55686

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

PARÇA YERLEŞTİRME ALGORİTMALARININ PASTAL OLUŞTURMA PROBLEMİNE UYGULANMASI

YÜKSEK LİSANS TEZİ

Müh. Filiz BUNYAK

Tezin Enstitüye Verildiği Tarih : 27 Mayıs 1996

Tezin Savunulduğu Tarih : 20 Eylül 1996

Tez Danışmanı : Doç.Dr. Füsun TUNALI (SELÇUK)

Diğer Jüri Üyeleri : Doç.Dr. Nadia ERDOĞAN

Doç.Dr. Muhittin GÖKMEN

EYLÜL 1996

ÖNSÖZ

Beni bu tez konusunda çalışmaya yönelten, çalışmam boyunca da desteğini esirgemeyen danışmanım Doç. Dr. Füsun Selçuk'a; özellikle yazım aşamasında büyük yardımını gördüğüm, tezimi okuyup kontrol eden Araş.Gör. İlker Ersoy'a; programın grafik editör'ünü yazan Müh. Sevi Tüfekçi'ye; kaynak araştırmada yardımlarından ötürü Müh. Tolga Aşveren'e; değerli önerilerinden yararlandığım, Yük.Müh. Ersoy Pekşen ve Araş.Gör. Hasan Keçeci'ye; hayatım boyunca beni her konuda destekleyen Aileme,
teşekkür ederim.

EYLÜL 1996 Filiz BUNYAK

İÇİNDEKİLER

ŞEKİL LİSTESİ

ÖZET

Bu tezde, endüstride geniş uygulama alanları bulunan parça yerleştirme problemi ele alınmıştır. Özel olarak tekstil sektöründeki "Otomatik Pastal Hazırlama" işlemine uygulanması ele alınmıştır.

Problem iki ana kısımda incelenmiştir, dikdörtgen parçaların ya da çokgenlerin kapsayan dikdörtgenlerinin yerleştirilmesi ve doğrudan çokgen yerleştirme. Çalışmada algoritmik geometri yöntemleri ile sezgisel yaklaşımların kaynaştırılması yoluna gidilmiştir.

Birinci kısım, dikdörtgen yerleştirme, iki ana adımdan oluşur: Hangi parçanın yerleştirileceği kararı ve seçilen parça için uygun bir konum bulunması. Parçaların hangi sıra ile yerleşeceğine, genetik algoritmalar kullanılarak karar verilir. Verilen sıradaki parçaların yerleştirilmesi için ise sezgisel bir yaklaşım; alt-sol yaklaşımı kullanılır. Alt-sol sezgisel yaklaşımı kullanılarak, her parça sıralamasına tek bir yerleşim karşı düşürülür. Her sıralamaya yerleşim veriminin bir fonksiyonu olan uygunluk değeri atanır. Genetik operatörler kullanılarak bu uygunluk değeri, dolayısıyla da yerleşke verimi arttırılmaya çalışılır.

Çokgen yerleştirmede, çokgenlerin birbirleri ile örtüşmeden yerleştirilmeleri problemi, minkowki çokgen işlemleri kullanılarak çözülür. Yerleştirilen herbir çokgen için, daha önce yerleştirilmiş çokgenlerle örtüşmeyeceği alan bulunur. Bu alan içinde kalınarak, en alt, en sol konum ya da kapsayan dikdörtgeni minimum yapan konum saptanır, parça yerleştirilir, yapı güncellenir.

Tezin ikinci bölümünde yerleştirme problemlerinin genel tanıtımı ve sınıflandırılması yapılır. Üçüncü bölümde dikdörtgen parçaların yerleştirilmesi, dördüncü bölümde çokgen yerleştirilmesi incelenir.

SUMMARY

Part Nesting Algorithms for Nonconvex Polygons with Application to Automatic Marker Making

An essential step in the manufacture of clothing is the generation of cutting plan or marker. The marker determines how the parts that make up an article of clothing are cut from a bolt of cloth. For any marker making task, the set of parts is determined by the range of sizes and styles required for a particular cutting. The job of the marker maker is to pack the parts in a rectangle of smallest length whose width is determined by width of the bolt of cloth. Some parts may be rotated by180 degrees or flipped. Some parts may also may be rotated a small amount, no more than 3 degrees. Parts can not be rotated by arbitrary angles because cloth has a direction called nap. The pieces are cut out of layers of cloth on a long cutting table. To improve cloth utilization, the parts for many articles are included in the same marker. The efficiency of a marker is the ratio of the sum of all parts area to the total area. Cutting process itself can have severe impacts on the company's profit , especially when material of high value is involved. A poor way of cutting may result in a large amount of trim loss which means that material and production resources have been wasted. Unfortunately generating an optimal marker (shortest marker of a given width containing a given set of parts) is NP- complete. Using a CAD system, well trained people can generate near-optimal markers manually, but this is a time consuming job. Current software for automatic makers doesn't reach human performance so they are used with human intervention. Automatic generation of markers would better enable manufacturers to keep up with customer demands for different styles and sizes.

The problem of automatic marker making is a derivative of a problem known in the literature under different names such as : cutting stock, trim loss, bin packing, dual bin packing, strip packing, knapsack, loading, assortment, depletion, dividing, layout, nesting, partitioning or even capital budgeting, change making, assembly balancing, memory allocation, multiprocessor scheduling etc. These topics interest different disciplines such as Information and Computer Sciences, Management Science, Engineering Sciences, Mathematics, Operational Research. All of them have the common logical structure that there is on one hand stock of large objects and on the other hand a list of small items; and geometric combinations of small items are assigned to large objects.

The problem of automatic marker is the problem of two dimensional non-convex polygon nesting. The methods for the solution of nesting problems cover a wide range of methods. These can be classified in three main groups : analytical

methods, heuristic approaches, and others. Analytical methods are mostly used in the nesting of rectangular elements. They are too slow even for moderate sized problems because of the huge number of states that must be considered. For convex and non-convex part nesting AI approaches such as heuristic search, knowledge-based systems, case-based reasoning, neural networks etc. or improvement strategies based on ' meta-heuristics' such as simulated annealing, genetic algorithms, tabu search or other techniques like database/substitution, greedy heuristics, Monte Carlo placement, lattice packing, clustering methods are used.

This work covers two dimensional translational non-convex polygon nesting. As stated in the first paragraph rotation is not allowed because of the properties of the application: cloth has a direction called *nap*,no rotation other than up to 3 degrees are allowed but pieces can flip about x or y axis.

In this work the problem is treated in two main part:

1. Placement of rectangular pieces (2-D Bin packing)
2. Placement of nonconvex polygonal pieces (nesting)

In the first part the pieces to be nested are approximated to rectangles, by substitution to their bounding boxes and the problem is treated like 2D bin packing problem. 2D bin packing place some rectangular boxes (our bounding boxes) in an open ended bin (bolt of cloth) without overlapping with each other. As 2D rectangle packing is NP-complete in order to obtain near optimal solution in reasonable time an approach based on heuristic is taken and Bottom-Left Heuristic is chosen as placement strategy.

Bottom-Left Heuristic is a bottom-left stability preserving heuristic. If a rectangle is placed in a bottom-left stable position it can no more move downwards or slide to left. At any stage of the heuristic the bin contains a set of empty spaces (holes). There is always at least one hole which is the unbounded area over the placed boxes. To place a rectangle each hole is examined to find whether this rectangle fits in, some candidate places are found. The process of finding candidate places for a rectangle can be viewed as sliding a mechanical device composed from a pair of bar of length equal to the length of the rectangle, and a spring pulling them outwards. At any time the height of the bars is equal to the local maximum of the bottom subtracted from local minimum of the top, in the interval determined by the rectangle width. Whenever the height of the bars is equal or greater than the height of the rectangle to be placed, the position is marked as a candidate. After traversing each hole and extracting the candidates, the candidate that preserve bottom-left stability is taken. Then the rectangle is placed and data structure is updated to represent new configurations of the holes. For each rectangle to be packed same process is applied in the given order of the rectangles. For poorly ordered list of rectangles the algorithm can lead to bad packing but even a simple ordering of decreasing width applied to the parts to be packed leads to satisfactory results. At this point instead of using a simple ordering to improve the result, a job scheduling algorithm based on genetic algorithms is used whose output (order) is used to decide the order of rectangles to be packed in the placement stage.

Genetic Algorithms is first described as a methodology for studying natural adaptive systems and designing artificial adaptive systems in 1975 by J.Holland. It is now frequently used as an optimization method, based on analogy to the process of natural selection in biology where from one generation to the next weak elements are eliminated and those with the best performance in the current environment (fittest) survive. GAs are widely recognized as an effective search paradigm in job scheduling. GA maintains a population of strings (chromosomes) that encode candidate solutions to a problem. These strings are the analog of chromosomes in natural evolution. A fitness function defines the quality of the solution.

In this work GA is used to improve result of BL heuristic by finding efficient ordering of rectangles to be packed. Genotype is a string of integer coding the order of rectangles. Phenotype is the arrangement obtained after packing the rectangles following the order coded in the genotype. the process has five step:

1. Initialization
2. Reproduction
3. Evaluation
4. Selection
5. Termination

In the first stage a population is initialized randomly. Order based coding is used that is if N is the number of rectangle to be packed, genotype consists of an array of N integer and genotype is initialized by assigning N distinct number in the range 0-(N-1) to each element of the array. For a genotype g, i is the order of the rectangle no g[i]. A number of genotype is created to initiate the population. For each genotype correspond a fitness value, which is a function of the efficiency of the layout obtained using the order coded in the genotype.

Reproduction is made simulating roulette wheel selection. To each genotype from the population is assigned a portion in the wheel proportional to its fitness function. By means of this, good genotypes have more chance to be selected. A pairs of genotypes are selected and the crossover operator is applied to them. As a the crossover operator, order base crossover is used : A cut point is randomly found, part of the genotype from start to cut point is taken from one parent and directly copied to the child, remaining part of the parent is copied to child following the order in the second parent. Child constructed with this method inherit important information from both parent but mainly from first parent. Using the crossover operator stated above a pool of child is formed. At the evaluation stage for each child its phenotype is constructed decoding the genotype and a fitness value is calculated.

parent1 : 1 - 2 | 3 - 4 - 5
parent2 : 5 - 4 | 3 - 2 - 1
---
child : 1 - 2 - 5 - 4 - 3

An example of order based crossover

Selection process replace some individuals from previous population by the individuals from the child pool. Many threshold values have been tried to decide which individual to discard from previous population and which children to include. Change in the population does not always occur in a regular way in order to prevent to get stuck on a local minimum mutation operator is used. Two kind of mutation operator is defined for this application. First randomly select two integer in the genotype string and swap their place. Second operator again randomly select a rectangle and rotate it 90 degrees. Not to destroy regular conversion to a better population mutation operator is applied to the population with a low probability. Mutation operator change directly an individual in the population no selection process needed.

These processes are applied to an initial population iteratively. Termination condition is a predetermined number of iteration. If in a certain stage of the iteration population got hung up to a local minimum some part of the population is replaced with randomly generated individuals.

Genetic algorithms is used for its following properties:

1. GAs work with a coding of the parameter set, not the parameter themselves.
2. Gas search a population of points, not from a single point.
3. Gas use objective function information, not derivatives or other auxiliary knowledge.
4. Gas use probabilistic transition rules, not deterministic rules.

It has been seen from the solutions obtained that use of genetic algorithms to schedule rectangles to be packed with BL heuristic, improve considerably solution of BL placement.

Second part of the work: placement of small pieces covers a somewhat different area of research. The first part solve 2D bin packing problem. It deals mainly on the scheduling of rectangles, geometric properties of the pieces have not been taking into account. In the second part it is worked mainly on computational geometry. There are mainly three steps in the program:

- Determination of all non-overlapping positions of a polygon relative to all previously placed polygons.
- Selection of one of these positions, according to a selection criteria
- Placement of the polygon and rearrangement of the data structure

For the first step, " Minkowski Polygon Operations" is used. Outside of the resulting polygon from minkowski sum A⊕(-B), define all the positions, where the reference point of polygon B can be placed, without overlapping polygon A. Minkowski sum has been calculated by using an algorithm based on sweeping.

As selection criteria, two heuristic methods is used; Selection of the position that results in minimum bounding box of the placed polygons and selection of the bottom-left position. Minimum bounding box selection favors filling the holes, but sometimes results in dense but narrow arrangements that doesn't use all the width of the cloth, left place is too narrow to be filled and become waste. A combination of these two heuristics results in a better layout.

As rearrangement of data structure; each time a new polygon is added to the layout, polygon union of the previously placed polygons with the new polygon is calculated. At each step all previously placed polygons are threaten like a unique polygon, this diminish considerably calculations.

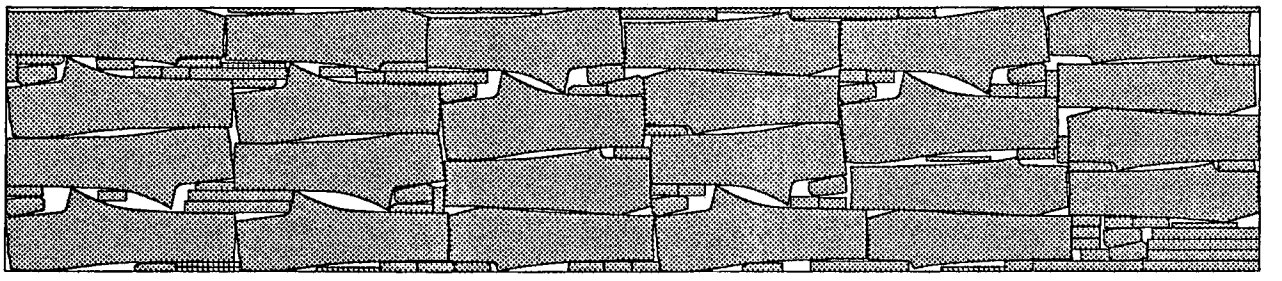
In brief this work combine computational geometry methods generally discarded by layout and packing communities, with their traditional optimization methods.

BÖLÜM 1

GİRİŞ

Otomatik Pastal Hazırlama

Konfeksiyon üretiminin temel adımlarından biri kesme planı ya da pastal hazırlanmasıdır. Pastal bir giysiyi oluşturan kalıpların kumaş topundan nasıl kesileceğini belirler. Kumaşın kullanım verimini arttırmak için, birçok giysinin kalıpları aynı pastala dahil edilir. Ortalama bir pastal 10-20m uzunluğunda 80cm.-1.5m eninde bir kumaş üzerine yerleştirilmiş 100-150 giysi kalıbından oluşur. Optimal pastal hazırlanması teorik olarak NP-karmaşıklıktadır (NP-complete). Deneyimli kişiler bir CAD sistemi kullanarak optimale yakın pastalları elle hazırlayabilirler, ancak bu hem zor hem de zaman alan bir iştir. Otomatik pastal hazırlanması hemen kullanıma sokulabilecek bir proje olduğundan oldukça ilgi çekici bir araştırma konusudur [1].



Şekil 1.1 Örnek Bir Pastal

Şekil 1.1 örnek bir pastalı gösterir. Dikdörtgenin kısa kenarı pastalın enini, uzun kenar pastalın boyunu belirtir. Pastalın enini kullanılan kumaşın eni belirler. Pastal hazırlama işinin amacı verilen parçaları (kalıpları) en kısa boylu dikdörtgenin içine yerleştirmektir. Bazı kalıplar 180° döndürülebilir veya aynalanabilir. Kumaş doku veya deseninin yönü olduğundan parçalar ara açılarda döndürülemez. Kalıp üzerinde kalıbın yerleştirilmesi gereken yönü belirten **"düz ipliği"** denen bir işaret (vektör) vardır ve bu vektörün yerleşim sonunda kumaştaki desen yada doku yönüne paralel olması gerekir. Bazı parçaların 2-3° yi geçmemek kaydıyla döndürülmesine izin verilebilir[1].

1.1 Problemin Teorik Yönü

Probleme giriş parametresi olarak, eni W, boyu L olan bir dikdörtgen ve bir poligon kümesi P verilir. Her poligon $p \in P$, köşelerinin koordinatları ve $\theta(p)$ dönebileceği maksimum açı (poligonun açısı $v(p)$ vektörü ile verilir) ve 180° dönüp dönemeyeceği, aynalanıp aynalanamayacağı ve tekstil sektörüne has birtakım bayraklar ile tanımlanır. Pastal hazırlama problemi bu poligon kümesinin verilen dikdörtgene birbirleri ile örtüşmeden, izin verilen dönüşümler kullanılarak yerleşip yerleşemeyeceği belirlemektir.

Pastal hazırlama NP-zorluktadır (NP-hard). Çünkü NP-tam (NP-complete) kutu paketleme problemi pastal hazırlamaya getirilebilir. Gary ve Johnson'un kutu paketleme tanımı alınırsa kutu paketleme problemi : Sonlu bir parça kümesi U, bir pozitif tamsayı büyüklük $s(u)$ her $u \in U$ için, bir kutu kapasitesi B ve başka bir pozitif sayı K olmak üzere. Problem U 'yu $U_1, U_2, \ldots, U_k$ şeklinde ayrık kümelere her bir kümedeki büyüklükler 's' lerin toplamı B'den küçük yada B'ye eşit olacak şekilde ayırıp ayırılmayacağıdır. Bu problem $u \in U$ şeklindeki herbir parçanın yerine eni 1 yüksekliği $s(u)$ olan dikdörtgenler koyarak pastal hazırlamaya çevirilebilir ve pastal hazırlayıcıdan bunları döndürmeden yüksekliği B, eni K olan bir dikdörtgene yerleştirmesi istenebilir. Açıktır ki eğer kutu-paketleme problemi çözülebilir ise pastal hazırlayıcı dikdörtgenleri K yığın halinde yerleştirebilir. Pastal varsa daha sonra bu soldan sağa gidilerek her dikdörtgeni olabildiğince sola, aşağıdan yukarı gidilerek her dikdörtgen olabildiğince aşağı kaydırılarak B ya da daha az yüksekliğe sahip, K ya da daha az sayıda yığın elde edilir. Böylece kutu paketleme problemine bir çözüm bulunur [1].

BÖLÜM 2

PARÇA YERLEŞTİRME PROBLEMİ

2.1 Problemin Tanıtımı

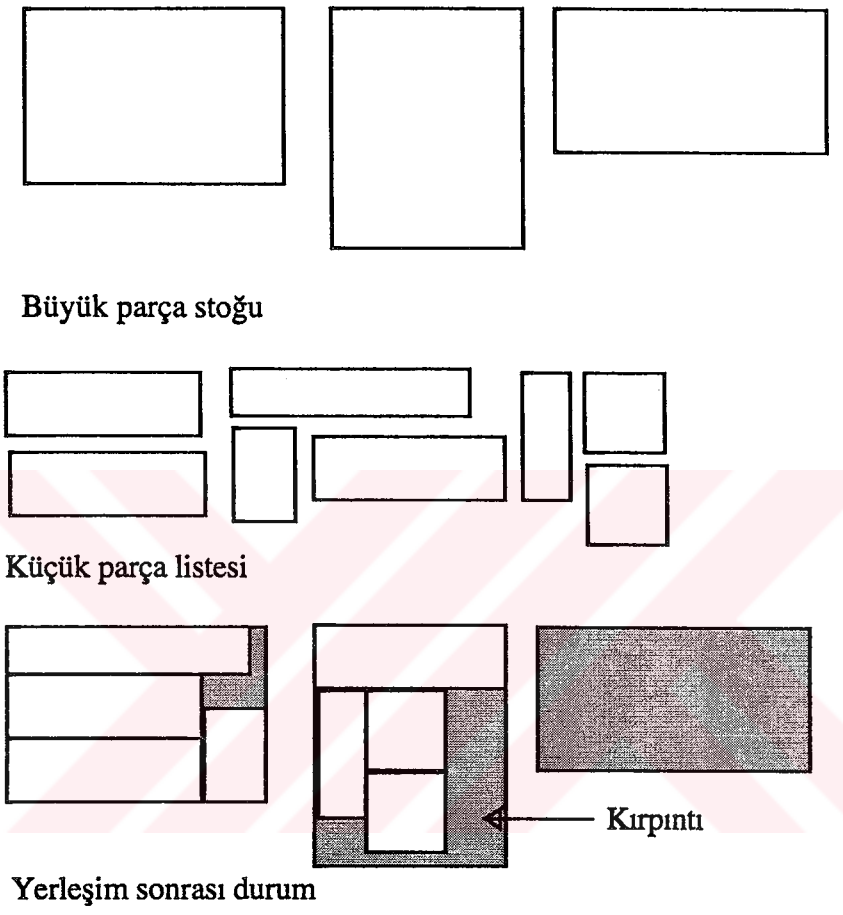
Parça yerleştirme problemleri literatürde değişik isimler altında yer alırlar. Örneğin:

- Stok kesme
- Kutu paketleme, şerit paketleme, vektör paketleme, sırt çantası problemleri
- Araç, palet, konteyner yükleme problemleri
- Tasnif (assortment), tasarım, bölme, serim, yerleştirme, bölmeleme problemleri
- Bütçeleme, para üstü verme, bellek ayırma, çok işlemcili sistemlerde iş sıralama problemleri

1990'da Dyckhoff [2] kesme ve paketleme problemleri tiplerinin bir sınıflandırmasını yaptığı çalışmasında *kesme ve paketleme* problemlerinin ortak yapılarını aşağıdaki şekilde özetler.

a) İki temel veri grubu vardır. Grupların elemanları bir yada daha fazla boyutlu, sabit şekilli geometrik nesnelerdir.
   - Büyük nesnelerin oluşturduğu stok
   - Küçük nesnelerin oluşturduğu siparişler listesi

b) Kesme veya paketleme problemleri küçük nesnelerin geometrik kombinasyonlarının oluşturduğu deseni büyük nesnelere atama işlemidir. Büyük nesnenin küçük nesnelerce örtülmeyen kısımları kırpma kaybı (kırpıntı) olarak değerlendirilir.



Şekil 2.1 Kesme ve paketlemenin temel yapısı

Paketleme ve yükleme problemleri dar anlamlarında, geniş boş nesnelerin (araçların, paletlerin, konteynerlerin, kutuların vb.) küçük nesnelerce doldurulmasıdır Bu büyük nesnelerin boş alanlarının küçük parçalara kesilmesi ve bu parçaların da bir kısmının küçük nesnelerce kaplanması diğerlerinin de kırpıntı olarak kalması ile aynı şeydir. Kesme ve paketleme problemleri arasındaki sıkı ilişki malzeme ve onun kapladığı uzay dualitesinden kaynaklanır.

2.2 Problem Hiyerarşisi

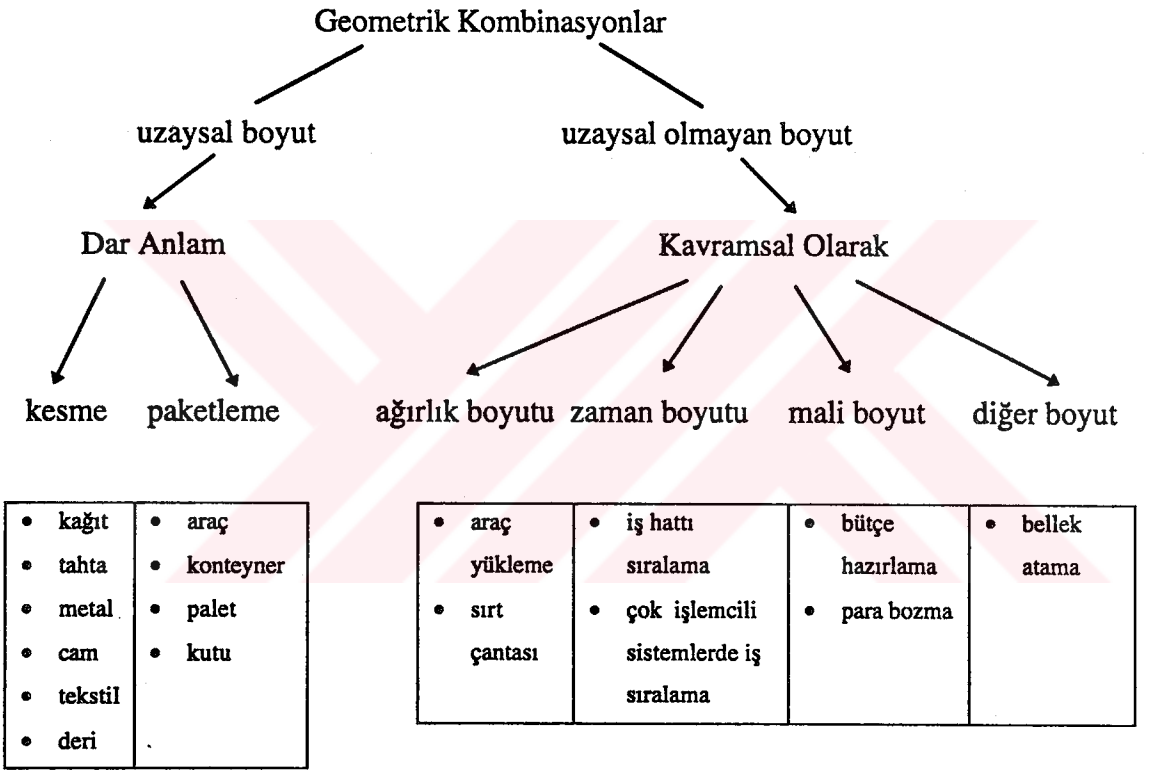
Karen Daniels yaptığı çalışmada kapsama, serim ve paketleme problemlerinin tanımlarını verirken hiyerarşik bir yapı kurar [3]. Buna göre: ***Kapsama problemi*** bir kap, parça kümesi ve bu parçalar üzerinde izin verilen bir küme dönüşüm kümesi verildiğinde, parçaların kap içinde birbirleri ile örtüşmeyen düzenlemelerini bulmak olarak tanımlanır. ***Paketleme problemi*** kapsama probleminin bir uzantısıdır. Sonucu kapsama gibi örtüşme olmayan bir düzenlemedir ancak buna ek olarak, kaba yada parçalara uygulanan bir kıstası optimize eder. Genellikle amaç, ya verilen parçaların sığacağı "minimal" kabı bulmak (minimal kıstası örneğin alan olabilir), yada verilen kabın içine sığacak "maksimal" parça kümesini bulmak olabilir. Buna göre paketleme bir optimizasyon problemi iken, kapsama verilen parçaların verilen kaba yerleşebilmesinin olurluluğunu, araştırır. Kapsama problemi, paketleme optimizasyon problemine ilişkin olurluluk problemidir.

Paketleme ve ***serim,*** bazen birbirleri yerine kullanılsalar da genelde serim ek kısıtları yada amaçları olan paketleme problemlerine karşı düşer. Örneğin VLSI seriminde iki amaç vardır: yonga alanını minimize etmek ve modüller arası bağlantıların uzunluklarını minimize etmek.

Yükleme terimi genelde üç boyutlu serim problemi için kullanılır. ***Yerleştirme*** terimi düzensiz şekillerin paketlemesine ilişkin, özellikle birbirinin içine yerleştirilebilen parçalar için kullanılır [3] .

2.3 Geniş Anlamında Kesme/Paketleme Problemi

Kuramsal olarak ele alındığında kesme ve paketleme uzaysal olmayan boyutta da gerçeklenebilir.



Şekil 2.2 Kesme-Paketleme problemlerinin uygulama alanları

2.4 Kesme, paketleme problemlerinin temel özellikleri ve tipleri

Dyckhoff [2] kesme ve paketleme problemlerinin temel karakteristikleri ve tiplerini aşağıdaki şekilde sıralar.

1) Boyut

2) Nicelik Ölçümü

- *Ayrık :* Belli şekildeki parçaların sayısı olarak
- *Sürekli :* Belli şekildeki parçaların toplam uzunluğu, ağırlığı vb. ilişkin boyuta oranla.

3) Parça Şekilleri

Parçaların geometrik gösterimleri. Konum gözardı edilirse parçalar,

- Biçim
- Büyüklük
- Yön

bilgileri kullanılarak tekil olarak tanımlanabilir.
Yerleştirme problemleri için önemli bir sorun şeklin biçiminin

- Düzgün
- Düzgün olmayan

formda olmasıdır.

Düzgün formlar dikdörtgen, daire gibi az sayıda parametre ile tanımlanabilen formlardır, düzgün olmayan formlar çokgenlerdir bunlar simetrik olmayan ve dışbükey olabilirler.

4) Tasnif

Yerleştirilmek istenilen parçaların hepsinin formlarının aynı olması yada farklı formlarda parçalara izin verilmesi durumu. Birinci durumda parçalar arası fark sadece büyüklük ve yönlerindedir. Parçaların hepsi benzer (congruent) hatta aynı olduklarında özel çözüm yöntemlerine yönelinir.

5) Elde edilebilirlik/Hazır bulunma:

- Parçaların sayılarının alt ve üst sınırı
- Parçaların ardışıllık ve sıraları
- Parçaların ne zaman paketlenebileceği yada paketlenmesi gerektiği

6) Parça Sırası

Parçaların bir sıra izlemesi yada zamanın rol oynaması, üretim hattı dengeleme, kızgın çelik çubukların kesilmesi yada yüklenenmesi gereken parçaların değişik teslim tarihleri olması gibi durumlarda gözlenebilir.

7) Desen kısıtları

- Küçük şekiller arasında yada büyük şekilleri bölen kesimler arasında minimum yada maksimum bir uzaklığın gerekmesi. Örneğin cam kırarken yada bir konteyneri yüklerken.
- Küçük şekillerin birbirine göre veya kapsayan şekle göre yönü, desenli parçalar kesildiğinde yada kırılgan parçalar yüklendiğinde önem kazanır.
- Bir desende küçük parçaların frekansı özellikle değişik parçaların kombinasyonu veya bazı parçaların başka parçalara göre sayısı konularındaki kısıtlamalardır.
- İzin verilen kesimlerin tipi ve sayısı. Dikdörtgensel yada blok parçalar kesiliyor ve giyotin kesim uygulanıyorsa, giyotin desenlerinin karmaşıklığı kesim yönünün değişmesine ve herbir etaptaki paralel kesime bağlıdır.

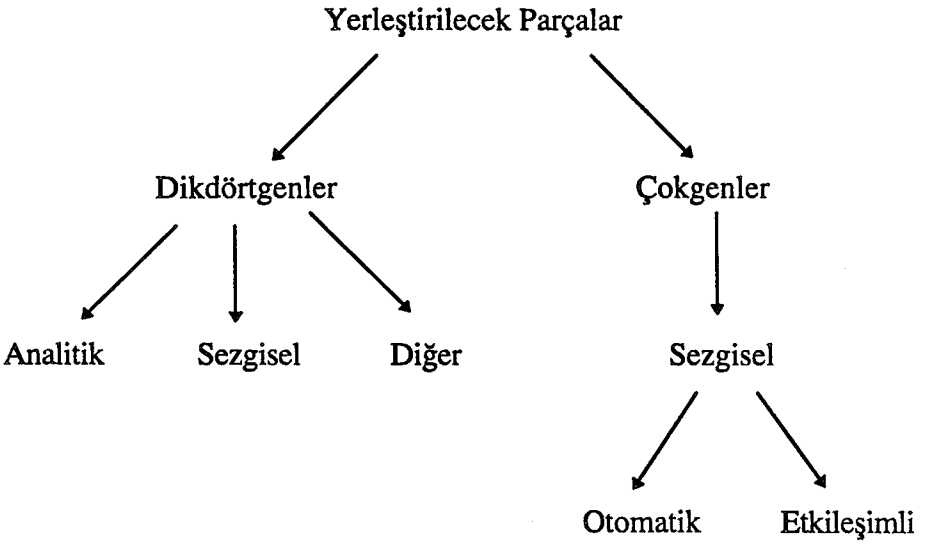
Daniels, Dyckhoff ve Finke'nin sınıflandırmasına yer verir [3]. Dyckhoff ve Finke literatürde parçaların konteynerlere atanması ve parçaların tasnifine göre dört temel paketleme-kesme tipi olduğunu ve aynı tipteki problemlerin çözümü için aynı tekniklere yönelindiğini söyler.

1. **Kutu paketleme:**Heterojen parçalar seçme kaplara atanır.
2. **Stok Kesme:** Kutu paketlemeye benzer ancak kesilmek istenen parçalar şekillerine göre denklik sınıflarına bölmelenebilir.
3. **Sırt Çantası:** Bütün kaplar kullanılmalıdır ve parçalar değişiktir.
4. **Palet Yükleme:** Bütün kaplar kullanılmalıdır ama parçalar aynıdır.

Zorluğu nedeniyle kutu paketleme problemleri sezgisel yöntemler kullanılarak çözülür. Stok kesmede farklı parça sayısı az olduğundan desen tekrarlayarak çözüm bulunabilir. Sırt çantası problemleri genellikle branch and bound veya dinamik programlama yöntemleri ile çözülür. Palet yüklemede konteynerler çoğunlukla aynı tiptedir, böylece bir konteyner için bulunan çözüm diğerlerine de uygulanabilir.

2.5 Yerleştirme Probleminin Varolan Çözüm Yaklaşımları

Parça yerleştirme problemi geniş bir çözüm yöntemi yelpazesine sahiptir. Problemin tamamını yada bir kısmını çözmek için bazı durumlarda matematiksel programlamanın kesin teknikleri kullanılır (lineer veya dinamik programlama gibi). Bazı araştırmacılar sezgisel arama, bilgi tabanlı (knowledge-based), durum tabanlı sorgulama ve yapay sinir ağları gibi yapay zeka yöntemleri kullanır. Bazıları tavlama benzetimi (simulated annealing), genetik algoritmalar, tabu arama yöntemi gibi meta heuristic iyileştirme yöntemleri kullanır. Ayrıca veritabanında yerine koyma (database substitution), hırslı sezgisel (greedy heuristic), monte carlo yerleştirme, kafes(lattice) paketleme, kümeleme (clustering) yöntemleri de kullanılmıştır [3].



Şekil 2.3 Parça Yerleştirme Probleminde Kullanılan Çözüm Yaklaşımları

Parça yerleştirmede kullanılan çözüm yaklaşımlarının parça tiplerine göre dağılımı şekil 2.3 de ki gibi özetlenebilir.

BÖLÜM 3

KUTU PAKETLEME PROBLEMİ

Sonlu bir dikdörtgensel kutular kümesi $E=\{e_1,e_2....e_n\}$ verildiğinde $W=\{(x_1,y_1)....(x_n,y_n)\}$ kutuların boyutları olmak üzere $0 \leq x_i,y_i \leq L$. Kutu paketleme, E'deki tüm kutuları ya da bazılarını, birbirleriyle ve yerleştirildikleri kabın çeperleri ile örtüşmeden $X > L$, $Y > L$ boyutlarındaki kaplara, kutuların toplam alanının kap alanına oranı maksimum olacak şekilde yerleştirmedir.

3.1 Pastal Oluşturmada Kutu Paketleme Kullanımının Sebepleri

1) *Malzeme ihtiyacını belirleme,* pastalda kullanılacak parçalar için ne kadar malzeme gerekeceğinin kabaca hesabı için kullanılır.

2) *Etkileşimli çalışma öncesi kabaca yerleştirme,* parçalar kullanıcı tarafından elle yerleştirilecekse, kullanıcıya kolaylık sağlamak için parçaların kapsayan dikdörtgenleri esas alınarak kabaca ama hızlı yerleştirme yapılır.

3) *Kümeleme* yönteminde, uygun kümeler halinde birleştirilmiş poligonların, kapsayan dikdörtgenlerinin oluşturduğu kutuları paketlemede kullanılır.

4) Doğrudan poligon yerleştirme algoritmasına sıralama bilgisinin aktarılmasında kullanılır.

3.2 Dikdörtgen Parça Yerleştirme ile İlgili Çalışmalar

Gilmore ve Gomory'nin (1961, 1965, 1967) parça yerleştirme, kesme, paketleme üzerine çalışmaları konu üzerine temel çalışmalardandır. Yöntemleri esas olarak doğrusal programlama modeli çerçevesinde simpleks algoritmasına dayanmaktadır. Burada doğrusal karar modelinin amaç fonksiyonunda beklenen iyileştirmeyi sağlayacak yeni bir kolon, yardımcı problem çözülerek oluşturulur. Çözülecek yardımcı problem sırt çantası veya yükleme problemi tipinde tamsayılı bir doğrusal programlama modelidir. Problem çok fazla değişken içerdiğinden, hesap yükü fazladır. Bu yükü kısmen azaltmak için Gilmore ve Gomory, probleme giyotin kesim kısıtlaması getirir [4]. (Giyotin kesim, malzemeyi bir uçtan diğer uca kesip ikiye bölen kesimdir.)

[5]'de Isranı ve Sanders dikdörtgen parça yerleştirilimindeki sezgisel yöntemleri inceler ve karşılaştırır. [2]'de Dyckhoff kesme ve paketleme problemlerini tiplere ayırır, özelliklerini tanıtır, çözüm yaklaşımlarını inceler. [6]'da K. Dowsland ve W. Dowsland paketleme ve paketleme türevi problemleri ve ilgili çalışmaları inceler.

[7]'de Beasley, giyotin kesim için iki boyutlu kesim desenleri oluşturmada hırslı arama (greedy) yöntemini kullanan; kullanılacak kesim desenini seçmek için ise doğrusal programlama kullanan bir sezgisel yöntem önerir. [8]'de Wascher "Çok Kriterli Karar Verme" (Multiple Criteria Decision Making) tekniği olarak bilinen yöntemin stok kesme problemlerinin çözümünde kullanılabileceğini belirtir. [9]'da Kural-Tabanlı yeni bir sezgisel yaklaşım önerilir. [10]'da Farley, Gilmore ve Gomory'nin üç adımlı algoritmasını tamsayı programlama ile birleştiren bir sezgisel yöntem açıklar. [11]'de Albano ve Orsini sezsisel bir yöntem önerir.

Tezde, parça yerleştirme problemindeki sezgisel çözüm yöntemlerinin incelenmesine ağırlık verilmiştir. Özellikle Bengtsson[12] ve Chazelle[17] incelenmiş; gerçekleme sırasında da Chazelle'in "alt-sol" sezgisel yaklaşımına [17] dayanan bir yöntemin genetik algoritmalar ile başarımının arttırılması üzerine çalışılmıştır. Bölüm 3'ün devamında bu yaklaşımlar ve gerçekleme özellikleri tanıtılacaktır.

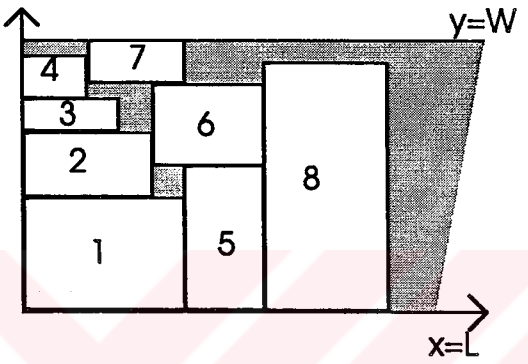
3.2.1 Bength-Erik Bengthsson Sezgisel Yaklaşımı

[12]'de Bengtsson bir ucu açık dikdörtgene parça yerleştirmek için aşağıdaki sezgisel yaklaşım önerilir. Buna göre mümkün yerleşimlerden parçaların çıkıntı oluşturmadığı yığınlar seçilir. Şekil 3.1'de 1,2,3,4 numaralı şekillerin yatay genişlikleri yerleşme sırasıyla azaldığı için çıkıntı yada sarkma oluşturmazlar. 5,6,7 numaralı parçaların da eklenmesinden sonra, bunların yatay genişlikleri artan sırada olmasına rağmen yığında çıkıntı oluşmaz. Parçalar soldan sağa bölümler halinde yerleştirilir.

3.2.1.1 Tanımlar

Bölüm: Yatay olarak tarandığında, tarama satırı en fazla bir parçayı kesen sıralı parçalar kümesidir. Verilen algoritma ile oluşturulan kesitlerde, ardışıl parçalar seyrek durumlar dışında bir yatay kenar boyunca birbirlerine değerler.

Yığın: Her bir parçanın, ya y-eksenine, ya da en azından önceki bölümlerden birine ait parçalardan birinin dikey kenarına değdiği bölümlerin sıralı kümesidir. (Değilen parçanın hemen bir önceki bölüme ait olması gerekmez.)
Örneğin (1-2-3-4), (5-6-7) ve (8), (*1-2-3-4-5-6-7-8*) yığınını oluşturan üç bölümdür.



Şekil 3.1 8 parçanın olası bir yerleşimi

k numaralı parçanın sağ üst köşesinin koordinatları (x_k,y_k) olsun. Bir bölüm, x_k değerleri azalan bir sıra izlerse *iyi kurulmuş* bir bölümdür. Tüm bölümleri iyi kurulmuş yığın *iyi kurulmuş* bir yığındır.

Şekil 3.1'deki yığın iyi kurulmuştur çünkü $x_1 \geq x_2 \geq x_3 \geq x_4$ ve $x_5 \geq x_6 \geq x_7$ şartları sağlanır, üçüncü bölümde de sadece bir parça vardır.

3.2.1.2 Bengthsson Yerleştirme Algoritması

```
procedure pack;
if good-hope then
begin
        piece:=first-piece-in-size;
        while piece ≠nil do
        begin
                if piece is free then
                begin
                        add a section with piece as base;
                        pack;
                        check for minimum;
                        delete last section
                end;
                piece=next-piece-in-size
        end;
end;
```

good-hope fonksiyonu basit bir tahmin yapar. Örneğin kalan parçaları hiç kayıpsız yerleştirme imkanı olsa bulunan sonucun o ana kadar bulunmuş en iyi sonuçtan daha iyi olup olamayacağını döndürür. Olabilirse *good-hope* true değerini geri döndürür.

First-piece-in-size, kenarlarından biri en uzun kenar olan parçayı verir.

next-piece-in-size fonksiyonu ise azalan kenar sırası listesindeki bir sonraki parçayı verir. Her parça listede iki kez yer aldığından parçaların iki yönü de denenmiş olur.

Bir bölüm oluşturmak için seçili parça taban olarak alınır, yerleştirilmesi mümkün en büyük uzunluğa sahip parçalar bölüme eklenir. Şekil 3.1'de 6 numaralı parça hala yerleştirilmemiş parçalar arasında en $\leq (x_5 - x_2)$ ve boy $\leq (W - y_5)$ şartlarını sağlayan en uzun kenarlı parça olarak seçilmiştir. Genelde eni geniş parçaların önce yerleştirilmesi (birinci şart) istense de bölümün en üst parçası için ikinci şart (tavana yaklaşma) daha önemlidir.

Rekürsif çağrıdan çıkmak için :

1. Kısmi sonuçta kabul edilemez kayıp olması,
2. Yerleştirilecek parça kalmaması,
3. Tüm kombinasyonların hesaplanmış olması,

durumlarından birinin olması gerekir. ***check for minimum*** tüm parçaların yerleştirilip yerleştirilmediği testini de içerir.

3.3 Genetik Algoritmaların Kutu Paketleme Probleminde Kullanılması

3.3.1 Genetik Algoritmalarının Tanıtımı

Genetik algoritmalar, evrimsel genetik ve Darwin'in doğal seçimine benzerlik kurularak geliştirilmiş iteratif bir arama yöntemidir. Arama, kullanıcı tarafından tanımlanan bir fonksiyonu (uygunluk fonksiyonu) optimize etmeye çalışır. Bu işlemi gerçeklemek için GA tüm arama uzayında "bireyler" olarak adlandırılan bir aday noktalar "popülasyonu" kullanır. Her iterasyonda, genellikle, uygunluk fonksiyonunca tanımlanan çözüme daha önceki nesildeki bireylere kıyasla daha uygun bireylerden oluşan bir nesil oluşur. İterasyonlar arttıkça, bireyler uygunluk fonksiyonunun optimumuna doğru kayacaktır. Yeni bir nesil oluşturmak için GA üç etaptan geçer:

1. Eski nesildeki her bir bireyin uygunluk değerini hesaplama

2. Bireyleri uygunluk değerlerini göz önüne alarak seçme

3. Seçilen bireyleri genetik operatörler, çarprazlama(crossover), mutasyon gibi, kullanarak uyuşturma. Algoritmik bakış açısından bu işlem mevcut çözümleri yerel olarak değiştirip birleştirmek olarak görülebilir.

GA'yı çekici kılan, başlangıçta bilinmeyen bir arama uzayından topladığı bilgileri yığıp daha sonraki aramaları alt arama uzaylarına yönlendirmek için kullanmasıdır. Bu aramadaki mekanizma daha önceki denemelerde bulunmuş yüksek başarımlı "yapı taşlarını" birleştirmektir.

Üç önemli özellik GA'yı diğer yaklaşımlardan ayırır:

1. GA bir arama noktası değil, bir grup arama noktası (sonuç adayı) üzerinde çalışır. Yani arama uzayında yerel global arama yapar.

2. GA aranan ortamdan sadece bireylerin uygunluk değerlerini bulmak için bir amaç fonksiyonu ister, başka bilgi yada kabule gerek yoktur. iferansiyel alabilme, türev gibi)

3. Seçme ve birleştirme etaplarında deterministik kurallar değil olasılık kuralları kullanılır.

Klasik GA'da dördüncü bir fark daha sayılabilir; GA optimize edilecek parametreleri kodlar ve bu kodlar üzerinde işlem yapar. (doğrudan parametreler üzerinde değil genotype-phenotype farkı). Parametrelerin kodlanması ilk optimizasyon problemini kombinezonsal bir bir probleme çevirmektir çünkü GA kombinezonsal arama için bir mekanizmadır [14].

3.3.2 Genetik Algoritmaların Matematiksel Temeli

Genetik algoritmaların aramayı nasıl daha yüksek uygunluk değerine sahip bölgeye kaydırdığını anlamak için şema notasyonu incelenmelidir. Goldberg, 'Genetic Algorithms in Search, Optimization, and Machine Learning'[13] adlı kitabında şema kavramını ve genetik algoritmalarının temel teorisini anlatır. Şema, arama uzayında bir küme bireydir. Çoğu GA'da bireyler sabit uzunluklu ikili düzendeki bir katar ile temsil edilir. İkili düzen kullanıldığında bir şema {0, 1, *} alfabesi üzerinde tanımlanmış bir desendir. Desende 0'lar ve 1'ler tanımlayıcı bitler olarak anılır, ' * ' bulunduğu basamağın 0 veya 1 olabileceğini belirtir . Tanımlayıcı bitlerin sayısı şemanın *mertebesini*

(order) o(H) verir. En sol ve en sağ tanımlayıcı bitlerin arasındaki uzaklık şemanın *tanımlayan uzunluğunu* (defining length)δ(H) verir. Örneğin **0*11**1 şemasının mertebesi 4, tanımlayan uzunluğu 6 dır. Şema, genetik operatörlerin popülasyonda bulunan yapı taşları üzerindeki etkilerini analiz etmek için temel yöntemler sunar.

3.3.2.1 Çoğalmanın Şema Üzerindeki Etkileri

Verilen bir *t* zamanında A(t) populasyonu; m(H,t) ise popülasyonda H şemasına ait örneklerin sayısını göstersin. Çoğalma sırasında bir A_i katarı $p_i = f_i / \sum f_j$ olasılığı ile seçilir ("f " uygunluk kodu olmak üzere). t+1 adımında A(t+1) popülasyonunda H şemasının m(H, t+1) kopyası bulunması beklenir.

$$m(H,t+1) = m(H,t)\ n\ f(H) / \Sigma f_j \qquad (3.1)$$

f(H) t. adımda H şemasına ait katarların uygunluk değerlerinin ortalaması olmak üzere.Tüm popülasyonun ortalama uygunluk değeri $\bar{f} = \Sigma f_j / n$ olarak yazılırsa aşağıda ki formül elde edilir.

$$m(H,t+1) = m(H,t)\frac{f(H)}{\bar{f}} \qquad (3.2)$$

Bu formülden de görülebilir ki ortalama uygunluk değerinin üzerinde bir uygunluk değerine sahip şemalar bir sonraki popülasyona daha fazla temsilci aktaracak iken diğerlerinin temsilci sayısı azalacaktır. Önemli bir nokta da bu aktarım olayının belli bir A popülasyonunda bulunan her H şeması için paralel olarak geliştiğidir.

Belli bir H şemasının uygunluk değerinin, ortalama uygunluk değerinden c bir sabit olmak üzere, $c\bar{f}$ kadar fazla oldu varsayılsın.

$$m(H,t+1) = m(H,t)\frac{(\bar{f}+c\bar{f})}{\bar{f}} = (1+c).m(H,t) \quad \textbf{(3.3)}$$

Böylece t = 0'dan başlanırsa sabit bir c değeri için geometrik ilerleme fonksiyonu elde edilir.

$$m(H,t) = m(H,0).(1+c)^t \quad \textbf{(3.4)}$$

3.3.2.2 Çaprazlamanın Şema Üzerindeki Etkileri

Çaprazlama, katarlar arasında yapısal ama aynı zamanda da rastlansal bir bilgi değiştokuşudur. Çaprazlama da rastgele iki katar ve yine rastgele bir kesim noktası seçerek, katarların kesim noktalarından bölünmesi ile oluşan alt katarları çaprazlar, yani ilk katarda yerlerini değiştirir. Çaprazlamanın şemaya etkisi bir alttaki örnekte görülebilir:

A = 0 1 | 1 1 | 0 0 0
H_1 = * 1 * | * * * 0
H_2 = * * * | 1 0 * *

A, hem H_1 hem de H_2 şemalarına uyan bir katar, "|" kesim noktasıdır. Çaprazlama işlemi sonucunda H_1 bozulurken H_2 işlemden etkilenmez. Açıktır ki H_1'in bozulma olasılığı H_2'ye göre daha fazladır, çünkü kesim noktasının iki uç arasında kalma olasılığı daha fazladır. Bir şemanın çaprazlamayı bozulmadan atlatabilmesi için, kesim noktasının tanımlayan uzunluğun dışında kalması gerekir. l katar uzunluğu, δ şemanın tanımlayan uzunluğu olmak üzere, bir şemanın çaprazlamayı bozulmadan atlatabilme olasılığı **(3.5)** ile ifade edilir:

$$p_s = 1 - \delta(H)/(l-1) \quad \textbf{(3.5)}$$

çarprazlanma olasılığı p_c olursa p_s'in alt sınırı **(3.6)** ifadesi ile verilir.

$$p_s \geq 1 - p_c \cdot \frac{\delta(H)}{l-1} \tag{3.6}$$

Çaprazlama ve çoğalma operatörlerinin birbirinden bağımsız olduğu kabul edilirse şemanın, bir sonraki popülasyonda beklenenen temsilci sayısı:

$$m(H,t+1) \geq m(H,t).\frac{f(H)}{\bar{f}}\left[1 - p_c \cdot \frac{\delta(H)}{l-1}\right] \tag{3.7}$$

Hem çoğalma hem çaprazlama operatörleri uygulandığında m(H,t+1) iki şeye dayanr: Şemanın uygunluk değerinin popülasyon ortalamasından yüksek yada alçak olması ve şemanın kısa yada uzun tanımlayan uzunluğa sahip olması

3.3.2.3 Mutasyonun Şema Üzerindeki Etkileri

Mutasyon bir basamağın p_m olasılıkla değişmesi. Bir şemanın mutasyonu bozulmadan atlatabilmesi için tüm allele'lerinin (genlerin alabileceği sayılar) korunması gerekir. Herbir allele'in kurtulma olasılığı 1- p_m olduğuna o(H) mertebesindeki bir şemanın kurtulma olasılığı $(1- p_m)^{o(H)}$ dir. p_m 'in çok küçük değerleri için($p_m << 1$) şemanın kurtulma olasılığı 1-o(H) p_m olarak ifade edilebilir.

Sonuç olarak çoğullama, çarprazlama, mutasyon etkileri toplandığında **Şema Teoremi**'ne diğer adıyla **Genetik Algoritmaları Teoremi'**ne ulaşılır.

$$m(H,t+1) \geq m(H,t).\frac{f(H)}{\bar{f}}\left[1 - p_c \cdot \frac{\delta(H)}{l-1} - o(H)p_m\right] \tag{3.8}$$

Sonuç olarak kısa, düşük mertebeli, yüksek uygunluk değerine sahip şemalar örneklenir, birleştirilir,potansiyel olarak daha yüksek uygunluk değerine sahip katarlar

oluşturmak üzere yeniden örneklenir. Bu tipteki şemalara *yapı taşı* denir. Her akla yakın kombinasyonu deneyerek yüksek başarımlı katarlar yaratmaya çalışmak yerine, geçmiş örneklemelerin en iyi kısmi sonuçlarından gitgide daha iyi sonuçlar elde edilebilir. Genetik algoritmalar optimale yakın sonucu böyle kısa, düşük mertebeli, yüksek başarımlı yapı taşlarını birleştirerek bulur.

3.3.3 Kutu Paketlemede Mevcut GA Yaklaşımları

Goodman [15]'de GA nın kutu paketleme problemine uygulanmasının kapsamlı bir incelemesini yapar, mevcut çözümleri anlatır yeni bir yaklaşım sunar.

Bir yaklaşım, kutu paketlemeyi, bir *kutu listesi* ve listeyi paketlemeye dönüştüren bir *kod çözücü algoritmaya* dayanır. Listeye değişik mutasyon ve çaprazlama işlemleri uygulanabilir.

3.3.3.1 Kod Çözme Algoritmaları

Kod çözme algoritması, herhangi bir kutu listesinden, geçerli bir paketleme deseni oluşturur. Geçerli desen, kutuların birbirleri ile örtüşmedikleri, kap çeperini de aşmadıkları desendir.

PACK (Symbolic Layout IDE) Kod çözme algoritması listeden sırayla kutuları alır, Kabın üst köşelerinden birine yerleştirir. Kutuyu en uzak köşeye atar gibi yönlendirir. Kutu yerçekiminin etkisindeymiş gibi dikey olarak hareket eder. Kutular kararlı bir konuma gelene kadar zigzag yapar. SLIDE PACK hızlı, hesaplaması kolay bir algoritmadır. Zaman

karmaşıklığı $O(n^2)$ dir. Listedeki n adet kutu (n!) farklı sırada bulunup, (n!) farklı kod oluşturabilir.

SKYLINE PACK (Ufuk çizgisi) kod çözme algoritması listedeki herbir kutu için kısmen yerleştirilmiş kapta tüm kararlı konumlar denenir. *Kararlı konum* kutunun bir köşeye ya da daha önce yerleştirilmiş kutuların oluşturduğu bir çukura yerleştiği konumdur. Algoritma, adını daha önce yerleşmiş kutuların oluşturduğu çizginin üzerinde dolaşmasından alır. Algoritmanın zaman karmaşıklığı $O(n^4)$ dir.

3.3.3.2 Kutu Listesine Uygulanabilecek Genetik İşlemler

[15]'de kutu listesine uygulanan çaprazlama, mutasyon, değerlendirme, seçme işlemleri çeşitleri incelenir. Uygulanan çaprazlama işlemi genellikle sıralama tabanlıdır. Seçilen ebeveynlerden birincisindeki sıra, kesim noktasına kadar, aynen çocuğa aktarılır geri kalan genler ikinci ebeveyndeki sıra ile çocuğa kopyalanır. Sıralama tabanlı Smith çaprazlam yöntemi aşağıdaki şekilde uygulanır.

katar1 : 1-2 | 3-4-5
katar2 : 5-4 | 3-2-1

çocuk : 1-2 | 5-4-3

Mutasyon için ise Smith yine iki yöntem önerir. *KARIŞTIR* listenin bir kısmını rastgele sıralar. *DÖNDÜR* rastgele seçtiği kutuları 90° döndürerek yeni bireyler oluşturur.

Değerlendirme (uygunluk fonksiyonunun hesabı) için yaygın olarak, paketlenen kutuların toplam alanının kap alanına oranı kullanılır. Kısmi paketlemeyi değerlendiren, yani henüz tüm parçaların paketlenmesi bitmediği durumda ki başarımı değerlendirip bu sonuca göre paketlemeye devam edip etmeme kararı alan veya kutuların ara boşluklara yerleştirilmesini desteklemek için bazı kıstaslar koyan yöntemlerin varlığı incelemede belirtilir ve yeni bir kodlama yaklaşımı HCR (hiyerarşik kromozom gösterimi) tanıtılır.

3.3.3.3 Hiyerarşik Kromozom Gösterimi (HCR)

HCR, yapı taşlarını (şema) koruyacak şekilde tasarlanmış, klasik liste yaklaşımından farklı bir koddur. Kodda genler elemanları değil, elemanların oluşturduğu grupları temsil eder. Anlamlı yapı taşları, eleman grupları ve bunların grup içi yerleşim düzenleridir. GA'nın temelindeki fikir zaten arama uzayını, umut verici bölgeleri bulmak için taramak ve böyle elde edilen bilgileri bu bölgelerdeki aramada kullanmaktır. Kodlama, yapı taşlarının ebeveynlerden yeni nesile aktarımını sağlamıyorsa sonuç rastgele aramadan veya basit evrimden sadece biraz daha iyi olur, HCR bu yapı taşlarını korumayı hedefler [15].

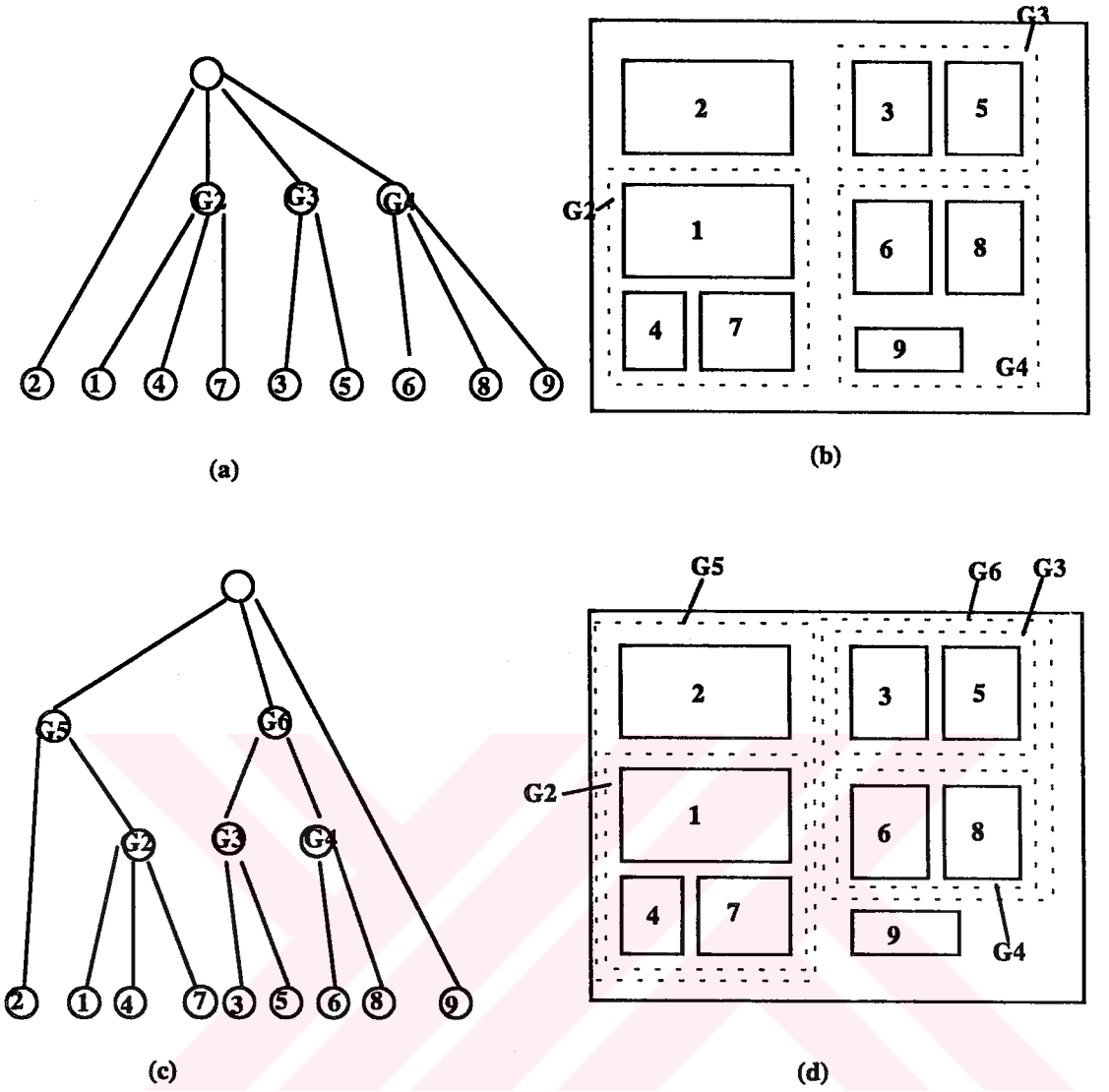
HCR'de 2-seviyeli bir gösterimde, herbir kromozom G1,G2...Gm gibi eleman gruplarından ve elemanların bağıl (grup içi) koordinatlarından oluşur. Örneğin Şekil 3.4 b' de gösterilen yerleşimin kromozomu Şekil 3.3 deki gibidir.

```
<
{ [2] } (xG1,yG1)                                  : grup G1, 1 eleman, seviye 0
{ [1(x1,y1)], [4(x4,y4)], [7(x7,y7)] } (xG2,yG2),  : grup G2, 3 eleman, seviye 1

{ [3(x3,y3)], [5(x5,y5)] } (xG3,yG3),              : grup G3, 2 eleman, seviye 1
{ [6(x6,y6)], [8(x8,y8)], [9(x9,y9)] } (xG4,yG4)   : grup G4, 3 eleman, seviye1
> (0,0).
```

Şekil 3.2 2-Seviyeli HCR örneği

```
<
{                                                  :G5 grubunun başı,
seviye 2
        { [2] } (xG1,yG1)                          : grup G1, 1 eleman,
seviye 0
        { [1(x1,y1)], [4(x4,y4)], [7(x7,y7)] } (xG2,yG2),   : grup G2, 3 eleman, seviye 1
}(xG5,yG5),                                        : grup G5 in sonu


{                                                  :G6 grubunun başı,
seviye 2
        { [3(x3,y3)], [5(x5,y5)] } (xG3,yG3),      : grup G3, 2 eleman, seviye 1
        { [6(x6,y6)], [8(x8,y8)] } (xG4,yG4)       : grup G4, 2 eleman, seviye 1
}(xG6,yG6),                                        :G6 grubunun sonu

{ [9] } (xG7,yG7)                                  : G7 , 1 eleman
seviye 0
> (0,0).
```

Şekil 3.3 3 Seviyeli HCR örneği



Şekil 3.4 a-b Şekil 3.2' deki HCR'nin ağaç ve yerleşimi
c-d Şekil 3.3' deki HCR'nin ağaç ve yerleşimi

HCR'de Kod Çözme, Her adımında 2 yada daha fazla grup kaynaştırılarak tek grup haline getirilirek yapılır. En son adımda tüm gruplar tek bir grup halinde birleşmiş olur.

HCR'de çaprazlama işlemi için aşağıdaki adımlar izlenir:

1. İlk ebeveynden rastgele bazı gruplar seçilir Gf
2. İkinci ebeveynden Gf deki elemanları içermeyen bazı gruplar Gm seçilir
3. Gf ve Gm de yer almayan grup ve elemanlardan sezgisel yaklaşımlar kullanarak, yeni gruplar oluşturulur.

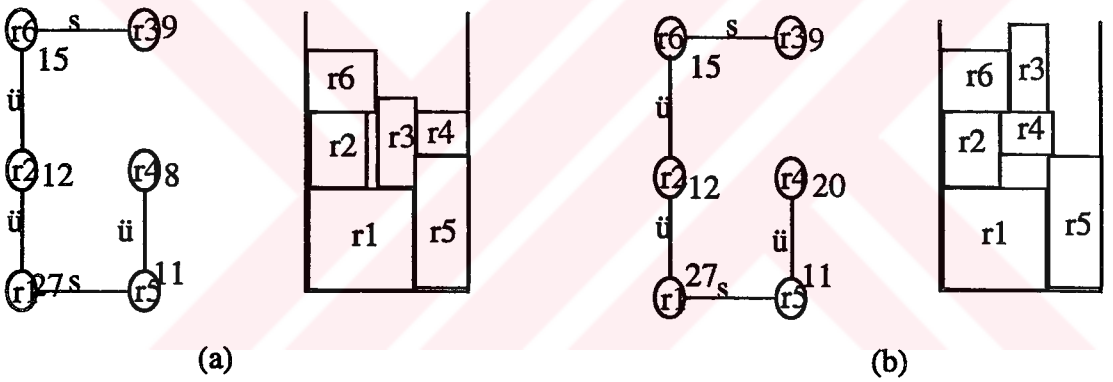
1 ve 2. aşamalarda alınan grup sayısı değiştirilerek GA ve sezgisel yaklaşım dengesi düzenlenir. 3. aşamada yeni gruplar oluşturulurken önceki serimlerden elde edilen bilgiler de kullanılabilir.

*HCR'de Mutasyon,*mevcut grupların parçalanması, oluşan yeni alt grupların ayrık gruplar olarak bırakılması veya kalan gruplara kaynaştırılması ile uygulanır.

3.3.3.4 Kröger'in Kutu Paketleme İçin GA Çözümü

Kröger [16]'da paketleme için, herbir düğümü yerleşimdeki bir dikdörtgeni (ve yönünü) gösteren bir ikili ağaç kodlaması kullanır. Ağaçta iki tür dal vardır; ü-dal, s-dal. r_i dikdörtgeninden r_j dikdörtgenine doğru bir ü/s dalı olması; r_j'nin r_i'ye bitişik üst / sağ komşu olduğunu gösterir. Bir düğüme sadece bir ü/s daldan gelinir (kök hariç). Bu dal dikdörtgenin konumunun sadece bir bileşenini belirler. Dik yerleşim yapıldığından kutuların yerini belirtmek için iki bileşen (kutunun sol alt köşesinin koordinatları) yeterlidir. İkinci bileşen r_j nin geçerli bir alt-sol desen olarak paketlenmesi sonucunda belirlenir. alt-sol paketleme parçanın yerleşebilceği en alt noktaya sola dayalı olarak yerleştirilmesidir.

Genetik koddan (Genotype), Paketleme desenine geçiş (Phenotype) kökten başlayarak, üst düğümü yerleştirilmiş dikdörtgenlerin sırayla yerleştirilmesinden elde edilir. Ancak adımlarda yerleştirilme şartını yerine getirmiş birden fazla dikdörtgen olabilir. Burada yapılan seçim sırası yerleşkeyi etkiler ve bir genetik koda ait paketleme deseninin tekilliğini engeller. Tekil bir paketleme deseni için düğümlere öncelik değerleri atanır. Aynı anda birden fazla yerleştirilmeye hazır dikdörtgen bulunması durumunda en yüksek önceliğe sahip dikdörtgen yerleştirilir.



Şekil 3.5 Farklı öncelik değerlerine sahip iki kodun paketleme sonucu

Şekil 3.5'deki örnekte sadece öncelik değerleri farklı iki genetik kodun yerleşimleri görülür. Şekil 3.5 (a) da r_1-r_2-r_6-r_5 yerleştirildikten sonra öncelik(r_3) > öncelik(r_4) olduğundan r_3, r_6 nın sağına yerleştirilir.alt-sol şartını sağlamak için de aşağı kayar. Şekil3.5 (b) de ise r_1-r_2-r_6-r_5-r_4-r_3 sırası izlenir.

Bu kodlamada herbir birey için G = (V,E,O,P) dörtlü bilgisi tutulur.

V = $\{r_1, r_2, ...r_n\}$ Dikdörtgen dizisi

E : Yukarıda tanıtılan ikili ağaç

O : R dikdörtgeni {döndürülmüş, döndürülmemiş}

P : R dikdörtgeninin öncelik değeri

Kodlamanın dezavantajı, oluşturacağı paketleme deseninin geçerlilik ve kalitesinin yerleşimin tümü yapılmadan bilinememesidir.

Değerlendirme fonksiyonu olarak üç yeni fonksiyon önerilir

h : serimin yüksekliği

w : serimin genişliği

s_h / s_w : en üst/sağ katmandaki dikdörtgenlerin kenar uzunlukları

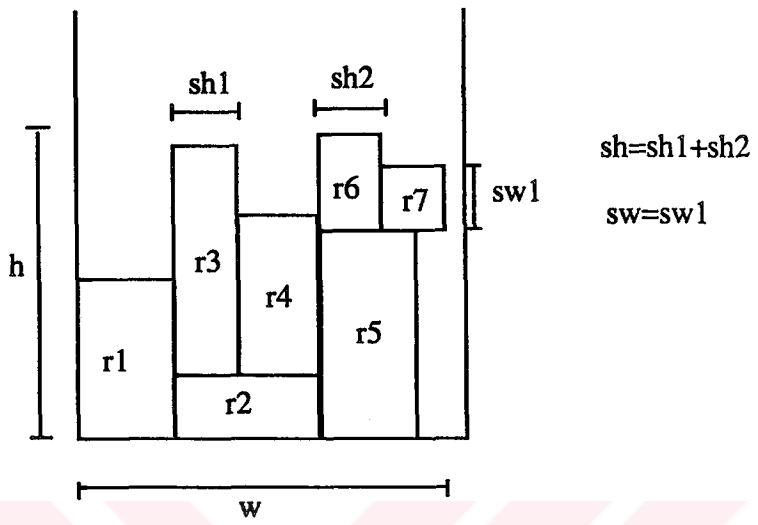
ölçek : $10^{\lceil \log_{10} w \rceil}$

olmak üzere

$$\varepsilon 1(\Phi) = \frac{olcek^2}{(((h * olcek) + w) * olcek) + s_h} \quad \textbf{(3.9)}$$

$$\varepsilon 2(\Phi) = \frac{olcek}{(h * olcek) + s_h} \quad \textbf{(3.10)}$$

$$\varepsilon 3(\Phi) = \frac{olcek^2}{(h * olcek) + (s_h + (h - s_w))} \quad \textbf{(3.11)}$$



Şekil 3.6 Örnek ölçüler

ε1 fonksiyonu; h, w, s_h özelliklerine bağlıdır. Aynı h değeri için küçük w değerine sahip birey yüksek uygunluk değeri alır. Aynı h, w değerlerine sahip iki birey ile karşılaşılınca s_h belirleyici olur. Küçük s_h değeri, üstteki yükselti dışında yoğun bir paketleme şemasına işaret eder. ε1 değerlendirme ölçütü olarak alındığında, kabın tüm genişliği w'nin doldurulmadığı görülür; dar ve yüksek içine dikdörtgenlerin sığmadığı deliklerin oluşur .

ε2 bu durumu engellemek için w değerini içermez. ε1 deki s_h kullanımından elde edilen deneyimlere dayanarak **ε3** de s_h 'a benzer şekilde s_w değeri hesaplara katılır. s_w nin maksimum değerinde verim artar.

[16]'da üç tür mutasyon tanımlanmıştır.

1. μE : Ağacın dal kümesinin, rj dikdörtgeninin yerini değiştirmek veya ri, rj dikdörtgenlerinin yerlerini karşılıklı değiştirmek için değiştirilmesi.

2. μ0 : rj dikdörtgeninin 90° döndürülmesi.

3. μp : rj dikdörtgeninin öncelik değerinin değiştirilmesi.

Yeniden Oluşturma ve Çarprazlama için ise şu yol izlenir:

Φf ve Φm bireylerinden $\Omega=(V, E_\Omega, o_\Omega, p_\Omega)$ şeklinde yeni bir birey oluşturmak için bir alt ağaç Ts=(Vs, Es) seçilir, Ω bireyine alınır. ($Vs \subseteq Vf$, $Es \subseteq Ef$) Φm'in paketleme şeması oluşturuluyormuş gibi ağaç dolaşılır. Bulunulan düğüm Ts de yer alıyorsa atlanır, yer almıyorsa alt-sol şartını sağlayan en alt noktaya yerleştirilir ve ağaçta bu yerleşime uygun dal ve düğümler oluşturulur.

3.4 Mevcut Çözümlere Yapılan Değişiklikler ve Gerçekleme

Tezde Chazelle'in [17] de tanıttığı alt-sol sezgiseli temel alınıp; genetik algoritmalarda kod çözme aşamasına uygulanarak başarımın arttırılması hedeflenmiştir. GA'da parça listesi ve bir kod çözme algoritması kullanan bir çözüm gerçeklenmiştir. Parça listesine uygulanan genetik işlemler sıralama tabanlıdır. Kod çözme algoritması ufuk

çizgisi (skyline pack) algoritmasına benzer ancak arada oluşan deliklerin de kullanılmasını sağlar.

3.4.1 Uygulanan Genetik İşlemler

Kod; Listedeki dikdörtgenlerin yerleştirilme işlemine giriş sırası olarak alınır.
*Çarprazlama s*ıralama tabanlıdır.

katar1 : 1 - 2 | 3 - 4 - 5
katar2 : 5 - 4 | 5 - 4 -3

çocuk : 1 - 2 | 5 - 4 - 3

Şekil 3.7 Çaprazlama

Çocuk katarda kesim noktasına kadar katar1 deki sıra izlenirken, kesim noktasından sonra daha önce sıralamaya alınmamış parçalar için katar2 deki sıra dikkate alınır.

Mutasyon; Rastgele seçilen iki parçanın yerleri karşılıklı olarak değiştirilmesi ile sağlanır.

Değerlendirme (Uygunluk) Fonksiyonu; Paketlenen parçaların toplam alanının, kap alanına oranı heas. Kap alanı, kap (kumaş) genişliği ile yerleşimin yüksekliğinin çarpımı ile bulunur.

3.4.2 Uygulanan Kod Çözme Algoritması: Alt-Sol Sezgiseli

Birçok uygulama, dikdörtgenlerin, verilen bir dikdörtgensel alana etkin olarak yerleştirilmesini gerektirir. Baker, Coffman ve Rivest üstü açık dikdörtgensel bir kutunun R, N dikdörtgenle $R_1...R_N$ toplam kutu yüksekliğini minimize edecek şekilde doldurulmasına kombinezonsal bir model önerir. Dikdörtgenlerin örtüşmemesi gerektiği şartına ek olarak, bu model sadece dik ve yönlü paketlemeyi göz önüne alır. Dik paketlemede bütün kenarlar R kutusunun tabanına yada dikey kenarlarına paraleldir. Yerleştirilen dikdörtgenler hiç dönmüyorlarsa da paketleme yönlüdür.

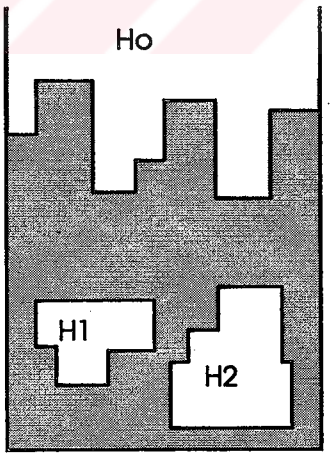
Optimal paketleme NP- zorlukta olduğu için değişik yaklaşımlar kullanılmıştır. Chazelle, Alt-Sol sezgisel yaklaşımını kullanır [17]. Strateji dikdörtgenleri yerleşebilecekleri en alt bölgeye sola dayalı olarak yerleştirmeye dayanır. Bu işlem, listede verilen sıraya göre tüm dikdörtgenlere uygulanır. Baker ve Coffman tarafından gösterilmiştir ki, düzgün sıralanmamış listeler optimal paketlemeye göre kötü sonuçlar verse de, listeyi sadece azalan genişliğe göre sıralamak toplam yüksekliğin, en fazla optimal yüksekliğin üç katı olacağını garanti eder. Pratikte görülmüştür ki AS stratejisi oldukça iyi sonuçlar verir ve basitliği birçok uygulama alanında kullanılmasını sağlar. Ama kavramsal basitliği uygulama düzeyinde devam etmez, $O(N^4)$ veya en iyi durumda $O(N^3)$ algoritmalar gerçeklenebilmiştir. Chazelle ise O(N) yer, $O(N^2)$ zaman karmaşılığı olan bir yöntem sunar.

Bir dikdörtgene yerleştirildiği konumdan daha aşağı yada sola kayamıyorsa, alt-sol kararlı paketlenmiş denir. Alt-Sol heuristiği bu şartı sağlar.

3.4.2.1 Alt-Sol Algoritmasının Tanımı

Sezgisel yaklaşımın herbir aşamasında, kutu bir küme boş alana (deliğe) sahiptir: $H_0, H_1, \ldots$. Bu delikler yatay ve dikey kenarları olan poligonlardır. H_0 deliği her zaman vardır ve o ana kadar paketlenmiş dikdörtgenlerin üzerindeki sınırlanmamış alandır.

Algoritma, sıradaki dikdörtgeni yerleştirmek için herbir Hj deliği içine R_i 'nin sığıp sığmayacağını test eder. İşlemi görselleştirmek için R_i dikdörtgeni aralarında onları dışa doğru iten bir yayın olduğu iki çubuk gibi düşünülebilir. Sonuçta dikdörtgenlerin boyu deliğin alt ve üst çeperlerinin elverdiğince büyüktür. Arama işlemi bu çubukları soldan sağa kaydırılması ve yüksekliğin değişimi gözlenerek, yerleştirilmek istenen dikdörtgenin yüksekliği h'yı aşmasını beklemekten oluşur. Bu durum oluştuğunda o anki durum bir çözüm adayı olarak rapor edilir.



Şekil 3.8 Genel delik düzeni

Çözülmesi gereken temel problemler:

1. Delikleri temsil etmek için nasıl bir veri yapısı kullanılmalıdır

2. Çubuklar delik içinde nasıl kaydırılmalıdır

3. R_i için bir konum saptandıktan sonra veri yapısı nasıl güncellenecektir

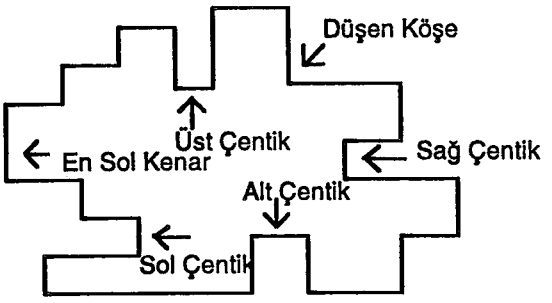
3.4.2.2 Deliğin Özel Noktaları

Her düğüm köşelerden oluşan çift yönlü bağlantılara sahip bir bağlantılı liste ile gösterilir. Köşeler listede delik çeperindeki sıralarıyla yer alırlar. Deliğin bazı köşe ve kenarlar diğerlerinden ayrı bir rol oynarlar.

En sol kenar: Bir kenarın komşu iki kenarları da yatay ise, incelenen kenarın sağında kalıyorlar ise ve incelenen kenarla yaptıkları açı dar açı ise bu kenar *en sol kenar* olarak tanımlanır.

Sol Çentik (Notch): Bir kenarın komşu iki kenarları da yatay ise, incelenen kenarın sağında kalıyorlar ise ve incelenen kenarla yaptıkları açı 180° den büyükse, bu kenar çentik olarak tanımlanır. Alt, üst ve sağ çentikler de benzer şekilde tanımlanır.

Düşen Köşe: Dar açılı, sağında yatay bir kenar, solunda dikey bir kenar bulunan köşe, *düşen köşe* olarak tanımlanır.

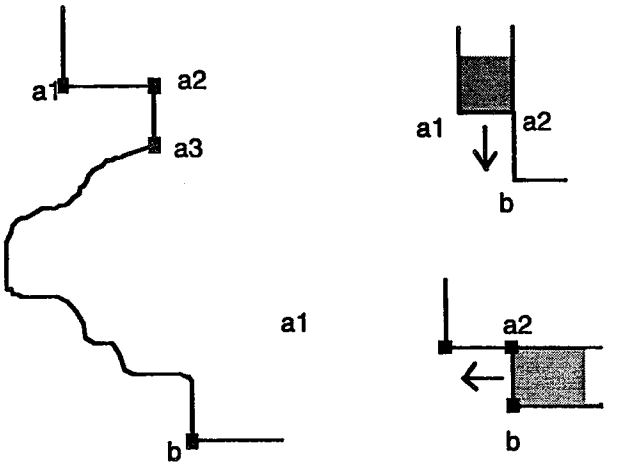


Şekil 3.9 Deliğin Önemli Nokta ve Kenarları

3.4.2.3 Veri Yapısının Kurulması

Veri yapısının kurulmasında ve daha sonra deliğin çözüm adaylarını bulmak üzere taranmasında, yöntemde parçaların alt-sol kararlı yerleştirilmelerinden doğan bazı delik özelliklerinin yardımı büyüktür. Parçaların alt-sol kararlı yerleştirildiği yöntemlerde, bir deliğin sağ veya üst çentiği olamaz ve en fazla bir düşen köşesi vardır.

Bu iddia şöyle kanıtlanır: Sağ ve üst çentiklerin olaması alt-sol kararlılığı bozar. En fazla bir düşen kenar olması ise Şekil 3.10'dan da yararlanarak da şöyle kanıtlanır: $a_1a_2a_3...b$ a_1 ve b düşen köşelerini birleştiren çeper parçası olsun; a_1, b saat yönünde verilmiş olsun. şekil 3.10 da üst çentik olmadığı için a_3 a_2 'nin altında yer alır. Sağ çentği engellemek için de a_2 ve a_3 nin $a_1a_2a_3...b$ arasında minimum x koordinatına sahip olmaları gerekir. Bu a_3 ün düşen kenar olmasını gerektirir; dolayısıyla a_3=b. Sonuç olarak a_1a_2 kenarına sahip dikdörtgen veya a_2b kenarına sahip dikdörtgen kararlı durumda değildir. Alt-sol sezgiseli kullanılarak böyle bir duruma ulaşılamaz.

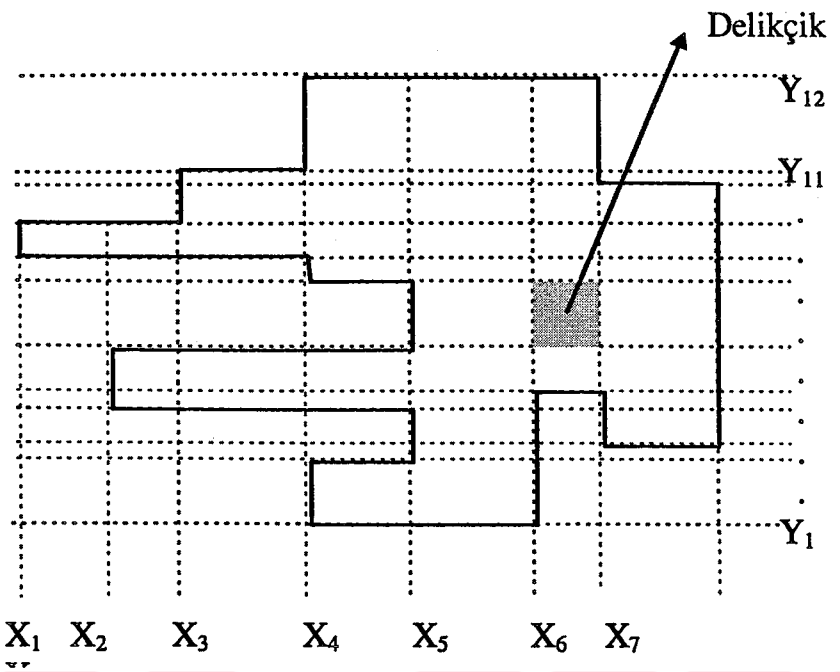


Şekil 3.10 Düşen kenar

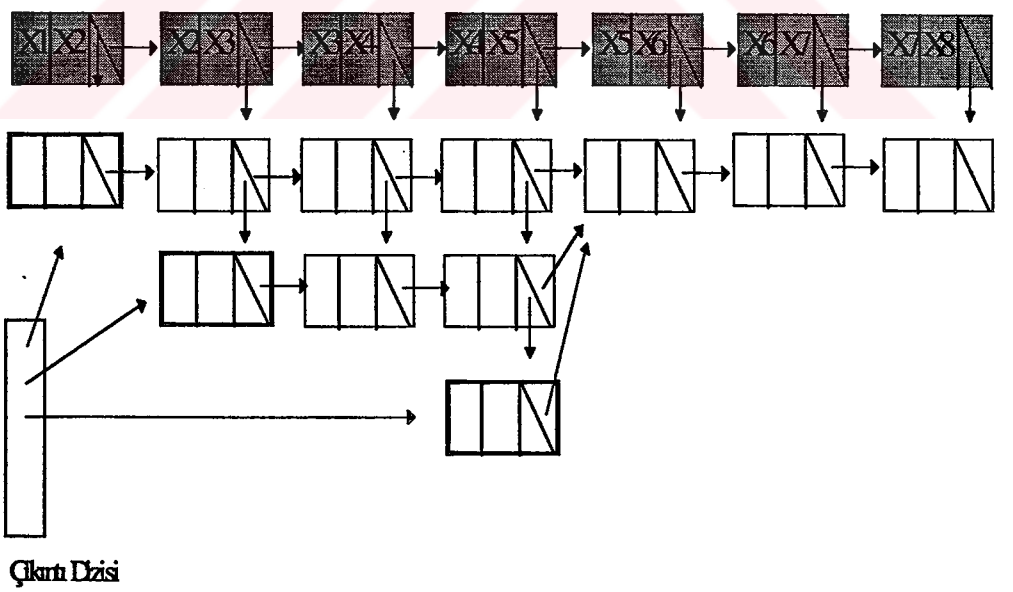
Gerçeklene veri yapısında delik, herbiri alt ve sağ komşusuna bağlı, dikdörtgen şekilli delikciklerden oluşur. Sağ çentiğin olmaması deliğin ancak tek bir sağ komşusu olmasını sağlar.

Delikcik :

```
struct delikcik
{
        int ytop;
        int ybottom;                    // delikciğin sınırları
        int xright;
        int xleft;
        int gecildi;
        struct delikcik  *ynext;        // delikciğin sağındaki deliğe işaretçi
        struct delikcik *xnext;         // delikciğin altındaki deliğe işaretçi
};
```



Şekil 3.11 Örnek bir delik



Şekil 3.12 Örnek deliğe ait veri yapısı

Delikçiklere erişmek için sütün listesi ve çıkıntı dizisi kullanılır.

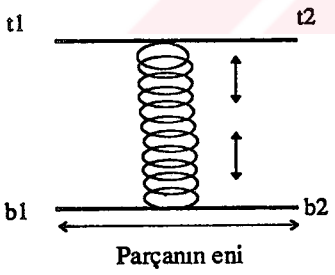
Sütun:

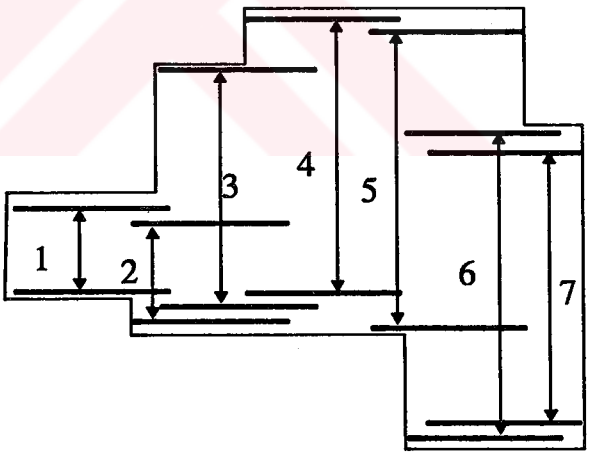
```
struct sutun
{
        int xright;
        int xleft;                  // sutun sınırları
        struct sutun  *xnext;       // sutunun sağındaki sutuna işaretçi
        struct delikcik *ynext;     // sutundaki ilk delikciğe işaretçi
};
```

Çıkıntı Dizisi :

Solunda delikçik olmayan, yani kendine gelen bir işaretçi olmayan sol çıkıntılara işaretçi dizisi.

3.4.2.4 Yeni Bir Parçanın Delikte Yerleştirileceği Konumun Saptanması





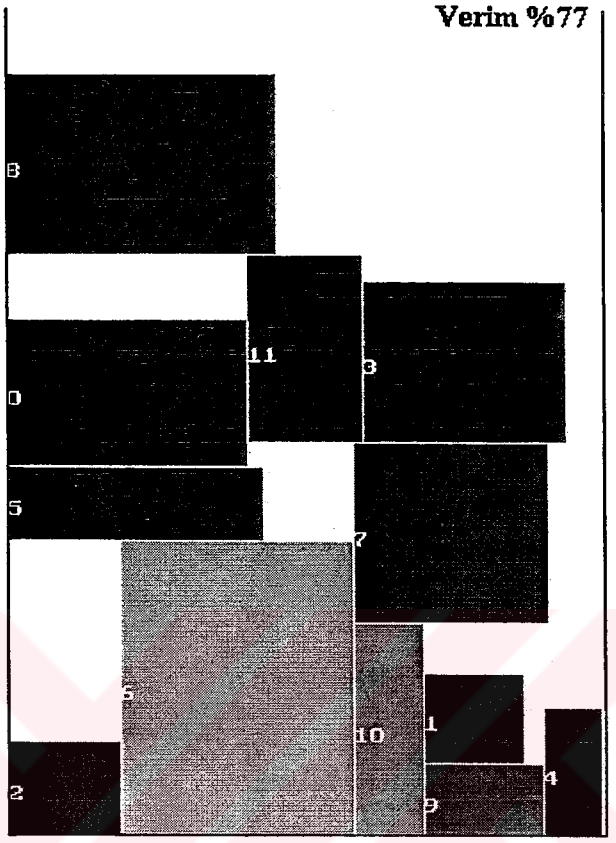
Şekil 3.13 Yaylı Çubuklar Modeli

Şekil 3.14 Deliğin taranması

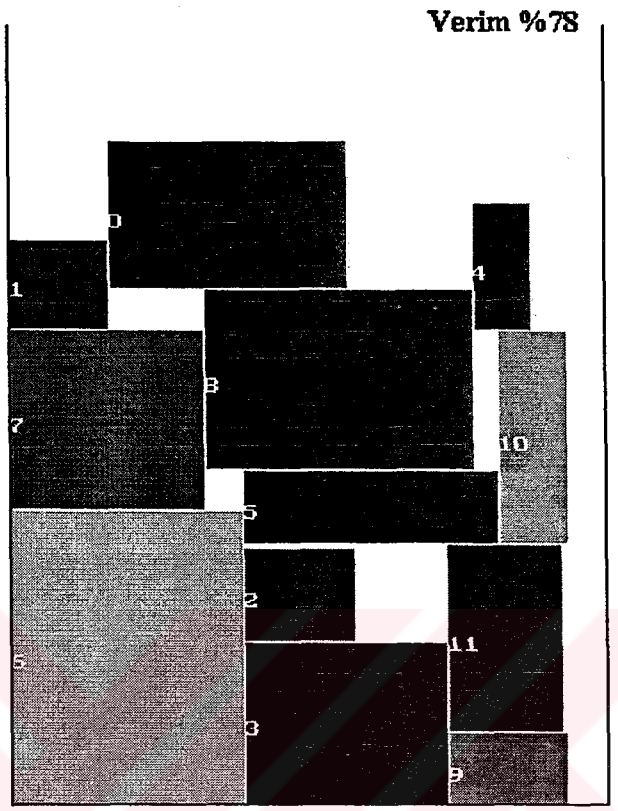
Deliğe yeni bir parçayı yerleştirmek için eni parçanın enine eşit iki çubuktan ve bu çubukların arasına yerleştirilmiş bir yaydan oluşan yaylı çubuklar benzetimi yapılır. Çubukların boyu alabileceği maksimum yükseklik olur. Yeni bir parça yerleştirileceği zaman, parçanın yerleşebileceği konumları bulmak için çubuklar delik içinde, çıkıntı dizisindeki herbir sol çıkıntıdan başlayarak soldan-sağa kaydırılır. Yay cihazının boyu, yerleşecek parçanın boyuna eşit yada büyük olduğunda bu konumlar, yerleşime aday konumlar olarak bir aday konum listesine eklenir. Tarama işlemi bittiğinde eğer aday konum bulunmuş ise bu konumlar arasından en altta ve en solda olan konum seçilir ve parça yerleştirilir.

Sütun listesi veri yapısının güncellenmesi işlemini kolaylaştırmak için kullanılır. Veri yapısının güncellenmesi sırasında parçanın tamamen doldurduğu delikcikler bağlantılı listeden çıkartılır; kısmen doldurulan delikciklerin sınırları değiştirilir; bu delikciklere gelen ve bu delikciklerden çıkan işaretciler yeni yapıya göre düzenlenir. Parçanın deliği parçalayıp parçalamadığı yani birden fazla deliğe bölüp bölmediği kontrol edilir.

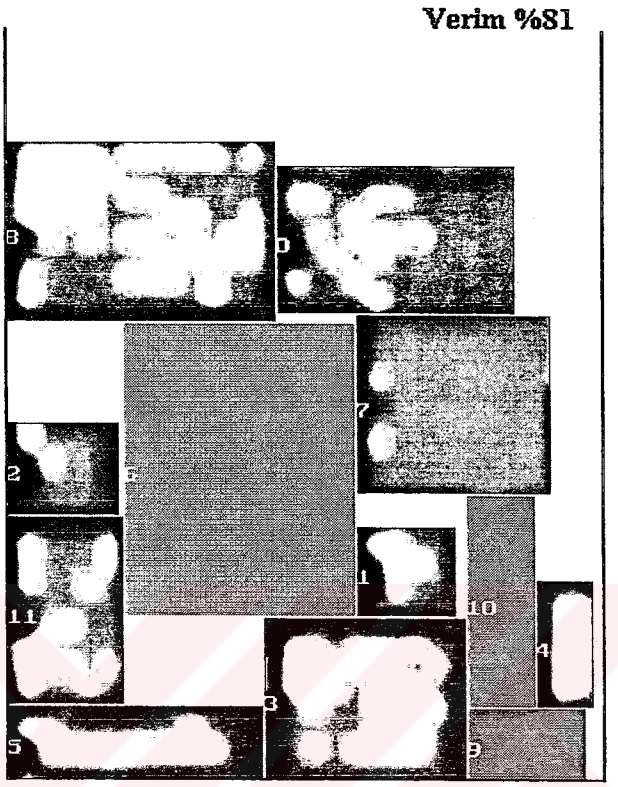
Şekil 3.15 - Şekil 3.19'da aynı parçaların oluşturduğu farklı genetik kodların, alt-sol sezgiseli kullanılarak yerleştirilmesi sonucu elde edilen farklı düzen ve verimler yer alır.



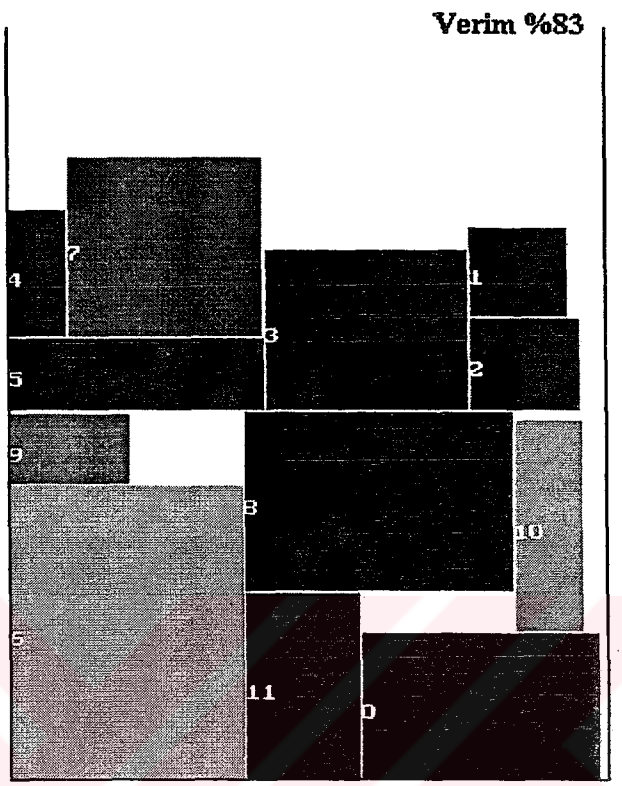
Şekil 3.15 Örnek Yerleşim 1



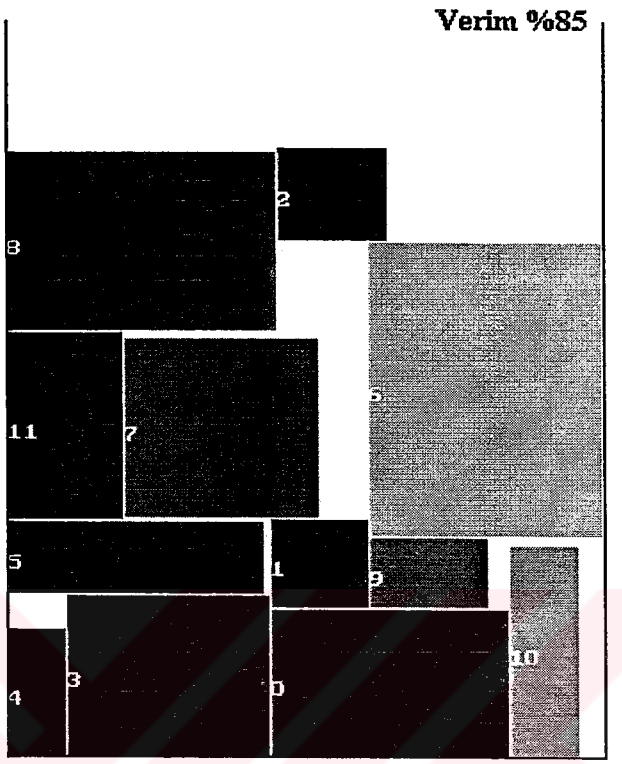
Şekil 3.16 Örnek Yerleşim 2



Şekil 3.17 Örnek Yerleşim 3



Şekil 3.18 Örnek Yerleşim 4



Şekil 3.19 Örnek Yerleşim 5

BÖLÜM 4

DİKDÖRTGEN OLMAYAN PARÇALARIN YERLEŞTİRİLMESİ

4.1 Dikdörtgen Olmayan Parçaların Yerleştirilmesi Üzerine Çalışmalar

[18]de anlatılan çalışmalarında, Adamowicz ve Albano iki temel adımdan bahseder:

1. *Modül* olarak adlandırdıkları, şekil veya şekil kümeleri için kapsayan dikdörtgenlerin bulunması

2. Dikdörtgen levha üzerinde bu modüllerin optimal yerleşiminin bulunması.

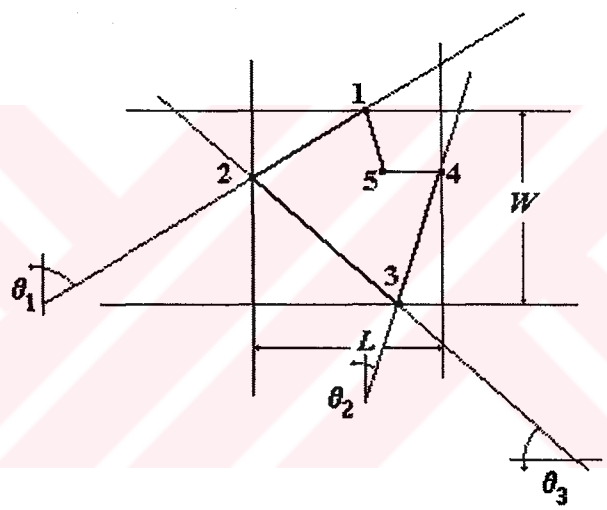
4.1.1 Optimal Kapsayan Dikdörtgen Bulunması

Optimal kapsayan dikdörtgeni bulmak için sezgisel-algoritmik karışımı bir yol izlenir. Bir poligon verildiğinde, bir grup iyi sonuç verebilecek dönüş açısı bulunur. Bu açıların bulunması için, deneyimli parça yerleştiricilerin kullandıkları bir teknikden yararlanılır. Buna göre dönüş açısı, parçanın kenarlarından bir ya da birkaçını, yataya ya da dikeye paralel getiren açı olarak seçilir. İşlemde sadece dışbükey köşeler dikkate alınır.

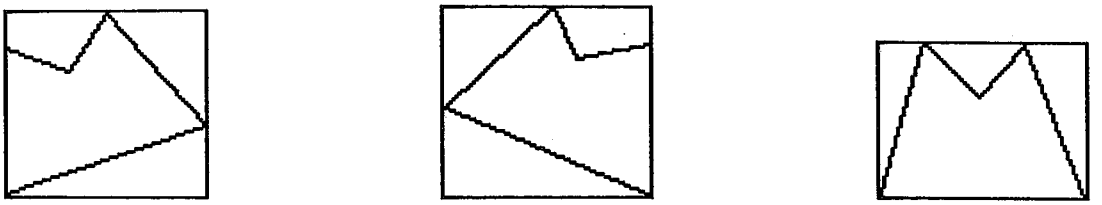
Dışbükey köşe, söz konusu köşeye gelen vektör ile çıkan vektörün vektörel çarpımının negatif olmadığı köşedir.

Gelen vektör (x_{i-1}, y_{i-1}) noktasından (x_i, y_i) noktasına gelen vektör, *çıkan vektör* ise (x_i, y_i) noktasından (x_{i+1}, y_{i+1}) noktasına giden vektördür.

Gelen vektör ile çıkan vektörün vektörel çarpımı: $(x_i - x_{i-1})(y_{i+1} - y_i) - (y_i - y_{i-1})(x_{i+1} - x_i)$



Şekil 4.1 Kapsayan dikdörtgeni bulunacak şekil



(a) (b) (c)

Şekil 4.2 Şekil 4.1'in sırasıyla θ_1, θ_2, θ_3, açıları kadar döndürülmüş hali

Bu açıların herbiri için çokgen döndürülür. En küçük kapsayan dikdörtgeni veren açı seçilir ve çokgen bu konumda sabitlenir.

Parça kümelerinin oluşturulması için ise *örtüşme çokgeninden* (no-fit polygon) yararlanılır.

B çokgeninin , A çokgenine göre *örtüşme çokgeni*, B çokgeninin A çokgenine değdiği ancak örtüşmediği tüm konumlar için B çokgeninin referans noktasının taradığı konumları tanımlar. Kısaca bu çokgen, A ve B çokgenlerinin birbiri ile örtüştüğü ve örtüşmediği bölgeler arasındaki sınırı oluşturur. Bu çokgen daha sonraki bölümlerde anlatılacak minkowski toplamının sonucu olan çokgenden farklı bir çokgen değildir.

4.1.2 İkişer \ İkişer Kümeleme

Çözülmesi gereken problemler:

- Bir çift şeklin seçilmesi,
- Şekillerden biri için mutlak bir konumun seçilmesi,
- İkinci şekil için birinciye göre bağıl bir konumun seçilmesi.

Çokgenlerin birbirlerine göre yerleştirilebileceği konumları bulmak için örtüşme çokgeni kullanılır. Bu konumlar arasından minimum kapsayan dikdörtgeni veren konum, örtüşme poligonunun kenarları üzerinde yer alır.

4.1.3 Parça Yerleştirmenin Graf Üzerinde Arama Problemi Olarak Ele Alınması

[19] da yerleştirme problemi, aday sonuçlar uzayında arama prosesine dönüştürülür. Bu uzay genellikle çok geniş olduğundan sezgisel yaklaşımlar kullanılır. Problem bir grup durum ve durum geçişlerini sağlayan bir grup işlem ile tanımlanır. Problem durumların düğüm, işlemlerin de dallarla temsil edildiği bir grafta, ilk düğümden hedef düğüme bir yol aramaya dönüşür. Yeni duruma, varolan yerleşime, bir parçanın daha yerleştirilmesi ile geçilir. Durum geçişinin maliyetini bu yerleştirme sonucu oluşan atık belirler.

Y_i : Anlık yerleşim

P_i : Yerleştirilen parça

$A(Y_i)$: Y_i yerleşiminde kullanılabilir alan olarak kalan bölge. Parçalar aşağı doğru yerleştiriliyorsa, yerleşmiş parçaların oluşturduğu sınırın üstünde kalan bölge.

olmak üzere,

$$\text{Eklenen\_Atık} = A(Y_{i-1}) - (A(Y_i) + \text{Alan}(P_i)) \quad (4.1)$$

4.1.3.1 Arama işleminin adımları

1. Başlangıç düğümünü GENERATED listesine koy.
2. GENERATED boşsa hata ver ve çık; değilse devam et.
3. GENERATED listesinden, R kuralına göre bir düğüm seç; EXPANDED (Genişleme) listesine koy, bu düğüm n olsun.
4. n bir sonuç düğümse, sonuç yol ile programdan çık; değilse devam et.
5. n'yi genişlet; yani ardından gelenleri üret, ardından gelen yoksa 2. adıma git; varsa GENERATED listesine koy 2. adıma git.

İşlemi tamamen belirlemek için, genişletilecek düğümü seçen R kuralı belirlenmelidir. Algoritmalar, arama işleminde probleme ait bilgilerin kullanılıp kullanılmamasına göre sınıflandırılır. Bu bilgi, sezgisel bilgi olarak adlandırılır.

[19] da n düğümünün optimal yol üzerinde olabilirliğini gösteren bir maliyet tahmin fonksiyonu öne sürülür :

$$\hat{f}(n) = \hat{g}(n) + \hat{h}(n) \qquad (4.2)$$

g(n) : başlangıç düğümünden, n düğümüne en kısa yolun maliyeti

$\hat{g}(n)$: g(n) tahmini

h(n) : n'den son düğüme en kısa yolun maliyeti

$\hat{h}(n)$: h(n) tahmini

Her iterasyonda, toplam maliyet tahmini $\hat{f}(n)$'yi en küçük yapan n düğümü seçilir. $\hat{g}(n)$ için açık bir seçim, başlangıç düğümünden n düğümüne kadar arama sırasında adım adım hesaplanmış en kısa yolun maliyetidir; atık fonksiyonu kullanılarak hesaplanır. $\hat{g}(n)$ katedilmiş bir yola ait maliyet iken $\hat{h}(n)$ henüz katedilmemiş, o ana kadar yerleştirilmemiş parçaların yerleştirilmesi sonunda beklenen maliyettir. $\hat{h}(n)$ n düğümünden hedef düğüme en düşük maliyetli yolun maliyetinin alt

sınır olduğunda aramanın optimal yol ile sonuçlanacağı [19]da belirtilir. $\hat{h}(n)$'nin seçimi, arama algoritmasının özelliklerini fazlaca etkiler. Ancak problem karmaşıklaştıkça h(n)'nin anlamlı bir tahminini bulmak olanaksızlaşır ya da zaman alır. Bu durumda optimal sonucu kaybetme pahasına hesap yükü daha az bir çözüme yönelinir.

4.1.3.2 Aramanın Etkinliğini Arttıracak Sezgisel Yaklaşımlar

1. Maliyet Tahmin Fonksiyonu:

Yukarıda da belirtildiği üzere $\hat{h}(n)$ değerinin bulunması, problem karmaşıklaştıkça olanaksızlaşır. Arama sonucunun optimal olması isteği bırakılır, "iyi" bir sonuç ile yetinilirse, bu fonksiyon için henüz yerleştirilmemiş parçaların toplam alanlarının bir yüzdesi beklenen atık olarak alınabilir. Bu karar, aynı zamanda büyük parçaların önce yerleşmesini sağlar.

2. İzleyen Sınırlaması:

Bir düğümün izleyenlerini bulmak için, n adet yerleştirilmemiş parçanın herbiri için k adet yön denenir ve izleyen olarak eklenir. Bu adım en çok zaman harcayan adım olduğundan bir takım kısıtlamalar getirilebilir.

3. Genişletme Bandı:

Arama sırasında k. düzeyde iken genişletilecek bir sonraki düğüm; t eşik değeri olmak üzere, (k-t) den daha aşağı bir düzeyde olmamalıdır. Aslında, arama uzun zamandır umut verici bir yol üzerinde ilerlemiş ise genişleme bandı, geri dönülüp önceden askıya alınmış daha iyi bir yola devam edilmesini engelleyebilir. Ancak hesap yükünü azaltmak için bazı fedakarlıklarda bulunmak zorunludur.

4. Sınır Sadeleştirme:

Her aşamada yerleşmiş parçalar ile, boş alan arasını ayıran sınır basitleştirilerek hesap yükü azaltılabilir.

4.1.4 Tektip Dışbükey Çokgenlerin Yerleştirilmesi

[20]de Verilen bir yüzeyi, minimum kayıpla tektip şekillerle kaplama üzerine çalışılır. Aynı algoritma içbükey çokgenlere, dışbükey çeperleri kullanılarak uygulanabilir. Algoritmanın esası, yerleştirmede kullanılacak çokgenin *döşeme çokgeni* olarak adlandırılan çokgenlere, minimum alan kaybı ile dönüştürülmesine dayanır.

Döşeme çokgeni, çoğullanınca yüzeyi tamamen kaplayan çokgenlerdir. Geometride bilinen birtakım üçgen, dörtgen, beşgen, altıgen döşemeler vardır. Çalışmada altıgen döşemeler, daha az alan kaybı ile sonuçlandıklarından kullanılırlar.

4.2 Tezde Gerçeklenen Çözüm

4.2.1 Amaç

Verilen çokgenler kümesinin örtüşmeyen düzenini bulmak; çokgenleri kabul edilebilir verim ve zamanda yerleştirmektir. Bu tezin kapsamında ötelemeli yerleştirme ele alınmıştır; parçaların sadece x,y yönlerinde ötelenmesine izin verilir, parçalar döndürülmez.

4.2.2 Çözülmesi Gereken Problemler

Önerilen yaklaşımda yerleştirilecek parçalar sırayla yerleştirilecekleri kaba alınır, pastal yerleştirmede kap, eni mevcut kumaşın eni, boyu ise yerleşim sonunda belirlenecek kumaş boyudur. Yeni yerleşecek parça daha önce yerleştirilmiş parçalarla örtüşmeyecek (üst üste binmeyecek) ve kap çeperini aşmayacak şekilde yerleştirilir. Parçalar yerleştikten sonra geri dönüş yoktur, bir kez yerleştirilen parçanın konumu değiştirilmez. İşlem tüm parçalar yerleşene kadar tekrarlanır.

Çözülmesi gereken üç temel problem vardır:

1. **Parçaların örtüşmeyeceği bölgeyi bulma:** Yerleştirilecek parçanın daha önce yerleştirilmiş parçalarla örtüşmeden ve kap çeperinden taşmadan yerleştirilebileceği tüm konumların bulunması.

2. **Aday konumlar arasından uygun konumu seçme:** 1'de saptanan konum kümesinden uygun bir konumun seçilmesi.

3. **Yeni parçalar yerleştikçe veri yapısını ve yerleşim düzenini güncelleme.**

Parçaların örtüşmedikleri bölgeyi bulmak için çeşitli çarpışma bulma (collision detection) algoritmaları kullanılabilir. Çalışmada Minkowski Toplamı ve Farkından yararlanılmıştır. Minkowski topalm ve farkı çokgen-çokgen kesişimi (örtüşmesi) ve çokgen-çokgen kapsama problemlerini "nokta çokgenin içinde mi ?" sınamasına dönüştürür.[19]da kullanılan örtüşmeme çokgeni "no-fit polyygon" da aslında minkowski toplamından farklı birşey değildir [3].

Aday konumlar arasında uygun konumu seçmek için iki yaklaşım kabul edilmiştir:

1. Kapsayan dikdörtgenleri hesaplayıp; kapsayan dikdörtgenin alanını en küçük yapan konumu seçme: Bu yöntem kümeleme işlemi için de uygundur. Kümelemeye dayalı çokgen yerleştirmede, parçalar önce parçaları kapsayan dikdörtgenin "kullanılan alan/tüm alan" oranını küçültecek şekilde kümelenir. İkinci aşamada tüm yerleşimi elde etmek için parçalar değil, parçaların oluşturdukları kümelerin kapsayan dikdörtgenleri uygun şekilde yerleştirilmeye çalışılır.

2. En alt -en sol konumu seçme: Alt-sol sezgisel yaklaşımına benzer şekilde parça, yerleşebileceği tüm uygun konumlar arasından en aşağıya sola dayalı olarak yerleştirilir.

Birinci yaklaşımın olumsuzluğu eğer kümeleme için değil de doğrudan yerleştirme için kullanılıyor ise, kapsayan dikdörtgen verimi iyi ancak kabın tüm enini kullanmayan ince uzun yerleşimlerle sonuçlanmasıdır. Buna karşılık şekillerdeki içbükey bölgeler iyi değerlendirir. Parçaların önceki şekillerin girintilerine yerleştirildikleri konumlar, kapsayan dikdörtgenin alanını arttırmadığından tercih edilme eğilimindedir.

İkinci yaklaşımda parçaları mümkün olan en aşağı sola dayalı yerleştirmede, "her parça yerleşebileceği en alt-sol konuma yerleştirilirse iyi bir yerleşim elde edilir"sezgisel yaklaşımına dayanır. Bu yaklaşımda kumaş eninin tümü kullanılma eğilimindedir, ancak girinti dolduran konum yerine daha alttaki bir konum seçildiğinden iç kırıntılanmayla, yani parçalar arasında boşlukların kalmasıyla sonuçlanır. Her iki yaklaşımın da eksik yönleri bulunduğundan yaklaşımlarım birleştirilmesi ve karma bir sezgisel yaklaşımın kullanılması uygun görülmüştür.

Yeni parçalar yerleştikçe, veri yapısını ve yerleşim düzenini güncelleme için çokgen birleşiminden yararlanılır. Her parça yerleşiminden sonra yeni yerleşen parça ile daha önce yerleştirilmiş parçaların oluşturduğu çokgenin bileşiminin çeperi bulunur ve işleme bu yöntem ile elde edilen çokgen ile devam edilir.

4.2.3 Minkowski Toplam ve Farkı

Minkowski toplamı robotlar için hareket planlama, görüntü analizi gibi alanlarda yaygın olarak kullanılan bir kavramdır. [1] ve [3] de Minkowski çokgen işlemlerinin, paketleme ve yerleştirme problemlerinde ne kadar yararlı olduğunu gösterilmiştir. Minkowski toplam ve farkı kullanılarak, çokgen-çokgen kesişimi (örtüşmesi) ve çokgen-çokgen kapsama problemleri "nokta çokgen içinde mi ?" sınamasına dönüştürülür.

4.2.3.1 Tanım Ve Özellikler

Verilen iddia ve teoremler d-boyutlu öklid uzayında ayrık veya sürekli nokta kümeleri için geçerlidir.

Tanım: A ve B Öklid uzayında iki nokta kümesi olmak üzere $A \oplus B$ ile temsil edilen Minkowski toplamı

$$A \oplus B = \bigcup_{b \in B} A^b \tag{4.3}$$

şeklinde bir nokta kümesidir.

A^b , A'nın b kadar ötelenmiş halidir: A^b = {a+b | a∈ A}

İddia: $A \oplus B$, A ve B nin cebirsel toplamı olarak da yazılabilir.

$$A \oplus B = \{a+b | a \in A, b \in B\} = \bigcup_{b \in B} (A+b) \tag{4.4}$$

Tanıt : (⇒) x∈ $A \oplus B$ x∈A^b şeklinde bir b∈B bulunmasını gerektirir. Buna göre x=a+b olacak şekilde bir a∈A 'nın olduğu çıkar. Buna göre x aynı zamanda sağ tarafın elemanıdır. (⇐) benzer şekilde kanıtlanır.

Çıkarım: $A \oplus B = B \oplus A$

İddia : A ve B iki nokta kümesi, s ve t iki nokta olmak üzere,

$A^s \oplus B^t = (A \oplus B)^{s+t}$ **(4.5)**

4.2.3.2 Uygulama: Örtüşme Testi

Teorem: A ve B iki nokta kümesi, x düzlemde bir nokta olsun. $A \cap B^x \neq 0$ ancak ve ancak $x \in A \oplus (-B)$ isedir: Bu teorem **vuru(çakışma)** dönüşümü olarak adlandırılır. Burada $(-B)=\{-b \mid b \in B\}$ B nin global koordinat sisteminin orijinine göre aynalanmış halidir.

Tanıt :

($\Rightarrow$) $y \in A \cap B^x$ olsun. $y \in A$ ve $y \in B^x$ yani öyle bir $b \in B$ vardır ki y=b+x. Buna göre x = y+(-b).

y+(-b), $A \oplus (-B)$'de bir nokta olduğuna göre $x \in A \oplus (-B)$ dır.

($\Leftarrow$)$x \in A \oplus (-B)$ ise, $a \in A$ ve $b \in B$ için x = a+(-b). x = a+ (-b), a = x+b şeklinde yazılırsa, a hem A hem de B^x 'e aittir bu da $A \cap B^x \neq 0$ olmasıdır.

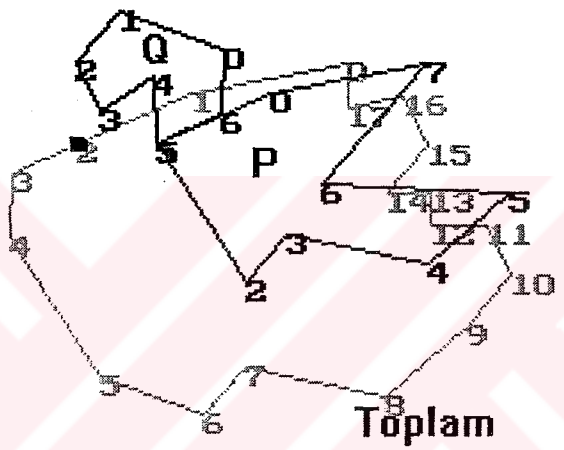
Çıkarım: A^s ve $B^t \neq 0$, ancak ve ancak $(t-s) \in A \oplus (-B)$ ise doğrudur.

Tanıt: A^s ve B^t ,ancak ve ancak ikisi de (-s) kadar ötelendikten sonra kesişiyorlarsa, kesişirler. Bu da ancak ve ancak $A \cap B^{t-s} \neq 0$ ise gerçekleşir.

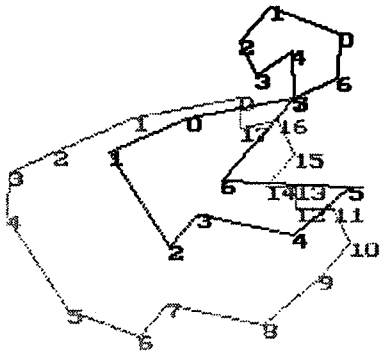
Buna göre A ve B, ancak ve ancak $A \oplus (-B)$., orijini içerirse kesişir. Bu pastal hazırlamada kullanılabilecek bir sonuçtur, çünkü gerçeklenen pastal yerleştirme programında her parçanın (çokgenin) yerel bir koordinat sistemi vardır. Çokgenin köşe noktaları yerel koordinat sisteminde tanımlanır. Çokgenin global koordinat sistemindeki konumu, çokgenin orijininin konumu ile belirlenir. P(s) P poligonunun yerel orijininin s noktasına taşındığını göstersin. Çıkarımdan P(s) ve Q(t) nin ancak ve ancak, (t-s), $P \oplus (-Q)$ nin içinde ise kesişeceğini bilinmektedir. Q' nun yerel orijini

global orijin ile çakıştığından (-Q), Q yerel orijini etrafında 180° döndürülerek elde edilir. Şekil 4.3 de görüldüğü gibi Q poligonunun yerel orijini P⊕(-Q) minkowski toplamının dışında kaldığı sürece Q P'nin üzerine binmez (kesişmez)

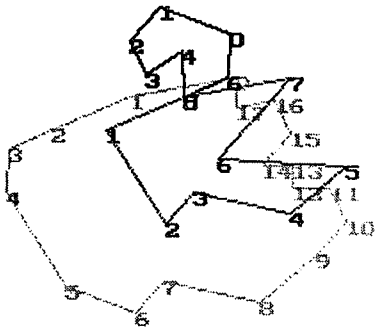
Şekil 4.4, Şekil 4.5, Şekil 4.6'de Q poligonunun, P⊕(-Q) 'nun çeperi üzerinde dolaştırıldığında, P poligonu ile örtüşmediği görülür.



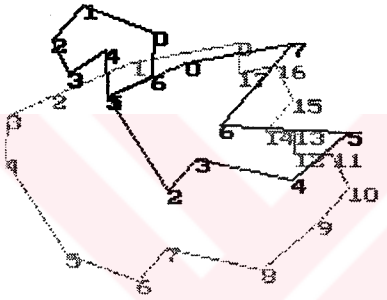
Şekil 4.3 Minkowski Toplamı



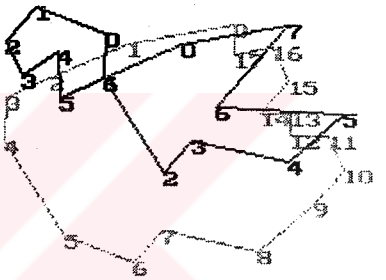
a



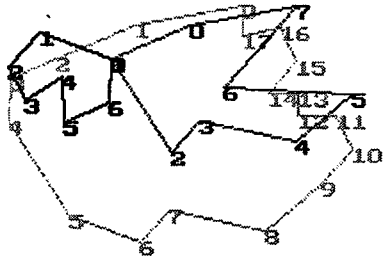
b



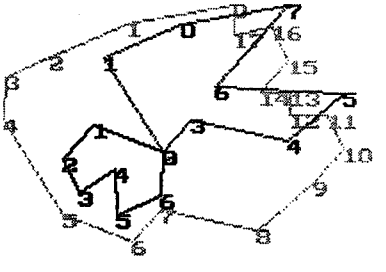
c



d

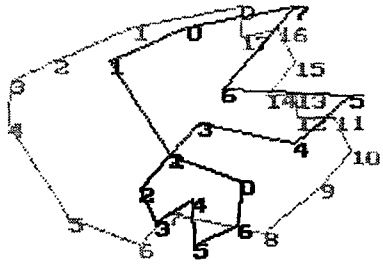


e

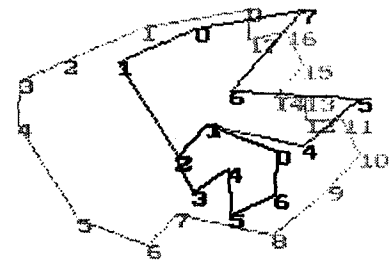


f

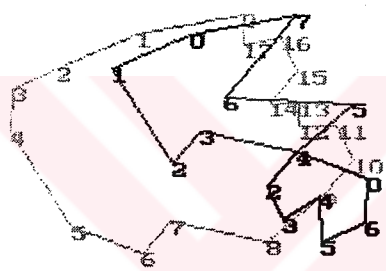
Şekil 4.4 Q poligonunun, P⊕(-Q) 'nun çeperi üzerinde dolaştırılması -1



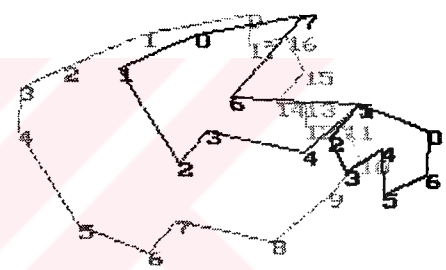
a



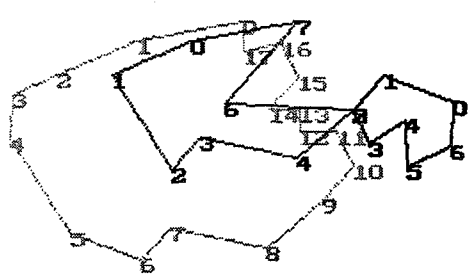
b



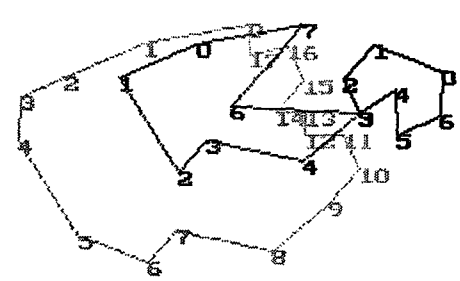
c



d

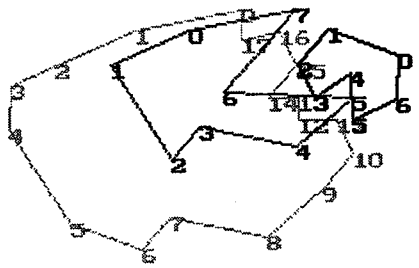


e

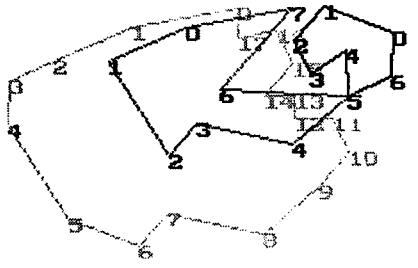


f

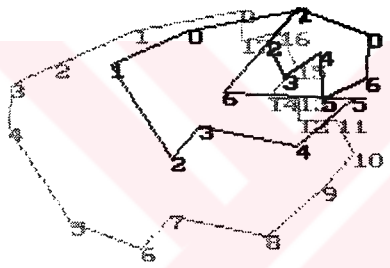
Şekil 4.5 Q poligonunun, P⊕(-Q) 'nun çeperi üzerinde dolaştırılması -2



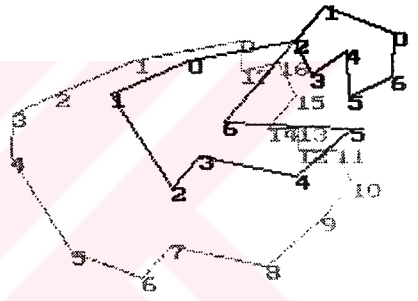
a



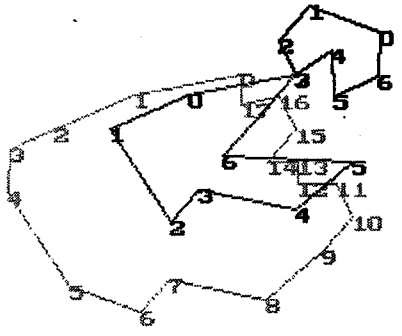
b



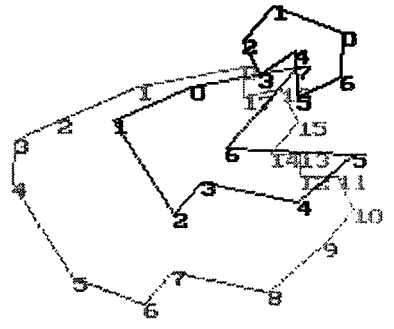
c



d



e



f

Şekil 4.6 Q poligonunun, P⊕(-Q) 'nun çeperi üzerinde dolaştırılması -3

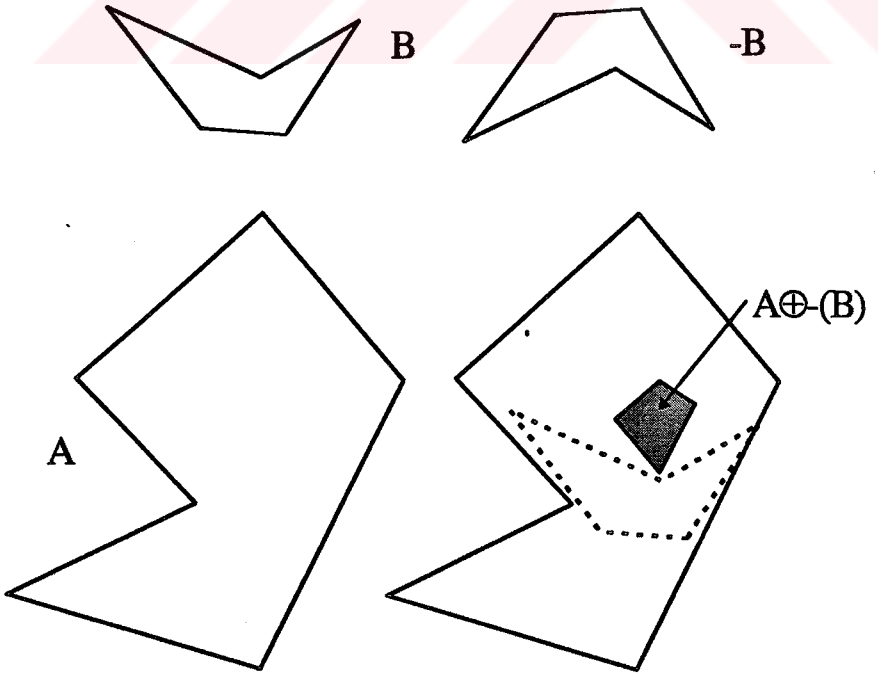
4.2.3.3 Minkowski Farkı

$$A\theta B = \bigcap_{b \in B} (A - b) \qquad (4.5)$$

Teorem : $A\theta B = \overline{\overline{A} \oplus B}$

Teorem : (B+t)⊆A ancak ve ancak t∈Aθ -B≠0 dir. Bu teorem **ıska (kapsama)** dönüşümü olarak adlandırılır.

Iska dönüşümü, pastal yerleştirmede büyük parçalar yerleştirildikten sonra, bu parçaların aralarını küçük parçalarla doldurma sırasında kullanılır: Delik A poligonu, deliğe yerleştirilecek parça da B poligonu olmak üzere B poligonun yerel orijini Aθ -B poligonunun içinde olduğu sürece B poligonu A poligonundan yani delikten dışarı taşmaz.



Şekil 4.7 Kapsama Dönüşümü

4.2.3.4 Minkowski Toplamı Bulma Algoritması

İçbükey poligonların minkowski toplamı delikler içerebilir. Tezde P⊕ -Q ' nin dış çeperi hesaplanmıştır.

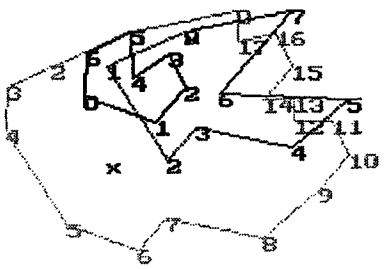
Toplamın dış çeperini bulmak için aşağıdaki metod uygulanır:

(-Q) poligonu, A poligonunun çevresinde saatin ters yönünde süpürülür. Bu işlem sırasında taranan alanın en sağda kalan kısmı P⊕ -Q' nin dış çeperini oluşturur.

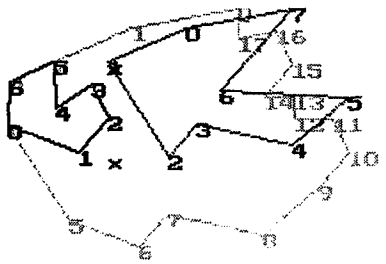
4.2.3.5 Minkowski Toplamının Tezde Uygulanması

- Yerleştirilecek parçanın aynalanmış kopyası ile daha önce yerleştirilmiş parçaların birleşiminin minkowski toplamı alınır;
- Yeni parçanın referans noktasının konumlanmaması gereken yasak bölge belirlenir.
- Toplama işlemi sonucunda oluşan poligonun çeperi yeni poligonun, daha önce yerleşmiş poligonlara değdiği ancak örtüşmediği sınırdır.

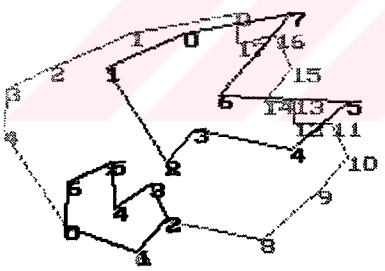
Şekil 4.8 ve Şekil 4.9-4 P⊕ -Q' nin dış çeperini bulma işleminin adımları görüntülenmiştir.



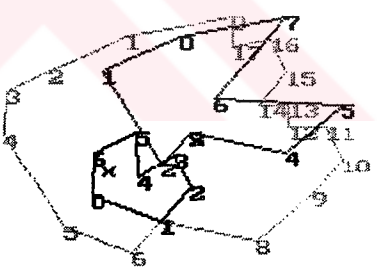
a) Q'nun referansı 0. noktada



b) Q'nun referansı 1. noktada

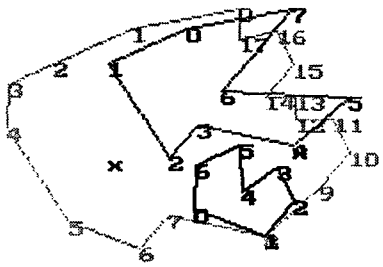


c) Q'nun referansı 2. noktada

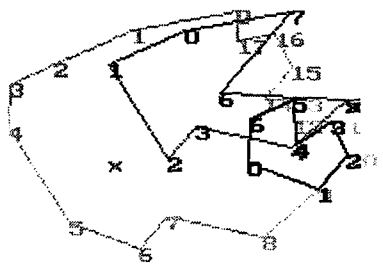


d) Q'nun referansı 3. noktada

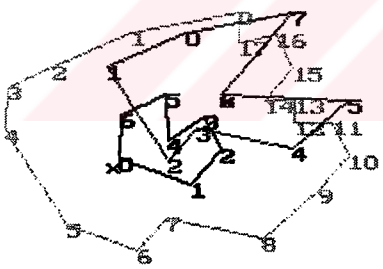
Şekil 4.8 P⊕ -Q' nun dış çeperini bulma işleminin adımları 1



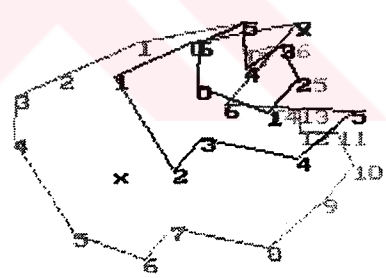
e) B'nin referansı 4. noktada



f) B'nin referansı 5. noktada



g) B'nin referansı 6. noktada



h) B'nin referansı 7. noktada

Şekil 4.9 P⊕ -Q' nun dış çeperini bulma işleminin adımları 2

2.4 Çokgen Birleşimi

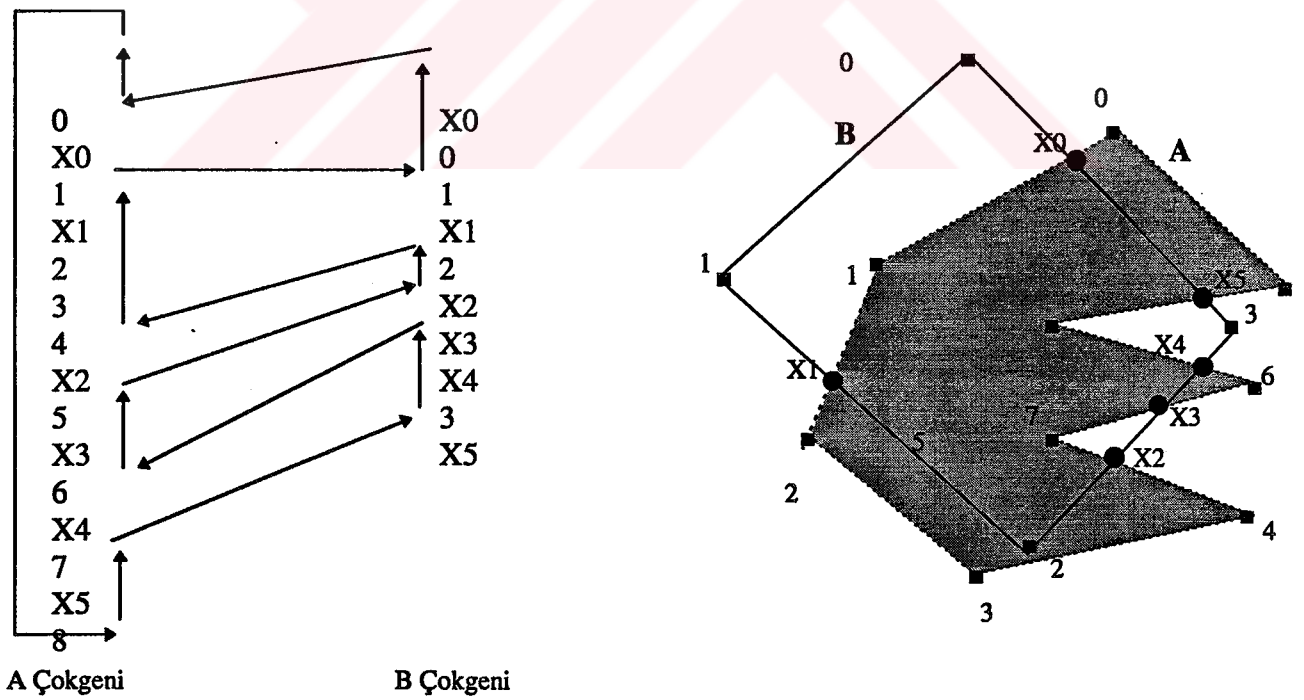
Çokgen birleşimi, Weiler- Attherton kırpma algoritması benzeri bir yaklaşımla gerçeklenmiştir. Bu ıklaşımda çokgenler saatin ters yönünde verilen noktalar ile temsil edilir. Bu durumda poligon içi poligon ·vresi saat tersi yönünde dolaşıldığında izlenen vektörün sağında kalır.

ıUB nin dış çeperini bulmak için aşağıdaki adımlardan geçilir:

ve B çokgenlerinin kesişim noktaları bulunur.

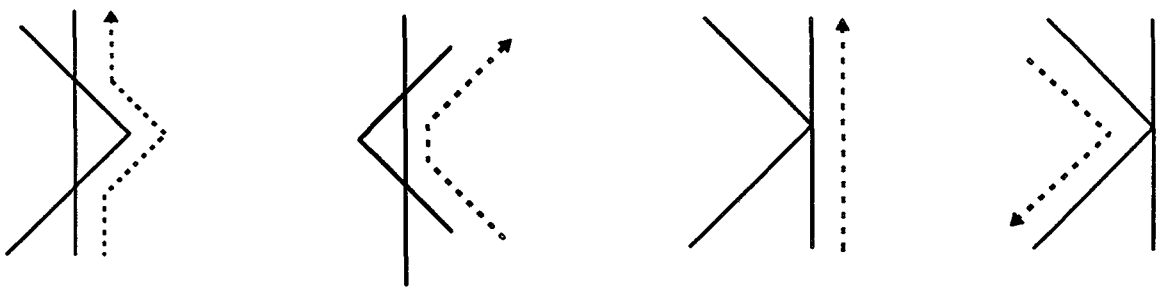
esişim noktası yok ise olası üç durum vardır:

- A'nın bir noktası B çokgeninin içinde ise A çokgeni, B çokgeninin içindedir : $A \subseteq B$.
- B 'nin bir noktası A çokgeninin içinde ise B çokgeni A çokgeninin içindedir : $B \subseteq A$.
- A ve B çokgenleri birbirlerinden ayrıktır.



a b

Şekil 4.10 Poligon Birleşim Örneği



a) kenar - köşe geçişi b) kenar - kenar geçişi c) teğet kenar-köşe

Şekil 4.11 Önemli geçiş Şekilleri

dnewvertex(polygonno,vertexno,A,B,xlist)
)nno : izlenen poligonun numarası
no : bulunulan köşe
)oligonlar
kesişim matrisi

ı poligonun, vertexno nolu köşe ile başlayan kenarı üzerinde kesişim noktası varsa; kesen kenarlardan, vertexno nolu köşeye ın kesişimi veren kenarın başlangıç köşesi geri döndürülür. Kesişim yoksa -1 geri döner.

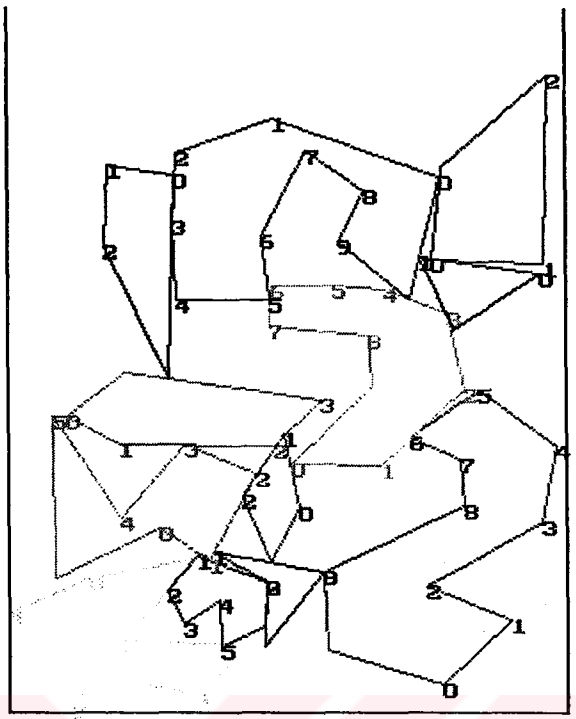
:xtpointofinterval(polygonno,k,vertexno,A,B,xlist,direction)
)nno : izlenen poligonun numarası
en kenarın başlangıç köşesi
no : bulunulan köşe
)oligonlar
kesişim matrisi

ı kenar üzerinede bulunulan kesişim noktasından başka kesişim noktaları da varsa, kesen kenarlardan, bulunulan kesişim ı-kenarın hedef köşesi aralığında bulunulan kesişim noktasına en yakın kesişime ait kesen kenarın başlangıç noktası geri :ülür. Bu aralıkta başka kesen kenar yoksa -1 geri döndürülür.

Şekil 4.12 Poligon Birleşim Algoritması 1

```
LYGONUNION(POLYTYPE A, POLYTYPE B, POLYPTR unionptr)
  0:ayrik poligonlar
  1:A, B nin icinde
  2:B, A nin icinde
  3:kesisiyorlar



- kesisim listesini bul;
rtex = vertexno =  birlesimin uzerinde bir nokta;



 noktayi union listesine ekle;
  union_noktasayısı ++;
  k=findnewvertex(polygonno,vertexno,A,B,xlist);
  if (k!=-1)                                          //kesisim noktasina gelinmis
  {
          while (k!=-1)
          {
             kesisim noktasini union listesine ekle;
             union_noktasayısı++;
             diger poligona gec;
             l=nextpointofinterval(polygonno,k,vertexno,A,B,xlist,direction);
             if (l==-1)                               //koseye gelindi.
                   yeni kenarin hedef noktasini union listesine at;
             k=l;
          }
  }
  else
             poligon uzerinde ilerle;

( !( (polygonno==startpolygon) && (vertexno==startvertex) ) );
                                                  // başlangıç noktasına gelinene kadar
```

Şekil 4.13 Poligon Birleşim Algoritması 2



Şekil 4.14 Örnek Çokgen Yerleşimi

SONUÇLAR ve ÖNERİLER

SONUÇLAR

Bu tez bazı istisnalar dışında (Kantarovitch 1939) 1960 larda Gilmore ve Gomory'nin makaleleri ile üzerinde çalışılmaya başlanan, ancak elde edilen sonuçların doğrudan endüstride kullanım alanı bulması sebebiyle günümüze kadar güncelliğini koruyan bir konuyu ele alır. Pastal yerleştirme NP-zorlukta olmasına rağmen uygun sezgisel yaklaşımlarla, kabul edilebilir zaman ve verimle sonuç elde edilebileceğini gösterir. Bu tezin diğer çalışmalardan en büyük farkı algoritmik geometri ile uğraşan araştırmacıların kullandığı, fakat paketleme-yerleştirme problemi ile uğraşan araştırmacıların kullanmadığı tekniklerle klasik paketleme-yerleştirme tekniklerini birleştirmesidir. Tezin birinci bölümünde kullanılan alt-sol sezgisel yaklaşımının sonuçları genetik algoritmaları kullanılarak 2-3 % mertebesinde iyileştiştirilmiştir. İkinci bölümde doğrudan çokgen yerleştirme problemini çözmek için Minkowski poligon işlemlerinden yaralanılmıştır. İşlemler içbükey çokgenler üzerinde gerçeklenmiştir. İkinci bölümde optimizasyondan çok olurluluk problemi ile ilgilenilmiştir. Ancak hazırlanan, algoritmik geometri tabanlı programlar ve teknikler, farklı sezgisel yaklaşımlar kullanılarak geliştirilmeye uygundur.

ÖNERİLER

Mevcut Uygulamanın Geliştirilmesi:

Kutu paketleme bölümünde bellek ve hız kaygılarından kısıtlanan genetik işlemlerdeki iterasyon sayısı ve toplumdaki birey sayısı, uygun bir sistemde arttırılarak, başarım arttırılabilir. Asıl işlem yükü getiren genetik algoritmanın değerlendirme fonksiyonu hesabı, kaba taneli paralelleştirmeye uygundur; paralelleştirme yoluyla hızlanma sağlanabilir.

İkinci bölümde En yoğun hesap yükünü getiren minkowski toplamındaki süpürme işleminin hızlandırılması, tüm algoritmayı hızlandıracaktır. Algoritmalar kaba taneli paralelleştirilmeye uygundur. Bundan yaralanılarak hızlanma sağlanabilir. Poligon sadeleştirme ile hesap yükü azaltılabilir.

Yeni Uygulama Alanları:

Tanıtılan algoritmalar gerekli uyarlamalar yapılarak, mevcut pastalı sıkıştırma, büyük parçalar yerleştirildikten sonra küçük parçaları aralara yerleştirerek verimi arttırma, iyi yerleştirilmiş bir pastala benzeterek yerleştirme işlerinde kullanılabilir.

Uzaysal boyutu arttırma (3-Boyutlu paketleme, yerleştirme) veya döndürmeyi de hesaba katarak yeni bir serbestlik derecesi katma ile problem boyutu artırılabilir.

Tezin iki bölümünde de iyileştirici çalışmaya gidilebilir ancak en büyük kazanç iki katman arasına yerleştirilecek bir ara katmanla, kutu paketlemeden elde edilen bazı sonuçların, parça sırası gibi, çokgen yerleştimede kullanılması ile sağlanabilir.

KAYNAKLAR

[1] Milenkovic, V., Daniels, K., Li, Zhenyu, Automatic Marker Making, Proceedings of the Third Canadian Conference on Computational Geometry, 1991.

[2] Dyckhoff, Herald, A Typology of Cutting and Packing Problems, European Journal of Operational Research, No.44, pp.145-159, 1990.

[3] Daniels, K., Containment Algorithms For Nonconvex Polygons With Application to Layout, PhD thesis, Harvard University, Division of Applied Sciences, 1995.

[4] Dikili, A. Cemil, Gemi Üretiminde Bilgisayar Destekli Optimum Malzeme Kullanımı, Doktora Tezi, İTÜ Fen Bilimleri Enstitüsü, 1991.

[5] Israni, Sharat S., Sanders, Jerry L., Performance Testing of Rectangular Parts - Nesting Heuristics, Int. J. Prod. Res., Vol.23, No.3, pp.437-456, 1985.

[6] Dowsland, Kathryn A., Dowsland, William B., Packing Problems, European Journal of Operational Research, No.56, pp.2-14, 1992.

[7] Beasley, J.E., An Algorithm For the Two-dimensional Assortment Problem, European Journal of Operational Research, No.19, pp.253-261, 1985.

[8] Waescher, Gerhard, An LP-based Approach To Cutting Stock Problems With Multiple Objectives, European Journal of Operational Research, No.44, pp.175-184, 1990.

[9] Dietrich, Robert D., Yakowitz, Sidney J., A Rule-based Approach To The Trim-loss Problem, Int. J. Prod. Res., Vol.29, No.2, pp.401-405, 1991.

[10] Farley, Alan A., The Cutting Stock Problem in the Canvas Industry, European Journal of Operational Research, No.44, pp.247-255, 1990.

[11] Albano, Orsini, A Heuristic Solution of the Rectangular Cuttinng Stock Problem,The Computer Journal, Vol. 23, No.4, pp 338-343, 1980

[12] Bengtsson, Bengt-Erik, Packing Recatnagular Pieces - A Heuristic Packing, The Computer Journal, Vol.25, No.3, pp.353-357, 1982.

[13] Goldberg, David, Genetic Algorithms in Search, Optimisation And Machine Learning, University of Alabama Press.

[14] Renders, Jean-Michel, Flasse, Stephane, Hybrid Methods Using Genetic Algorithms For Global Optimization, IEEE Trans. on Systems, Man and Cybernetics, Vol.26, No.2, April 1996.

[15] Goodman, Erik D., Tetelbaum, Alexander Y., Kureichik, Victor M., A Genetic Algorithm Approach to Compaction, Bin Packing, and Nesting Problems, Tech. Report, No.940702, Case Center for CAE&M, Michigan State University, Jul. 1994.

[16] Kröger, Schwenderling, Vornberger, Parallel Genetic Packing on Transputers,Editor Joachim Stender, Parallel Genetic Algorithms Theory & Application, IOS Press 1993

[17] Chazelle, B.,The Bottom-Left Bin-Packing Heuristic: An Efficient Implementation, IEEE Transactions on Computers, Vol. C-32, No.8, pp.697-707, Aug. 1983.

[18] Adamowicz, Michael, Albano, Antonio, Nesting Two-dimensional Shapes in Rectangular Modules, Computer Aided Design, Vol.8, No.1, pp.27-33, Jan. 1976.

[19] Albano, Antonio, Sapuppo, Giuseppe, Optimal Allocation of Two-dimensional Irregular Shapes Using Heuristic Search Methods, IEEE Trans. on Systems, Man and Cybernetics, Vol.SMC-10, No.5, pp.242-248, May 1980.

[20] Dori, Ben-Bassat, Efficient Nesting of Congruent Convex Figures, Communication of ACM, Vol 27, No.3, pp 228-235, Mar 1984

ÖZGEÇMİŞ

19 Mart 1970'te İstanbul'da doğan Filiz Bunyak, 1989 yılında Saint-Benoit Lisesinden mezun olmuştur. İ.T.Ü. Elektrik-Elektronik Fakültesi Kontrol ve Bilgisayar Mühendisliği Bölümündeki lisans öğrenimini 1994 yılında tamamlamış ve aynı yıl Fen Bilimleri Enstitüsü, Kontrol ve Bilgisayar Anabilim Dalında yüksek lisans öğrenimine başlamıştır. 1994 yılından bu yana İ.T.Ü. Bilgisayar Bilimleri Anabilim Dalında araştırma görevlisi olarak çalışan Filiz Bunyak, IEEE üyesi ve IEEE İ.T.Ü. Öğrenci Kolu başkanıdır.