi**	**Parça Sayısı**	**Geliştirilen Sezgisel Yazılım**		**Diamino Programı**		**Geliştirilen Metasezgisel Yazılım**	
			İterasyon Sayısı	**Fire (%)**	**İterasyon Sayısı**	**Fire (%)**	**İterasyon Sayısı**	**Fire (%)**
Örnek 1	Dikdörtgen	14	1	4,286	4344	0	100	0
Örnek 2	Dikdörtgen	20	1	6,4	4654	2,13	1000	5
Örnek 3	Dikdörtgen	25	1	6	4537	1,67	500	3
Örnek 4	Dikdörtgen	31	1	2,97	3364	0	800	3,334
Örnek 5	Dikdörtgen	48	1	4,4	4278	0,46	500	2,778
Örnek 6	Kare	55	1	0	3054	0	100	0
Örnek 7	Kare	77	1	1,104	2443	0	100	1,104



Şekil 5.1: Test örneklerinin fire oranlarının karşılaştırmalı incelenmesi

Örneklerde görüldüğü gibi geliştirilen metasezgisel yazılım birbirine benzer ve az sayıda parçadan oluşan stok kesme problemlerinde daha başarılı sonuçlar vermektedir. Geliştirilen sezgisel yazılım ise 0 ile 28 saniye arasında başarılı sonuçlara ulaşabilmektedir. Performans açısından ise Diamino programı en iyi sonuçları üreten yazılım olmuştur. Geliştirilen metasezgisel yazılım 1000 iterasyondan fazla iterasyon sayısında yerel optimumda takılmaktadır. Kare şekillerden oluşan problemlerde Diamino programı başarılı sonuçlar vermiştir. Geliştirilen metasezgisel yazılımın veri araması geliştirilerek Diamino programından daha başarılı sonuçlar elde edilebilir.

Gelecek araştırma ile ilgili olarak geliştirilen metasezgisel yazılımın iterasyon kısmı geliştirilip, kullanıcıya stok malzemesi ile ilgili önermeler bildirecek hale getirilmelidir. Stok kesme problemlerinde en önemli sorun yerleştirmenin % 100 verimlilikle yapılıp yapılamayacığının bilinmemesidir. Bu noktada örneğin tekstil sektöründe yapılan yerleştirme sonucunda kullanıcıya önermelerde bulunulabilir. Örneğin fire olan kısmı yok etmek için büyük stok malzemesi daha küçük parçalara bölünüp dikiş ile birleştirilebilir. Bu sayede önermelerle % 100 verimliliğe ulaşılabilir ve malzeme kullanımı daha etkin hale getirilebilir. Ayrıca düzensiz şekillerin yerleştirilmesi üzerine benzer yaklaşımlar kullanılarak çalışmalar yapılabilir.

6. KAYNAKLAR

[1] Beasley, J. E. & Hoare, N. P.,."Placing boxes on shelves: a case study", *Journal ofthe Operational Research Society*, 52, 605–614,(2001).

[2] Gilmore, P. C.,& Gomory, R. E.,"Multistage cutting stock problems of two and more dimensions",*Operations Research*, 94–120, (1964).

[3] Gilmore P.C. & Gomory R.E.,"A linear programming approachto the cutting stock problem part II", *Operation Research*, 11, 863-888, (1963).

[4] Dowsland, K.A. ve Dowsland W.B.."Packing problems",*European Journal of Operational Research*, 2-14, (1992).

[5] Chen, C., Sarin, S. & Ram, B.,"The pallet packing problem for non-uniform box sizes",*International Journal of Production Research*, 29(10), 1963-1968, (1991).

[6] Lodi, A. & Monaci, M.,"Integer linear programming models for the 2-staged two dimensional knapsack problem",*Math. Prog.*, 94, 257–278, (2003).

[7] Holland, J.H.,*Adaption in Natural and Artificial Systems*, Ann Arbor: University of Michigan Pres,(1975).

[8] Hopper, E. and Turton, B.,"A Review of The Application of Meta-Heuristic Algorithms to 2D Strip Packing Problems", *Artificial Intelligence*, (2001).

[9] P versus NP problem, (17 December 2012), http://en.wikipedia.org/wiki/P_versus_NP_problem (2012).

[10] Leung, T.W., Yung, C.H. & Troutt, M.D.,"Applications of Genetic Search ad Simulated Annealing to The Two-Dimensional Non-Guillotine Cutting Stock Problem",*Computers and Industrial Engineering*, 40, 201-214, (2001).

[11] WascherG., Haubner H., Schumann H.,"An improved typology of cutting and packing problems", European *Journal of Operational Research*,183,(2006).

[12] P versus NP problem, (17 December 2012), http://en.wikipedia.org/wiki/P_versus_NP_problem (2012).

[13] Soke, A. ve Bingul, Z.,"Hybrid genetic algorithm and simulated annealing for two-dimensional non-guillotine rectangular packing problems",*Engineering Applications of Artificial Intelligence*, 19, 557–567, (2006).

[14] Dyckhoff, H.,"A typology of cutting and packing problems",*European Journal of Operational Research*, 44, 145–159, (1990).

[15] Gilmore, P. C.,& Gomory, R. E.,"Multistage cutting stock problems of two and more dimensions",*Operations Research*, 94–120, (1964).

[16] Gonçalves, J. F.,"A hybrid genetic algorithm-heuristic for a twodimensional orthogonal packing problem",*European Journal of Operational Research*, 183, 1212–1229, (2007).

[17] Gümüş, G.,"Dikdörtgen Şekilli Malzeme Kesme Problemleri İçin Genetik Algoritma Tabanlı Çok Ölçütlü Bir Çözüm Yaklaşımı", Yüksek Lisans Tezi, *Anadolu Üniversitesi Fen Bilimleri Enstitüsü*, Endüstri Mühendisliği Anabilimdalı, Eskişehir,(2012).

[18] Erdoğan, Y.,"İki Boyutlu Kesme Problemleri İçin Sezgisel Yaklaşım İle Bir Uygulama", Yüksek Lisans Tezi, *Bahçeşehir Üniversitesi Fen Bilimleri Enstitüsü*, Endüstri Mühendisliği Anabilimdalı, İstanbul, (2010).

[19] Gomez, A. ve Fuente, D.,"Resolution of strip-packing problems with genetic algorithms",*Journal of the Operational Research Society*, 51, 1289-1295, (2000).

[20] Jakobs, S,"On genetic algorithms for the packing of polygons",*European Journal of Operational Research*, 88, 165-181, (1996).

[21] Liu, D. and Teng, H.,"An improved BL-algorithm for genetic algorithm of the orthogonal packing of rectangles", *European Journal of Operational Research*, 112,413-420, (1999).

[22] Zachariadis, E. E., Tarantilis C.D. ve Kiranoudis C.T.,"A Guided Tabu Search for the Vehicle Routing Problem with two-dimensional loading constraints",*European Journal of Operational Research*, 195, 729–743, (2009).

[23] Adakçı, S.,"Stok Kesme Problemi ve Aliminyum Sektöründe Uygulaması", Yüksek Lisans Tezi, *İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü*, Endüstri Mühendisliği Anabilimdalı, İstanbul, (2010).

[24] Ergün, K.,"Kesme ve Paketleme Problemleri ve Araştırmaya Yönelik Bir Metot Geliştirilmesi ve Bu Metodun Etkinliğinin Sınanması", Yüksek Lisans Tezi, *Balıkesir Üniversitesi Fen Bilimleri Enstitüsü*, Balıkesir, (2004).

[25] Haessler, R. W., P. E. Sweeney,"Cutting Stock Problems and Solution Procedures",*European Journal of Operational Research*, 54, 141-150, (1991).

[26] Mori,H., "Unit Commitment Using Tabu search with Restricted and Neighborhood", *Proc. of ISAP'96*, 0221, 422-427, (1998).

[27] Pham, D.T. and Karaboğa, D.,"Intelligent Optimisation Techniques",London: Springer – Verlag, (2000)

[28] Söke, A.,"Genetik Algoritma Ve Benzetilmiş Tavlama ile İki Boyutlu Giyotinsiz Kesme Problemlerine Olasılıksal Yaklaşım", Yüksek Lisans Tezi ,*Kocaeli Üniversitesi Fen Bilimleri Enstitüsü*, Kocaeli, (2003)..

[29] Resende M.G.C.,"Riberio C.C., Greedy randomized adaptive search procedures",*International Series in Operations Research and Management Science*, 57, (2002)..

[30] Mitchell, M. & Forest S.,"Genetic Algorithms and Artificial Life", Vol. 1, Cambridge;MIT Press, 267-289, (1994).

[31] İşçi, Ö. ve Korukoğlu, S.,"Genetik Algoritma Yaklasımı ve Yöneylem Arastırmasında Bir Uygulama",*CBÜ İİBF Yönetim ve Ekonomi Dergisi*, 10(2), 191-208, (2003).

[32] Euro Special Interest Group on Cutting and Packing, (4 March 2012), http://www.paginas.fe.up.pt/~esicup (2012).

[33] Company History, (1 September 2012), http://www.lectra.com/en/about-lectra/company-history.php, (2012).