T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

OPTICUT:
DESEN MİNİMİZASYONLU TEK BOYUTLU STOK KESME PROBLEMİ İÇİN YENİ BİR SEZGİSEL YAKLAŞIM

DOKTORA TEZİ

Nahsen KAYHAN

Endüstri Mühendisliği Anabilim Dalı

Endüstri Mühendisliği Bilim Dalı

HAZİRAN 2025

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

OPTICUT:
DESEN MİNİMİZASYONLU TEK BOYUTLU STOK KESME PROBLEMİ İÇİN YENİ BİR SEZGİSEL YAKLAŞIM

DOKTORA TEZİ

Nahsen KAYHAN

Endüstri Mühendisliği Anabilim Dalı

Endüstri Mühendisliği Bilim Dalı

Tez Danışmanı: Doç. Dr. Esra TEKEZ

HAZİRAN 2025

Nahsen KAYHAN tarafından hazırlanan "OPTICUT: Desen minimizasyonlu tek boyutlu stok kesme problemi için yeni bir sezgisel yaklaşım" adlı tez çalışması 23.06.2025 tarihinde aşağıdaki jüri tarafından oy birliği ile Sakarya Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalı Endüstri Mühendisliği Bilim Dalı'nda Doktora tezi olarak kabul edilmiştir.

Tez Jürisi

Jüri Başkanı :	**Prof. Dr. Ercan ÖZTEMEL** Marmara Üniversitesi
Jüri Üyesi :	**Doç. Dr. Esra TEKEZ** (Danışman) Sakarya Üniversitesi
Jüri Üyesi :	**Prof. Dr. İhsan Hakan SELVİ** Sakarya Üniversitesi
Jüri Üyesi :	**Prof. Dr. Kasım BAYNAL** Kocaeli Üniversitesi
Jüri Üyesi :	**Dr. Öğr. Üyesi Halil İbrahim DEMİR** Sakarya Üniversitesi

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Sakarya Üniversitesi Fen Bilimleri Enstitüsü Lisansüstü Eğitim-Öğretim Yönetmeliğine ve Yükseköğretim Kurumları Bilimsel Araştırma ve Yayın Etiği Yönergesine uygun olarak hazırlamış olduğum "OPTICUT: Desen minimizasyonlu tek boyutlu stok kesme problemi için yeni bir sezgisel yaklaşım" başlıklı tezin bana ait, özgün bir çalışma olduğunu; çalışmamın tüm aşamalarında yukarıda belirtilen yönetmelik ve yönergeye uygun davrandığımı, tezin içerdiği yenilik ve sonuçları başka bir yerden almadığımı, tezde kullandığım eserleri usulüne göre kaynak olarak gösterdiğimi, bu tezi başka bir bilim kuruluna akademik amaç ve unvan almak amacıyla vermediğimi ve 20.04.2016 tarihli Resmi Gazete'de yayımlanan Lisansüstü Eğitim ve Öğretim Yönetmeliğinin 9/2 ve 22/2 maddeleri gereğince Sakarya Üniversitesi'nin abonesi olduğu intihal yazılım programı kullanılarak Enstitü tarafından belirlenmiş ölçütlere uygun rapor alındığını, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun ortaya çıkması halinde doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi beyan ederim.

(23/06/2025).

Nahsen KAYHAN

Babam ALİ KAYHAN'ın anısına…

TEŞEKKÜR

Tez çalışmam süresince bilgi birikimi ve rehberliğiyle bana yol gösteren değerli tez danışmanım Doç. Dr. Esra Tekez'e içten teşekkürlerimi sunarım.

Tez izleme komitesinde yer alan kıymetli hocalarım Prof. Dr. İhsan Hakan Selvi ve Dr. Öğretim Üyes Halil İbrahim Demir'e, değerli önerileri ve yapıcı eleştirileriyle çalışmamın gelişimine katkıda bulundukları için teşekkür ederim.

İlkokul yıllarımda bana büyük emek veren ve eğitim hayatımın temelini oluşturan sevgili öğretmenim Zeki Altıkulaç'a şükranlarımı ifade ederim.

Hayatımın her anında sevgi, fedakârlık ve destekleriyle yanımda olan değerli anneme, babama ve aileme en içten teşekkürlerimi sunarım.

Nahsen KAYHAN

İÇİNDEKİLER

Sayfa

KISALTMALAR

1DCSP	: Tek Boyutlu Stok Kesme Problemi
1DCSP-S	: Kurulum Maliyetli Tek Boyutlu Stok Kesme Problemi
CG	: Sütun Üretimi
CUI	: Cui ve Ark.'nın Algoritması
CUTGEN1	: Gau ve Wäscher'in Benchmark Veri Seti
ILP	: Tamsayılı Doğrusal Programlama
LP	: Doğrusal Programlama
MSHP	: Modifiye Edilmiş Sıralı Sezgisel Prosedür
OPTICUT	: Desen Minimizasyonlu Tek Boyutlu Stok Kesme Problemi İçin Yeni Bir Sezgisel Yaklaşım
SHP	: Sıralı Sezgisel Prosedür
WOS	: Web of Science
YL	: Yan ve Liu'nun Algoritması

SİMGELER

$\mathbf{D_i}$	: i tipi siparişlerin başlangıç miktar talebi
G	: Mevcut neslin kimliği
I	: Negatif olmayan tam sayılar kümesi
$\mathbf{l_i}$	: i tipi siparişin uzunluğu
M	: Bir modelin maksimum frekans sınırı
m	: Sipariş boyutu sayısı
N	: Q'daki desen sayısı
P	: Mevcut desen
pc	: Öncelik katsayıları seti
$\mathbf{Pr_i}$	: i tipi siparişin önceliği
Q	: Farklı desenlerin kümesi, $Q=\{Q_1,...,Q_N\}$
$\mathbf{Q_{(iJ)}}$	: Qj desenindeki i tipi öğelerin miktarı
$\mathbf{R_i}$	: i tipi siparişler için kalan miktar talebi
SL	: Stok uzunluğu
$\mathbf{u_B}$	: Stok başına maliyet
$\mathbf{u_S}$	: Kurulum başına maliyet
$\mathbf{V_i}$	: i tipi siparişin değeri
$\mathbf{X_i}$	: Mevcut desendeki i tipi siparişler için kullanım miktarı
$\mathbf{X_J}$	: Desenin frekansı Q_j
Z	: Mevcut kesim planının maliyeti
$\mathbf{Z_{en\ iyi}}$	: En iyi kesim planının maliyeti
$\mathbf{Z_{ILP}}$	: Amaç değeri
$\boldsymbol{\varepsilon_J}$	: $Q_{(j)}$nin kullanılıp kullanılmadığını ($\varepsilon_{(j)}=1$) veya kullanılmadığını ($\varepsilon_{(j)}=0$) gösteren ikili değişken

TABLO LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

OPTICUT: DESEN MİNİMİZASYONLU TEK BOYUTLU STOK KESME PROBLEMİ İÇİN YENİ BİR SEZGİSEL YAKLAŞIM

ÖZET

Tek boyutlu stok kesme problemi (1DCSP), kâğıt, çelik, ambalaj ve plastik film gibi endüstriyel sektörlerde sıkça karşılaşılan bir kombinatoryal optimizasyon problemidir. Bu problemin temel amacı, büyük stok malzemelerini belirli talepleri karşılayacak şekilde daha küçük parçalara keserek fire miktarını ve üretim maliyetlerini en aza indirmektir. Gerçek dünya uygulamalarında, fire miktarının yanı sıra kesim desenleri arasındaki geçişlerden kaynaklanan kurulum maliyetleri gibi ek faktörler problemi karmaşık hale getirir.

Tek boyutlu stok kesme probleminin desen minimizasyonlu varyantı, fire miktarını azaltırken aynı zamanda oluşturulan kesim desenlerinin sayısını en aza indirmeyi hedefler. Bu, malzeme kullanımını optimize ederken üretim süreçlerinde karmaşıklığı ve maliyeti düşürmek için bir denge kurmayı amaçlar. Daha az desen kullanımı, genellikle kurulum maliyetlerini azaltır ve operasyonel verimliliği artırır.

NP-tam olarak sınıflandırılan bu problem, boyut arttıkça hesaplama karmaşıklığı üstel olarak yükseldiğinden, kabul edilebilir sürelerde pratik çözümler üretmek için sezgisel yaklaşımlara gereksinim duyar.

Bu tezde, tek boyutlu stok kesme probleminin desen minimizasyonlu varyantını çözmek için OPTICUT adında yeni bir sezgisel algoritma geliştirilmiştir. OPTICUT, toplam maliyeti ve desen sayısını azaltmada mevcut en gelişmiş yöntemleri geride bırakarak, akademik araştırmalar ve endüstriyel uygulamalar için güçlü bir çözüm sunmaktadır. Bu tez, kapsamlı literatür taraması, detaylı hesaplama deneyleri ve duyarlılık analizleriyle algoritmanın etkinliğini kanıtlamaktadır.

Bu tezde, 1DCSP alanındaki literatür, Web of Science veri tabanında 1983-2025 yılları arasında yayımlanan 169 çalışma incelenerek detaylı olarak analiz edilmiştir. Bu çalışmalar toplam 2,695 atıf almış olup, malzeme fire minimizasyonunun (%42,28) en yaygın araştırma hedefi olduğu belirlenmiştir. Maliyet optimizasyonu (%12.08) ikinci sırada yer alırken, desen minimizasyonu daha az çalışılmış ancak pratik açıdan kritik bir konu olarak öne çıkmaktadır. Mevcut çözüm yöntemleri arasında sezgisel yaklaşımlar (örneğin, sıralı sezgisel prosedür), meta-sezgisel yöntemler (örneğin, genetik algoritmalar) ve kesin yöntemler (örneğin, tamsayılı doğrusal programlama) bulunmaktadır. Ancak bu yöntemler, genellikle fire ve desen kullanımını aynı anda optimize etmekte zorlanmakta ve özellikle büyük ölçekli veya karmaşık problemlerde yetersiz kalmaktadır. Bu durum, OPTICUT'ın gidermeyi hedeflediği bir açıktır.OPTICUT, 1DCSP'yi desen minimizasyonuyla çözmek için üç aşamalı bir hibrit yaklaşım benimsemektedir:

Başlangıç çözüm üretimi (sütun üretimi yaklaşımı ile): Gilmore ve Gomory'nin (1963) yönteminden esinlenerek, bu aşamada sütun üretimiyle başlangıçta uygulanabilir bir çözüm oluşturulur. Bu yöntem, fire minimizasyonunda genellikle etkili olsa da çok sayıda kesim deseni üretebildiğinden sonraki aşamalarda iyileştirme gerektirir.

Alternatif çözümler üretimi (desen üretimi yaklaşımı ile): İki farklı algoritma, Algoritma A ve Algoritma B, alternatif kesim desenleri havuzunu oluşturur. Algoritma A, siparişleri uzunluk ve artık değerlerine göre önceliklendirerek 20 yineleme boyunca değişen öncelik katsayılarıyla çalışır. Algoritma B ise 50 yineleme boyunca izin verilen fire miktarını kademeli olarak artırarak desen çeşitliliğini sağlar. Bu mekanizmalar, çözüm uzayını kapsamlı bir şekilde keşfetmeyi ve yüksek kaliteli desenler üretmeyi mümkün kılar.

Nihai optimizasyon (tamsayılı doğrusal programlama yaklaşımı ile): Önceki aşamalardan elde edilen desen havuzu, 45 saniyelik bir süre sınırı içinde tamsayılı doğrusal programlama (ILP) ile optimize edilir ve stok maliyetleri ile kurulum maliyetlerini içeren toplam maliyeti en düşük olan çözüm seçilir.

Algoritma sipariş değeri (V_i), sipariş önceliği (Pr_i) ve izin verilen fire artışı gibi dinamik parametreleri kullanarak, zorlu problem örneklerine uygun yüksek kaliteli kesim desenleri üretir. Uygulamanın ayrıntıları, sözde kod ve akış şemalarıyla sunulmuş, böylece şeffaflık ve tekrarlanabilirlik sağlanmıştır.

OPTICUT'ın performansı, CUTGEN1 (Gau & Wäscher, 1995) ve Umetani (2003, 2018) veri setlerinden alınan 1.840 benchmark örneğiyle test edilmiştir. Deneyler, Intel Core i7-1165G7 işlemci ve 16 GB RAM'e sahip bir bilgisayarda, Python ile PULP-CBC-CMD çözücüsü kullanılarak gerçekleştirilmiş olup örnek başına ortalama hesaplama süresi 165 saniye olarak kaydedilmiştir.

CUTGEN1 sonuçları: Sipariş çeşitliliği ve stok uzunluk oranlarına göre 18 problem sınıfında gruplandırılan 1.800 test örneğinde, OPTICUT toplam maliyeti %0.0013 ve desen sayısını %0.6 oranında azaltmıştır. En iyi mevcut algoritmalarla (CUI, MSHP, YL) karşılaştırıldığında, stok kullanımında %83, desen sayısında %50 ve toplam maliyette %67 oranında baskınlık elde ederek genel verimlilikte üstünlük sağlamıştır.

Umetani sonuçları: Kimyasal elyaf endüstrisinden alınan 40 gerçek dünya örneğinde, OPTICUT üstün performans göstermiştir. Stok maliyetinin stok uzunluğuna bağlı olduğu ve kurulum maliyetinin 100 olarak sabitlendiği varsayımında, MSHP ve CUI'ye kıyasla maliyette %0.51'e, desen sayısında ise %0.82'ye varan azalmalar elde edilmiştir.

'Fiber29_9080' örneği: OPTICUT, 35 stok rulosunu 6 desen ve 550 mm fire ile kullanırken, CUI aynı fire seviyesinde 7 desen kullanmıştır. Bu sonuç da OPTICUT'ın desen minimizasyon yetkinliğini açıkça ortaya koymaktadır.

150 CUTGEN1 örneği üzerinde gerçekleştirilen duyarlılık analizi, OPTICUT'ın parametre değişimlerine (V_i, Pr_i, fire artışı) karşı duyarlılığını incelemiştir. Sonuçlar, stok kullanımının tüm denemelerde sabit kaldığını, desen sayısının ise parametre aralıkları genişledikçe azaldığını ortaya koymuş; ancak bu etkinin minimal düzeyde olduğu belirlenmiştir. Bu tutarlılık, OPTICUT'ın farklı problem koşullarında güvenilirliğini artırmaktadır. Her ek deneme, hesaplama süresine yaklaşık 0,5 saniye eklemektedir.

OPTICUT, literatüre aşağıdaki temel katkıları sunmaktadır:

- Üstün performans: Stok uzunluğuna göre küçük ve büyük sipariş türlerini içeren çeşitli senaryolarda (CUTGEN1 grup 1 ve grup 3) ile gerçek dünya örneklerinde, mevcut yöntemlerden daha iyi sonuçlar elde etmiştir.

- Maliyet ve desen azaltımı: Malzeme maliyetinin yanı sıra toplam maliyeti ve desen sayısını ölçülebilir oranda azaltarak operasyonel verimliliği artırmaktadır.
- Esneklik: Kullanıcıların malzeme ve kurulum maliyetlerini ile deneme sayısını ayarlayarak ihtiyaçlarına göre optimize edilmiş farklı sonuçlar elde etmelerine olanak tanımaktadır.
- Yenilikçi özellikler: Dinamik değişkenlerin kullanımı ve hibrit yaklaşımıyla, literatürdeki çözümleri aşan yüksek kaliteli çözümler üretmektedir.
- Pratik uygulanabilirlik: Makul hesaplama süreleriyle gerçek dünya kesim operasyonlarında kullanılabilir bir çözüm sunmaktadır.

OPTICUT için aşağıdaki geliştirmeler önerilmektedir:

- Ticari çözücü entegrasyonu: Gurobi gibi bir çözücünün entegrasyonu, ILP çözüm süresini ve kalitesini iyileştirebilir.
- Paralel hesaplama: Sütün üretme Algoritması, Algoritma A ve B'nin eşzamanlı çalışması, işlem süresini %50'ye kadar azaltabilir.
- Çözüm sonrası desen azaltma: Çözüm sonrası uygulanacak ek algoritmalar, fire seviyesini koruyarak desen sayısını daha da düşürebilir.
- Çoklu stok uzunlukları: Algoritmanın farklı stok uzunluklarını işleyebilmesi, uygulanabilirliğini artıracaktır.

OPTICUT, desen minimizasyon varyantlı 1DCSP'yi çözmek için yenilikçi ve etkili bir yöntem sunmaktadır. Ampirik veriler, algoritmanın farklı senaryolarda mevcut yöntemlerden üstün performans sergilediğini göstermektedir. Sonuç olarak OPTICUT, maliyet ve desen minimizasyonunu birleştirerek kaynak verimliliği ve maliyet tasarrufu açısından önemli bir ilerleme sağlamaktadır.

OPTICUT: A NEW HEURISTIC ALGORITHM FOR THE ONE-DIMENSIONAL CUTTING STOCK PROBLEM WITH PATTERN MINIMISATION

SUMMARY

The one-dimensional cutting stock problem (1DCSP) is a well-known combinatorial optimization challenge encountered across various industrial sectors, including paper, steel, and plastic film production. The primary objective of 1DCSP is to efficiently cut larger stock materials into smaller pieces to meet specific demands while minimizing waste and production costs. In real-world applications, additional considerations such as setup costs—arising from transitions between cutting patterns—further complicate the problem. The variant addressed in this dissertation, known as the 1DCSP with pattern minimization, seeks to balance waste reduction with the minimization of the number of distinct cutting patterns, as fewer patterns typically reduce setup costs and improve operational efficiency. Classified as NP-hard, this problem exhibits exponential computational complexity as its size increases, necessitating heuristic approaches to achieve practical solutions within acceptable timeframes.

This dissertation introduces OPTICUT, a novel heuristic algorithm designed to address the 1DCSP with pattern minimization. OPTICUT aims to outperform existing state-of-the-art methods by reducing both total costs and the number of cutting patterns, offering a robust and efficient solution for both academic research and industrial applications. The study builds on an extensive literature review, comprehensive computational experiments, and sensitivity analyses to validate the algorithm's effectiveness.

The dissertation provides a thorough review of the 1DCSP literature, analyzing 169 publications from the Web of Science database spanning 1983 to 2025, with a total of 2,695 citations. The analysis reveals that material waste minimization is the most common research objective (42.28%), followed by cost optimization (12.08%). Pattern minimization, though less frequently studied, is critical in practical settings due to its impact on setup costs. Existing solution methods include heuristic approaches (e.g., Haessler's Sequential Heuristic Procedure, 1975), metaheuristics (e.g., genetic algorithms), and exact methods (e.g., integer linear programming, ILP). However, these approaches often struggle to simultaneously optimize waste and pattern usage, particularly in large-scale or complex scenarios, highlighting a gap that OPTICUT seeks to address.

OPTICUT employs a three-stage hybrid methodology to tackle the 1DCSP with pattern minimization:

Initial solution generation (using column generation approach): Drawing from Gilmore and Gomory (1963), this stage uses column generation to produce an initial feasible solution. While mostly effective at minimizing waste, it often generates a large number of patterns, necessitating further refinement.

Alternative solutions generation (using pattern generation approach): Two distinct algorithms, Algorithm A and Algorithm B, create a pool of alternative cutting patterns. Algorithm A prioritizes items based on their length and residual value, iterating over 20 generations with varying priority coefficients. Algorithm B adjusts allowable waste incrementally across 50 iterations, ensuring diversity in pattern selection. These mechanisms enhance the exploration of the solution space.

Final optimization (using integer linear programming approach): The pattern pool from the previous stages is optimized using ILP within a 45-second time limit, selecting the solution with the lowest total cost, which includes stock costs, setup costs, and waste-related expenses.

The algorithm incorporates dynamic parameters—such as item value V(i), priority Pr(i), and allowable waste increase to generate high-quality cutting patterns tailored to challenging problem instances. pseudocode and flowcharts detail the implementation, ensuring transparency and reproducibility.

OPTICUT's performance was rigorously evaluated using 1,840 benchmark instances from two datasets: CUTGEN1 (Gau & Wäscher, 1995) and Umetani (2003, 2018). Experiments were conducted on an Intel Core i7-1165G7 processor with 16 GB RAM, implemented in Python using the PULP-CBC-CMD solver, with an average computation time of 165 seconds per instance.

CUTGEN1 results: Across 18 problem classes grouped by order variety and stock length ratios, OPTICUT reduced total costs by 0.0013% and pattern usage by 0.6% compared to state-of-the-art algorithms (e.g., CUI, MSHP, YL). It achieved dominance in 83% of classes for stock usage, 50% for pattern count, and 67% for total cost, outperforming competitors in overall efficiency.

Umetani results: In 40 real-world fiber cutting instances, OPTICUT demonstrated superior performance, particularly with large stock lengths (e.g., 9080 mm). For stock cost = stock length and setup cost = 100, it achieved up to 0.51% cost reduction and 0.82% pattern reduction compared to MSHP and CUI, excelling in scenarios with diverse item sizes.

An illustrative example, "Fiber29_9080," showed OPTICUT utilizing 35 stock rolls with 6 patterns and 550 mm waste, compared to CUI's 7 patterns for the same waste level, underscoring its pattern minimization capability.

A sensitivity analysis on 150 CUTGEN1 instances assessed OPTICUT's robustness against parameter variations (V_i, Pr_i, and waste increase). Results indicated consistent stock usage across trials, with pattern count decreasing as parameter ranges widened, albeit with minimal impact. This stability enhances OPTICUT's reliability across diverse problem configurations, with computation time increasing by approximately 0.5 seconds per additional trial.

OPTICUT offers several key contributions:

Superior performance: It outperforms existing methods in diverse scenarios, including small and large item types relative to stock length, as evidenced by CUTGEN1 groups 1 and 3, and real-world fiber examples.

Cost and pattern reduction: The algorithm achieves measurable reductions in total cost and pattern count, enhancing operational efficiency.

Flexibility: Users can adjust stock and setup costs, as well as trial numbers, for tailored optimization.

Innovative features: Dynamic variables and a hybrid approach enable high-quality pattern generation, surpassing documented solutions in the literature.

Practical applicability: With reasonable computation times, OPTICUT is viable for real-world cutting operations.

Future enhancements could include:

Commercial solver integration: Incorporating solvers like Gurobi could reduce ILP computation time and improve solution quality.

Parallel computing: Simultaneous execution of Algorithm A and B could cut processing time by up to 50%.

Post-Pattern reduction: Additional algorithms could further minimize patterns without increasing waste.

Multiple stock lengths: Extending OPTICUT to handle varied stock lengths would broaden its applicability.

This dissertation demonstrates that OPTICUT is an effective and innovative solution for the 1DCSP with pattern minimization, validated by empirical results showing significant advantages over existing methodologies.

OPTICUT represents a significant advancement in the field of 1DCSP, bridging theoretical insights with practical utility. By addressing both cost and pattern minimization, it offers a promising framework for future research and industrial adoption, contributing to resource efficiency and cost savings in cutting stock operations.

1. GİRİŞ

Tek boyutlu stok kesme problemi (1DCSP), kâğıt, ambalaj, metal, mobilya, plastik film ve çelik gibi çeşitli endüstriyel sektörlerde karşılaşılan önemli bir kombinatoryal optimizasyon problemidir. Bu sektörler, talep belirsizliklerini yönetmek ve depolama ile taşıma maliyetlerini azaltmak için hammaddeleri büyük birimler halinde stoklar. Stok kesme probleminin temel amacı, büyük stok malzemelerini talepleri karşılayacak şekilde daha küçük parçalara verimli bir şekilde bölerek maliyetleri en aza indirmektir.

Kesim planları, kullanılan kesim desenlerini ve bunların frekanslarını içerir. Bu bağlamda kesim deseni, büyük bir stok malzemesinden kesilen daha küçük parçaların belirli bir düzenini veya kombinasyonunu ifade eder. Her desen, bir büyük parçadan elde edilen farklı türdeki küçük parçaların miktarlarını ve dizilimini tanımlar.

Bir kesim planını değerlendirirken fire maliyetleri önemli bir faktör olsa da gerçek dünya senaryolarında kurulum maliyetleri de büyük bir öneme sahiptir. Bu maliyetler, kesim desenleri arasındaki geçişlerden kaynaklanmakta ve üretim hattının ayarlamalar veya temizlik için durmasını gerektirmektedir (Haessler, 1975). Örneğin, kâğıt endüstrisinde kesim bıçaklarının yeniden konumlandırılması, her desen için farklı kurulumlar gerektirdiğinden üretim kapasitesini azaltmakta; bu da iş gücü ve enerji maliyetlerini artırmaktadır. Stok kesme probleminin kurulum maliyetlerini de en aza indirmeyi hedefleyen bir varyantı, kurulum maliyetli tek boyutlu stok kesme problemi (1DCSP-S) olarak adlandırılır. Kurulum maliyetlerinin desen değişiklikleri arasında sabit olduğu varsayıldığında, 1DCSP-S, desen minimizasyonlu 1DCSP'ye indirgenmektedir.

Desen minimizasyonlu tek boyutlu stok kesme problemi, gerçek hayatta karşılaşılan önemli bir optimizasyon problemidir. Literatürdeki gerçek hayat optimizasyon uygulamalarına örnek olarak, G. Sun ve ark. (2020) ile Chen Zheng ve ark. (2022) tarafından gerçekleştirilen çalışmalar gösterilebilir.

Desen minimizasyonlu 1DCSP, fire miktarını azaltırken kullanılan kesim desenlerinin sayısını en aza indirmeyi amaçlayarak bu iki hedef arasında bir denge kurar. Çoğu zaman, firenin azaltılması desen sayısını artırarak bir çatışma yaratır (Aliano Filho ve

ark., 2018). Bu nedenle, fireyi en aza indirmek ve makine ayarlamaları sırasında üretim süresinden tasarruf sağlamak amacıyla kesim desenlerinin sayısını azaltmak gibi her iki hedefi de optimize etmek büyük önem taşır.

Desen minimizasyonlu 1DCSP, NP-tam olarak sınıflandırılır ve optimal bir çözüm bulmanın hesaplama karmaşıklığı, problem boyutuyla üstel olarak artar (McDiarmid, 1999). Problemin doğal zorluğu, genellikle üretilen fire miktarıyla ilişkilidir. Desen minimizasyonlu 1DCSP'nin 'zor' örnekleri, tipik olarak müşteri tarafından sipariş edilen ürünlerin boyutlarının hammaddenin boyutuna göre büyük olduğu durumlarda ortaya çıkar. Bu senaryo, bir kavanozu büyük çakıl taşlarıyla doldurmaya benzer ve kaçınılmaz olarak önemli miktarda fire oluşmasına neden olur. Böyle durumlarda, hızlı hesaplanabilen sezgisel çözümler, genellikle optimal çözümden belirgin bir sapma gösterir. Buna karşılık, problemin 'kolay' örnekleri, müşteri tarafından sipariş edilen ürünlerin hammaddenin boyutuna göre küçük olduğu ve kavanozu ince kumla doldurmaya benzediği durumlarla karakterize edilir. Bu tür senaryolar genellikle minimum fire ile sonuçlanır ve sezgisel çözümler, optimal çözüme daha yüksek doğrulukla yaklaşabilir (Goulimis, 1990).

Bu doktora tezi, tek boyutlu stok kesme probleminin (1DCSP) desen minimizasyonlu varyantını inceleyerek, endüstriyel uygulamalarda fire miktarını ve kurulum maliyetlerini eş zamanlı olarak en aza indirmeyi amaçlamaktadır. Bu kapsamda, yeni bir sezgisel algoritma geliştirilerek mevcut literatürdeki yöntemlerin sınırlamalarını aşmak ve hem toplam maliyeti hem de kullanılan kesim desenlerinin sayısını azaltan etkili bir çözüm sunmak amaçlanmıştır. Çalışma, teorik bir katkı sağlamanın yanı sıra, kâğıt, çelik ve plastik film gibi sektörlerde pratik uygulanabilirliği olan bir optimizasyon aracı ortaya koyarak kaynak verimliliğini artırmayı ve operasyonel süreçleri iyileştirmeyi amaçlamaktadır.

Bu tez, hem yüksek fireli ('zor') hem de düşük fireli ('kolay') senaryolarda desen minimizasyonlu 1DCSP'yi etkinlikle çözmek için tasarlanmış yenilikçi bir yaklaşım olan OPTICUT sezgisel algoritmasını sunmaktadır.

Desen minimizasyonlu tek boyutlu stok kesme problemleri alanında son gelişmeler sınırlı kalsa da OPTICUT algoritması literatürdeki önemli bir boşluğu doldurarak dikkat çekici bir ilerleme sunmaktadır. Tezin temel katkıları aşağıda özetlenmektedir:

- Yeni bir sezgisel algoritmanın geliştirilmesi: Yüksek kaliteli kesim desenleri üreten, desen tabanlı bir sezgisel algoritma sunulmuştur. Bu algoritma, her nesil için farklı değerleri, öncelikleri ve fire toleranslarını dikkate alarak kesim sürecinin verimliliğini ve etkinliğini artırmaktadır.
- Yenilikçi üç aşamalı hibrid çözüm yaklaşımı: Algoritma, başlangıç çözümü oluşturmak için sütun üretimi, alternatif çözümler ve yüksek kaliteli desenler geliştirmek için üç farklı desen üretim yöntemi ile nihai optimizasyon için tamsayılı doğrusal programlamayı (ILP) entegre eden bir hibrit yaklaşım benimsemektedir. Literatür taramalarımıza göre, bu kadar çeşitli hibrit yöntemleri bir arada başarıyla kullanan başka bir çalışma bulunmamaktadır.
- Kapsamlı karşılaştırma değerlendirmesi ve üstün performans: OPTICUT algoritması, 1.840 benchmark örneği kullanılarak en gelişmiş algoritmalarla kapsamlı olarak karşılaştırılmıştır. Sonuçlar, OPTICUT'ın üstün performansını ortaya koymuş; toplam maliyeti %0,0013 oranında azalttığını ve desen sayısını %0,6 oranında en aza indirdiğini göstermiştir. Bu bulgular, kesim süreçlerini optimize etmedeki etkinliğini vurgulamaktadır.
- Gerçek hayat problemlerine uygulanabilirlik: Algoritma gerçek hayat problemleri için de makül sürelerde literatürdeki algoritmalardan ve mevcut paket yazılımlardan daha iyi çözümler bulmaktadır.

1.1. Tezin Yapısı

Bu tez şu şekilde yapılandırılmıştır: Birinci bölümde, tek boyutlu stok kesme problemi (1DCSP) tanımlanmış, önemi ve desen minimizasyonuyla ilgili temel kavramlar açıklanarak çalışmanın kapsamı ve amacı ortaya koyulmuştur. İkinci bölümde, literatürdeki ilgili çalışmalar sistematik bir şekilde incelenmiş; mevcut yöntemlerin güçlü ve zayıf yönleri değerlendirilerek OPTICUT'ın gidermeyi hedeflediği eksiklikler ve alana katkı potansiyeli belirlenmiştir. Üçüncü bölümde, OPTICUT algoritmasının ayrıntılı yapısı, üç aşamalı metodolojisi (sütun üretimi, desen üretimi ve tamsayılı doğrusal programlama) ve kullanılan matematiksel modeller açıklanmıştır. Dördüncü bölümde, algoritmanın performansı, CUTGEN1 ve Umetani veri setlerinden alınan 1.840 benchmark örneğiyle test edilmiş; hesaplama deneyleri sunulmuş, üç örnek uygulama ile sonuçlar detaylandırılmış ve diğer algoritmalarla karşılaştırmalı analizler gerçekleştirilmiştir. Son olarak, beşinci bölümde çalışmanın

bulguları özetlenmiş, OPTICUT'ın katkıları değerlendirilmiş ve gelecekteki çalışmalar için potansiyel geliştirme alanları tartışılmıştır.

1.2. Kavramlar ve Tanımlar

Tek boyutlu stok kesme probleminde sık kullanılan temel kavramlar aşağıda ayrıntılı olarak açıklanmıştır:

- Tek boyutlu stok kesme problemi (1DCSP): Belirli talepleri karşılamak amacıyla büyük stok malzemelerinin daha küçük parçalara verimli bölünmesini hedefleyen bir kombinatoryal optimizasyon problemidir.
- Sezgisel algoritma: Optimal çözümü garanti etmeyen, ancak kabul edilebilir bir sürede iyi bir çözüm üretmeyi amaçlayan problem çözme yaklaşımıdır.
- Optimizasyon: Belirli kısıtlamalar altında en iyi (maksimum veya minimum) sonucu elde etme sürecidir.
- Stok: Kesim işlemi için kullanılan standart boyutlardaki ham maddelerdir. Örnek: 6 metre uzunluğunda çelik çubuklar, kesim işlemi için stok olarak kullanılabilir.
- Sipariş: Müşterinin talep ettiği belirli boyutlardaki ürünlerin miktarını ifade eder. Örnek: Bir müşteri, 2 metre uzunluğunda 5 adet ve 1 metre uzunluğunda 3 adet çelik çubuk sipariş edebilir.
- Kesim: Stok malzemelerinin sipariş edilen boyutlara uygun olarak bölünmesi işlemidir. Örnek: 6 metre uzunluğundaki bir çubuk, 2 metre ve 1 metre boyutlarında parçalara kesilebilir.
- Fire: Kesim işlemi sonrasında kullanılmayan ve değerlendirilemeyen malzeme parçalarıdır. Örnek: 6 metre uzunluğundaki bir çubuktan 3 metre ve 2 metre kesildiğinde, geriye kalan 1 metre kullanılmazsa bu fire olur.
- Artık: Kesim işlemi sonrasında kalan ve ileride başka siparişlerde kullanılabilecek malzeme parçalarıdır. Örnek: 6 metre uzunluğundaki bir çubuktan 2 metre ve 3 metre kesildiğinde, geriye kalan 1 metre başka bir siparişte kullanılabilir.

- Desen: Stok malzemesinin siparişleri karşılamak için nasıl kesileceğini gösteren bir kesim planıdır. Örnek: 6 metre uzunluğundaki bir çubuk, "2+2+1+1" veya "2+1+1+1+1" gibi desenlerle kesilebilir.
- Kurulum maliyeti: Kesim işlemi için kullanılan makine veya ekipmanın her bir kesim deseni için hazırlanması sırasında ortaya çıkan maliyettir. Örnek: Bir çelik kesim makinesinin her yeni desen için ayarlanması, 10.000 TL'lik bir kurulum maliyeti gerektirebilir.
- Toplam maliyet: Kesim işlemi sırasında oluşan tüm maliyetlerin (malzeme maliyeti, kurulum maliyeti) toplamıdır. Örnek: 6 metre uzunluğundaki çubukların maliyeti 5,000 TL, kurulum maliyeti 1000 TL ise toplam maliyet 6,000 TL olur.
- Desen minimizasyonu: Kullanılan farklı kesim desenlerinin sayısını en aza indirme hedefidir; genellikle kurulum maliyetlerini azaltmak amacıyla uygulanır.
- Tamsayılı doğrusal programlama (ILP): Değişkenlerin tamsayı değerler alması gereken bir doğrusal optimizasyon problemi türüdür.
- Sütun üretimi: Doğrusal programlama problemlerini çözmek için kullanılan bir yöntemdir; değişkenler (sütunlar) başlangıçta açıkça tanımlanmaz, ihtiyaç duyuldukça üretilir.
- NP-Tam: Çözüm süresi problem boyutuyla üstel olarak artan ve çözülmesi zor olarak kabul edilen bir problem sınıfıdır.

1.3. Kesme ve Yerleştirme Problemlerinin Sınıflandırılması

Kesme ve yerleştirme problemleri, çok sayıda sektörde yaygın olarak kullanılan ve farklı varyasyonlarıyla dikkat çeken kombinasyonel optimizasyon problemidir. Bu problemlerin çözümüne yönelik yaklaşımlar, net bir sınıflandırma gereksinimi doğurmuştur. Bu ihtiyacı karşılamak amacıyla Dyckhoff (1990), sistematik bir tipoloji önererek literatürdeki terminoloji karmaşasını düzenlemeyi ve benzer yapılı problemleri gruplandırmayı hedeflemiştir. Dyckhoff'un şeması, problemlerin temel özelliklerini ve çözüm yöntemlerini belirli kriterler çerçevesinde kategorize eder. Bu tür problemler, yalnızca stok kesme ve fire yönetimi ile sınırlı kalmayıp, matemfire, operasyon araştırmaları, lojistik, üretim ve ekonomi gibi disiplinlerde de

benzer mantıksal yapılar sergiler. Sınıflandırma, bu geniş kapsamlı problemlerin çeşitliliğini sistematik olarak ele alır ve ortak yöntemlerle çözülebilecek problemleri bir araya getirir. Böylece, kesme ve yerleştirme problemleri hem kuramsal hem de uygulamalı olarak daha anlaşılır ve çözülebilir hale gelir. Literatürde yer alan bu problemler Tablo 1.1'de sıralanmıştır:

Tablo 1.1. Kesme ve yerleştirme problem türleri.

Problem Türü	**Açıklama**
Stok kesme ve fire problemleri	Büyük nesnelerin kesilerek küçük nesnelere dönüştürülmesi ve fire miktarının minimize edilmesi.
Kutu yerleştirme problemleri	Belirli bir alana kutuların en verimli yerleştirilmesi.
Çift yönlü yerleştirme problemleri	İki yönlü yerleştirme gerektiren durumlar için optimizasyon.
Şerit yerleştirme problemleri	Şerit şeklindeki nesnelerin belirli bir alana yerleştirilmesi.
Vektör yerleştirme problemleri	Çok boyutlu vektörlerin belirli kısıtlar altında yerleştirilmesi.
Sırt çantası problemleri	Belirli bir kapasiteye sahip sırt çantasına en uygun nesnelerin seçilmesi.
Araç yükleme problemleri	Araç, palet, konteyner yükleme ve araç içi yerleşim optimizasyonu.
Çeşit, tükenme, tasarım problemleri	Üretim ve tasarım süreçlerinde karşılaşılan optimizasyon problemleri.
Bölme, düzenleme, iç içe yerleştirme ve parçalama	Üretim süreçlerinde malzeme kullanımı ve yerleşim planlaması.
Sermaye bütçelemesi problemleri	Finansal kaynakların en iyi tahsis edilmesi.
Hat dengeleme problemleri	Üretim hattının verimli çalışması için iş yükünün dengelenmesi.
Bellek tahsisi problemleri	Bilgisayar sistemlerinde bellek yönetiminin optimize edilmesi.
Çok işlemcili zamanlama problemleri	İş yükünün birden fazla işlemciye optimal olarak dağıtılması.

Sınıflandırma, aşağıdaki temel unsurlara dayanır:

1.3.1. Boyutsallık

Boyutsallık, problemin geometrisini tanımlamak için gereken minimum boyut sayısını ifade eder. Bu özellik, problemin çözümünde kullanılan yöntemleri belirlemede kritik bir rol oynar. Temel boyutsallık türleri şunlardır:

- Bir boyutlu problemler: Örneğin, çubukların kesilmesi.
- İki boyutlu problemler: Örneğin, kumaş veya metal levhaların kesilmesi.
- Üç boyutlu problemler: Örneğin, kutuların yerleştirilmesi.
- Çok boyutlu problemler: Örneğin, zamanın dördüncü boyut olarak ele alındığı sermaye bütçelemesi veya vektör yerleştirme problemleri.

1.3.2. Miktar Ölçümü

Miktar ölçümü, nesnelerin sayısına veya toplam uzunluğuna göre yapılır. İki temel ölçüm türü vardır:

- Ayrık (tam sayı) ölçüm: Belirli bir şekle sahip nesnelerin sayısının dikkate alındığı durum.
- Sürekli (kesirli) ölçüm: Aynı şekle sahip birden fazla nesnenin toplam uzunluğunun dikkate alındığı durum.

1.3.3. Kesilen Cismin Şekli

Şekil, nesnelerin geometrik temsili olarak tanımlanır. İlgili boyutların bir uzayında form, boyut ve yönelimle açıkça tanımlanabilir. Tek boyutlu problemler açısından, boyut en önemli özelliktir ve aşağıdaki gibi sınıflandırılabilir:

- Düzenli (standart) şekiller: Örneğin, dikdörtgen veya kare.
- Düzensiz (standart olmayan) şekiller: Örneğin, karmaşık geometrik şekiller.

1.3.4. Çeşitlilik

Çeşitlilik, problemin çözümünde kullanılan nesnelerin şekillerinin ve sayılarının tanımlanmasıyla ilgilidir. İki temel durum vardır:

- Farklı şekillerin bulunduğu durumlar: Örneğin, farklı boyutlarda kutuların yerleştirilmesi.
- Tüm öğelerin aynı şekle sahip olduğu durumlar: Örneğin, aynı boyuttaki çubukların kesilmesi.

1.3.5. Kullanılabilirlik

Kullanılabilirlik, nesnelerin miktarlarının alt ve üst sınırları, sıraları, düzenleri ve işlemin yapılacağı tarih gibi unsurları kapsar. Bu özellik, problemin çözümünde dikkate alınması gereken kısıtlamaları belirler.

1.3.6. Desen Kısıtlamaları

Desen kısıtlamaları, geometrik kombinasyonlar oluşturulurken dikkate alınması gereken kısıtlamaları ifade eder. Dört ana grup altında toplanır:

- Mesafe kısıtlamaları: Küçük şekiller arasındaki minimum veya maksimum mesafeler.
- Yönelim kısıtlamaları: Küçük şekillerin birbirlerine veya büyük bir şekle göre yönelimi.
- Sıklık kısıtlamaları: Bir desendeki küçük öğelerin sıklığı.
- Kesim türü ve sayısı: İzin verilen kesim türleri ve sayısı.

1.3.7. Atama Kısıtlamaları

Atama kısıtlamaları, küçük öğelerin uygun desenlere yerleştirilmesi veya desenlerin büyük nesnelere atanması sırasında ortaya çıkar. Bu kısıtlamalar, atama türü, aşamaların sayısı, desenlerin sıklığı veya sırası gibi unsurları içerir.

1.3.8. Hedefler

Hedefler, problemin çözümünde maksimize edilmesi veya minimize edilmesi gereken kriterleri ifade eder. Hedefler şunlara odaklanabilir:

- Büyük nesnelerin veya desenlere atanan küçük parçaların miktarı.
- Desenlerin geometrisi veya sırası.
- Desenlerin sayısı.

1.3.9. Bilgi Durumu

Bilgi durumu, problem verilerinin deterministik, stokastik veya belirsiz olmasıyla ilgilidir. Bu durum, problemin çözümünde kullanılan yöntemleri etkiler.

Problemlerin temel özellikleri, boyutsallık, miktar ölçümü, şekil, çeşitlilik, kullanılabilirlik, desen kısıtlamaları, atama kısıtlamaları, hedefler ve bilgi durumu gibi kriterlere göre sınıflandırılmıştır. Tablo 1.2‘de özetlenen bu yaklaşım, kesme ve

yerleştirme problemlerinin çözümünde kullanılan yöntemlerin daha iyi anlaşılmasını ve uygulanmasını sağlar.

Tablo 1.2. Dyckoff sınıflandırma sistemi (Dyckhoff, 1990).

Kriter	**Seçenekler**
Boyutsallık	Bir boyutlu (1)
	İki boyutlu (2)
	Üç boyutlu (3)
	N > 3 olan n boyutlu (N)
Atama türü	Tüm nesneler ve öğelerin bir seçimi
	Nesnelerin bir seçimi ve tüm öğeler
Stok malzemelerinin çeşitliliği	Tek örnek stok malzemesi (O) (*one figüre*)
	Aynı şekil (I) (*Identical figüre*)
	Farklı şekiller (D) (*different figures*)
Küçük nesnelerin çeşitliliği	Az sayıda farklı şekil (F) (*few items*)
	Çok sayıda farklı şekil (M) (*many items*)
	Göreceli olarak az sayıda farklı (uyumsuz) şekil (R) (*Relatively few*)
	Benzer Şekillerde parçalar (C) (*congruent figures*)

Dyckhoff'un sınıflandırması, bu alandaki problemlerin sistemfire bir şekilde ele alınmasına olanak sağlamış; ancak, zamanla ortaya çıkan eksiklikleri ve sınırlamaları nedeniyle çeşitli genişletme ve iyileştirme önerilerinin konusu olmuştur. Bu genişletmeler, sınıflandırmanın daha esnek hale gelmesini, farklı problem türlerini daha iyi kapsamasını ve zaman boyutunu dikkate almasını amaçlamıştır. Aşağıda, bu genişletmeler ve eleştiriler detaylı bir şekilde özetlenmiştir.

Gradišar ve ark. (2002), Dyckhoff'un sınıflandırmasının üçüncü grubuna (bir desendeki küçük öğelerin sıklığıyla ilgili kısıtlamalar) ek bir olasılık sunarak sınıflandırmayı genişletmiştir. Bu öneri, sınıflandırmanın daha esnek hale gelmesini ve farklı problem türlerini daha iyi kapsamasını sağlamayı hedeflemiştir. Önerilen yeni boyut aşağıdadır:

- (G) Aynı büyük nesnelerden oluşan birkaç grup: Bu genişletme, bir desendeki küçük öğelerin sıklığıyla ilgili kısıtlamalara yeni bir boyut ekler ve sınıflandırmanın daha kapsamlı olmasını sağlar.

Dyckhoff (1990) sınıflandırmasının, stok kesme problemlerinin ardışık ve anlık çözümleri arasında ayrım yapamaması nedeniyle Trkman ve Gradišar (2007), zaman boyutunu dikkate alan yeni bir tipoloji önermiştir. Bu tipoloji, optimizasyonun zaman aralığını ifade eder ve üç seçenekten oluşur:

1. IN (anlık): Belirli bir dönemdeki veriler için kesme/yerleştirme probleminin optimizasyonu. Bu, tek bir zaman diliminde çözüm arayan problemleri kapsar.
2. CD (ardışık deterministik): İki veya daha fazla ardışık zaman dilimi için optimizasyon yapılır. Tüm zaman dilimlerine ait talep ve arz önceden bilinir. Bir zaman diliminin sonunda kullanılmayan malzeme, daha sonraki bir zaman diliminde kullanılabilir.
3. CS (ardışık stokastik): İki veya daha fazla zaman dilimi için optimizasyon yapılır, ancak yalnızca ilk döneme ait veriler kesin olarak bilinir. Gelecekteki verilere ilişkin olasılık dağılımları bilinebilir ve bir zaman diliminin sonunda kullanılmayan malzeme, daha sonraki bir zaman diliminde kullanılabilir.

Bu genişletme, kesme ve yerleştirme problemlerinde zaman boyutunu dikkate alarak, artık malzemelerin yeniden kullanılmasını ve gelecekteki belirsizliklerin yönetilmesini sağlar. Özellikle, gelecekteki siparişlere yönelik kesin talebin önceden bilinmediği durumları ele alır.

Wäscher ve ark. (2007), Dyckhoff sınıflandırmasının eksikliklerini aşağıdaki gibi özetlemiştir:

- Bazı problem türleri mevcut sınıflandırma sistemine uymamaktadır.
- Aynı problem birden fazla alt sınıfta sınıflandırılabilir, bu da belirsizlik yaratır.
- Alt sınıflar, birbirinden farklı özelliklere sahip problemleri bir araya getirebilir.

Bu eleştiriler, sınıflandırmanın kapsamını genişletmek ve daha tutarlı bir sistem oluşturmak için yeni bir tipolojinin geliştirilmesi gerektiğini ortaya koymuştur.

Wäscher ve ark. (2007), Dyckhoff (1990) sınıflandırmasının eksikliklerini gidermek ve kesme ve yerleştirme problemlerini daha iyi kapsayacak bir sistem oluşturmak amacıyla yeni bir tipoloji önermiştir. Tablo 1.3'te gösterilen yeni tipolojinin detayları, Dyckhoff'un sınıflandırmasındaki eksiklikleri gidermeye yönelik bir çerçeve sunar.

Tablo 1.3. Wäscher sınıflandırma sistemi.

Kriter	**Seçenekler**
Boyutsallık	Bir boyutlu
	İki boyutlu
	N > 3 olan n boyutlu
Problem türü	Çıktı maksimizasyonu (tüm küçük nesneler için talebi karşılamak için yeterli büyük nesne yok)
	Girdi minimizasyonu (tüm küçük nesneler için talebi karşılamak için yeterli büyük nesne var)
Sipariş uzunlukları dağılımı	Aynı sipariş uzunlukları
	Sipariş uzunluklarının dağılımında zayıf
	Sipariş uzunluklarının dağılımında güçlü
Stok uzunlukları dağılımı	Tek bir büyük nesne
	Çok sayıda büyük nesne
Sipariş nesnelerin şekli	Olağan şekil
	Olağandışı şekil

2. LİTERATÜR ARAŞTIRMASI

Tek boyutlu stok kesme problemi, endüstriyel üretimde malzeme kaybını azaltma ihtiyacından ortaya çıkmış ve matematiksel optimizasyonun önemli bir dalı haline gelmiştir. Problemin tarihsel gelişimi aşağıda özetlenmiştir:

- İlk formülasyon (1930'lar): Leonid V. Kantorovich (1939), üretim süreçlerinde kaynak verimliliğini artırmak için matematiksel modeller oluşturmuştur. Bu çalışmalar, stok kesme probleminin ilk sistemfire şekilde ele alınışı olarak değerlendirilir. Ancak, dönemin kısıtlı bilgisayar teknolojisi nedeniyle çözümler genellikle teorik düzeyde kalmıştır.
- Doğrusal programlama ve erken çözüm teknikleri (1950'ler): Kantorovich ve Zalgaller (1951), lineer programlama çerçevesinde stok kesme problemini ele almış, George Dantzig (1947)'in simplex yöntemi bu alanda teorik altyapı sağlamıştır. Ancak, büyük ölçekli problemler hala çözülememiştir.
- Sütun oluşturma yöntemi (1960'lar): Gilmore ve Gomory (1961), sütun oluşturma yöntemini geliştirerek stok kesme probleminin çözümünde çığır açmıştır. Bu yöntem, problemi ana problem ve alt problem olarak ikiye ayırarak endüstriyel uygulanabilirliği artırmıştır.
- NP-Tam karmaşıklık ve teorik İlerlemeler (1970'ler): Problemin NP-Tam olduğu kanıtlanmış, Branch-and-Bound gibi tamsayı programlama teknikleri uygulanmıştır. H. Dyckhoff (1971), kesme ve yerleştirme problemlerini sınıflandırarak stok kesme problemini sistemfire bir çerçeveye oturtmuştur.
- Bilgisayar teknolojisi ve sezgisel yöntemler (1980'ler): Bilgisayar teknolojisinin gelişmesiyle dinamik programlama, sezgisel ve meta-sezgisel yöntemler (simulated annealing, tabu arama) geliştirilmiştir. Bu yöntemler, hızlı ve optimuma yakın çözümler sunarak endüstriyel uygulamalarda yaygınlaşmıştır.
- Modern gelişmeler (2000'ler-günümüz): Yapay zekâ ve makine öğrenimi teknikleri (genetik algoritmalar, karınca koloni optimizasyonu) stok kesme

probleminin çözümünde kullanılmaya başlanmıştır. Hibrit yaklaşımlar ve modern optimizasyon çözücüleri (Gurobi, CPLEX) problemin çözümünü daha etkili hale getirmiştir. Sürdürülebilirlik ve kaynak verimliliği, problemin önemini artırmıştır.

Sonuç olarak, tek boyutlu stok kesme problemi, teorik bir problemden endüstriyel üretimde vazgeçilmez bir optimizasyon aracına dönüşmüştür. Bu gelişim, matematiksel teorinin ilerlemesi ve teknolojik yeniliklerin birleşimiyle mümkün olmuştur.

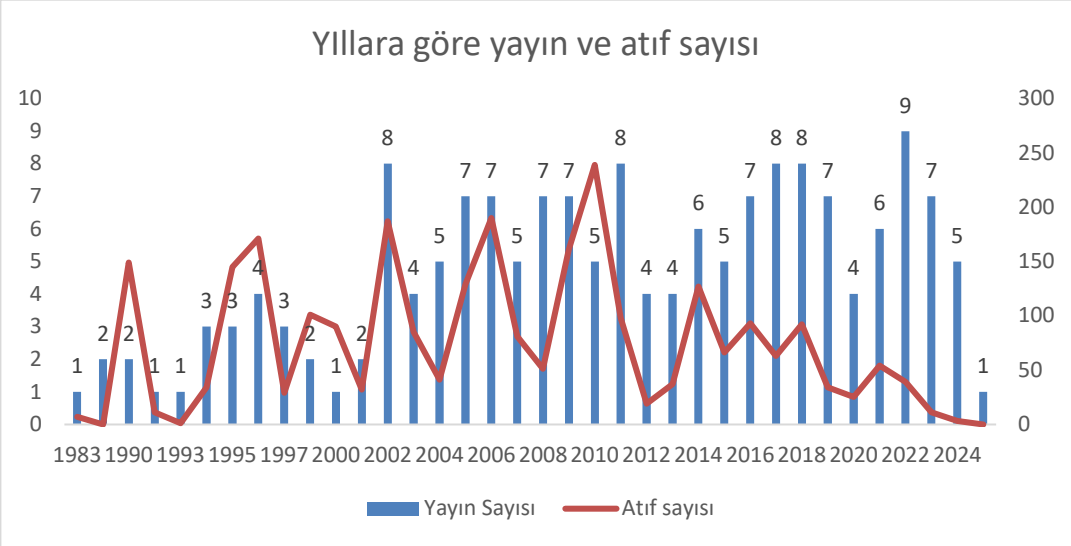
2.1. Web of Science Analizi

Web of Science (WoS), Clarivate tarafından sağlanan geniş kapsamlı bir akademik veri tabanı platformudur. İlk olarak 1997 yılında Institute for Scientific Information tarafından geliştirilmiştir. Bilim, sosyal bilimler, sanat ve beşerî bilimler gibi 256 disiplini kapsayan içerikler sunar. 1900'den günümüze uzanan yayınları içerir ve 79 milyon temel koleksiyon kaydı ile toplamda 171 milyon kayda sahiptir. Araştırmacılar, WoS ile atıf analizleri gerçekleştirebilir, literatürdeki trendleri ve yeni araştırma alanlarını belirleyebilir. İçerik seçimi; etki, zamanlama, hakem değerlendirmesi ve coğrafi çeşitlilik kriterlerine dayanır. Platform, Science Citation Index Expanded, Social Sciences Citation Index ve Arts & Humanities Citation Index gibi alt veri tabanlarının yanı sıra bölgesel indeksleri de bünyesinde barındırır. WoS, akademik çalışmaların değerlendirilmesi ve yaygınlaştırılmasında kritik bir rol oynar.

Analiz için WoS'da "one dimensional cutting stock problem" anahtar kelimesi ile arama yapılmıştır. Aramada tüm veritabanları açık tutulmuştur. Bu bölümde arama sonuçları detaylı olarak analiz edilmiştir.

Şekil 2.1 yıllara göre WoS'ta yapılan yayın ve atıf sayısını göstermektedir. Araştırma kapsamında incelenen 1983-2025 yılları arasındaki 169 yayın, toplamda 2695 atıf almış olup yayın başına ortalama 15,95 atıf oranı göstermiştir. Yayın sayısı 1983'te 1 iken, 2000'li yıllardan itibaren belirgin bir artış eğilimi sergilemiş ve 2022'de 9 yayınla zirveye ulaşmıştır. Atıf sayıları yıllar içinde dalgalanmalar göstermiş, özellikle 2010 yılı 239 atıfla en yüksek seviyeye ulaşmıştır. Son yıllardaki (2020 sonrası) yayın ve atıf sayılarındaki göreceli düşüş, bu yayınların henüz yeterli atıf alma süresini tamamlamamış olmasıyla ilişkilendirilebilir; bununla birlikte, genel yayın sayısındaki

artış trendi, araştırma alanına olan ilginin ve bilimsel faaliyetlerin zaman içinde yoğunlaştığını göstermektedir.



Şekil 2.1. WOS yıllara göre yayın ve atıf sayısı.

Tablo 2.1'de gösterilen 169 yayının belge türlerine göre dağılımında, en yaygın tür 108 adet ile makale olup, bunu 27 adet ile bildiri ve 14 adet ile tez takip etmektedir. Desteklenen proje 8 kez görülürken, daha az sıklıkta rastlanan türler arasında derleme ve editoryal materyal ise birer kez yer almıştır. Bu dağılım, incelenen alandaki bilimsel çalışmaların ağırlıklı olarak akademik dergilerde makale formatında yayımlandığını, bununla birlikte konferans bildirileri, tezler ve proje çıktılarının da literatüre önemli katkılar sağladığını göstermektedir.

Tablo 2.1. Türüne göre yayın sayıları.

Yayın Türü	**Toplam**
Makale	108
Bildiri	27
Tez	14
Desteklenen proje	8
Derleme	1
Editoryal içerik	1
Toplam	**169**

Tablo 2.2 yayın yerine göre makale sayılarını göstermektedir. European Journal of Operational Research dergisi %12,43'lük oranla en yüksek paya sahip olup, bunu

%8,28 ile tezler ve %4,73 ile desteklenen projeler izlemektedir. Computers & Operations Research, Journal of the Operational Research Society ve Pesquisa Operacional dergileri eşit %2,96'lık paylara sahipken, diğer dergilerdeki yayın oranları %2,37 ve altında seyretmektedir. Dikkat çekici bir bulgu olarak, yayınların %57,39'u "Diğer" kategorisinde toplanmıştır; bu durum, araştırmaların geniş bir yayın yelpazesine dağıldığını ve belirli bir dergide yoğunlaşmadığını göstermektedir. Yayınların çeşitli platformlarda yer alması, incelenen konunun disiplinler arası niteliğini ve farklı araştırma alanlarında ilgi gördüğünü ortaya koymaktadır.

Tablo 2.2. Yayın yerine göre makale sayıları.

Yayın yeri	**Toplam**	**%**
European Journal Of Operational Research	21	12,43
Tezler	14	8,28
Desteklenen Proje	8	4,73
Computers & Operations Research	5	2,96
Journal of the Operational Research Society	5	2,96
Pesquisa Operacional	5	2,96
International Transactions in Operational Research	4	2,37
International Journal of Production Research	4	2,37
Discrete Applied Mathematics	3	1,78
Annals of Operations Research	3	1,78
Diğer	97	57,39
Toplam	**169**	**100**

Amaç fonksiyonuna göre makale sayıları Tablo 2.3'te sunulmuştur. Malzeme firesi minimizasyonu %42,28 ile en sık karşılaşılan amaç olarak öne çıkarken, bunu %12,08 ile maliyet optimizasyonu ve %8,05 ile malzeme firesi minimizasyonu ve artık yönetimi kombinasyonu takip etmektedir. Diğer önemli hedefler arasında hesaplama verimliliği (%6,04) ve desen optimizasyonu (%4,03) yer almaktadır. Malzeme firesi minimizasyonu'nun diğer hedeflerle (desen sayısı optimizasyonu, üretim planlaması, kurulum maliyeti optimizasyonu gibi) birleştirildiği çalışmalar toplamda önemli bir orana sahiptir. Bu dağılım, araştırmaların ağırlıklı olarak operasyonel verimliliği artırmaya yönelik olduğunu, ancak çok amaçlı yaklaşımlar ve kaynak yönetimi gibi

spesifik hedeflerin de dikkate alındığını göstermektedir, böylece hem pratik endüstriyel ihtiyaçlara hem de teorik problemlere odaklanıldığı anlaşılmaktadır.

Tablo 2.3. Amaç fonksiyonuna göre makale sayıları.

Goal	Amaç	Toplam	%
Material Waste Minimization	Malzeme Firesi Minimizasyonu	63	42,28
Cost Optimization	Maliyet Optimizasyonu	18	12,08
Material Waste Minimization, Leftover Management	Malzeme Firesi Minimizasyonu, Artık Yönetimi	12	8,05
Computational Efficiency	Hesaplama Verimliliği	9	6,04
Material Waste Minimization, Pattern Optimization	Malzeme Firesi Minimizasyonu, Desen Optimizasyonu	7	4,70
Pattern Optimization	Desen Optimizasyonu	6	4,03
Material Waste Minimization, Production Planning	Malzeme Firesi Minimizasyonu, Üretim Planlaması	5	3,36
Resource Management	Kaynak Yönetimi	3	2,01
Material Waste Minimization, Setup Optimization	Malzeme Firesi Minimizasyonu, Kurulum Optimizasyonu	3	2,01
Material Waste Minimization, Multi-objective Approaches	Malzeme Firesi Minimizasyonu, Çok Amaçlı Yaklaşımlar	3	2,01
Cost Optimization, Pattern Optimization	Maliyet Optimizasyonu, Desen Optimizasyonu	2	1,34
Material Waste Minimization, Computational Efficiency	Malzeme Firesi Minimizasyonu, Hesaplama Verimliliği	2	1,34
Production Planning	Üretim Planlaması	2	1,34
Cost Optimization, Production Planning	Maliyet Optimizasyonu, Üretim Planlaması	2	1,34
Leftover Management	Artık Yönetimi	2	1,34
Material Waste Minimization, Resource Management	Malzeme Firesi Minimizasyonu, Kaynak Yönetimi	2	1,34
Setup Optimization, Production Planning	Kurulum Optimizasyonu, Üretim Planlaması	1	0,67
Resource Management, Setup Optimization	Kaynak Yönetimi, Kurulum Optimizasyonu	1	0,67
Leftover Management, Pattern Optimization	Artık Yönetimi, Desen Optimizasyonu	1	0,67
Multi-objective Approaches	Çok Amaçlı Yaklaşımlar	1	0,67
Setup Optimization	Kurulum Optimizasyonu	1	0,67
Cost Optimization, Setup Optimization	Maliyet Optimizasyonu, Kurulum Optimizasyonu	1	0,67
Cost Optimization, Leftover Management	Maliyet Optimizasyonu, Artık Yönetimi	1	0,67
Computational Efficiency, Pattern Optimization	Hesaplama Verimliliği, Desen Optimizasyonu	1	0,67
Toplam		**149**	**100**

Tablo 2.4 çözümde kullanılan yönteme göre makale sayılarını göstermektedir. Çalışmada kullanılan algoritma kategorilerinin dağılımında, 24 kez kullanılan hibrit yöntemler (hybrid methods) en yaygın yaklaşım olarak öne çıkmakta, bu da farklı algoritmaların birleştirilerek daha etkili çözümler üretme eğilimini göstermektedir.

Sezgisel yaklaşımlar (heuristic approaches) ve matematiksel programlama (mathematical programming) 19'ar kez kullanılarak ikinci sırada yer alırken, doğa esinli metaheuristikler (nature-inspired metaheuristics) 15 kez ve kesin yöntemler (exact methods) 13 kez ile bunları takip etmektedir. Daha spesifik kategorilerden hibrit metaheuristikler (hybrid metaheuristics) 9 kez, teorik analiz (theoretical analysis) ve yerel arama yöntemleri (local search methods) 6'şar kez kullanılmıştır. Makine öğrenimi yaklaşımları (machine learning approaches), stokastik yöntemler (stochastic methods) ve dinamik programlama (dynamic programming) daha az sıklıkta görülmüştür. Bu veriler, araştırmalarda hibrit ve sezgisel yöntemlerin ön planda tercih edildiğini, ayrıca matematiksel programlama ile doğa esinli yaklaşımların da kayda değer bir yer edindiğini göstermektedir.

Tablo 2.4. Kullanılan yönteme göre makale sayıları.

Gruplanmış Kategori	**Toplam**	**%**
Hibrit Yöntemler (Hybrid Methods)	24	16,11
Sezgisel Yaklaşımlar (Heuristic Approaches)	19	12,75
Matematiksel Programlama (Mathematical Programming)	19	12,75
Doğa İlhamlı Metasezgiseller (Nature-Inspired Metaheuristics)	15	10,07
Kesin Yöntemler (Exact Methods)	13	8,72
Hibrit Metasezgiseller (Hybrid Metaheuristics)	9	6,04
Teorik Analiz (Theoretical Analysis)	6	4,03
Yerel Arama Yöntemleri (Local Search Methods)	6	4,03
Ağ Akış Modelleri (Network Flow Models)	4	2,68
Sıralama Yöntemleri (Sequencing Methods)	4	2,68
Desen Tabanlı Yöntemler (Pattern-Based Methods)	4	2,68
Çok Amaçlı Optimizasyon (Multi-Objective Optimization)	4	2,68
Makine Öğrenmesi Yaklaşımları (Machine Learning Approaches)	3	2,01
Değerlendirme Teknikleri (Evaluation Techniques)	3	2,01
Stokastik Yöntemler (Stochastic Methods)	2	1,34
Özelleştirilmiş Modeller (Specialized Models)	2	1,34

Tablo 2.4. (Devamı) Kullanılan yönteme göre makale sayıları.

Gruplanmış Kategori	Toplam	%
İstatistiksel Yöntemler (Statistical Methods)	2	1,34
Dinamik Programlama (Dynamic Programming)	2	1,34
Problem Dönüşümü (Problem Transformation)	2	1,34
Metasezgiseller (Metaheuristics)	2	1,34
Ticari Çözücüler (Commercial Solvers)	1	0,67
Paralel Hesaplama Yöntemleri (Parallel Computing Methods)	1	0,67
Etkileşimli Yöntemler (Interactive Methods)	1	0,67
Ayrıştırma Yöntemleri (Decomposition Methods)	1	0,67
Toplam	**149**	**100**

Tablo 2.5, yazarların akademik üretkenliklerini farklı perspektiflerden değerlendirmektedir. İlk kısım, yazarları aldıkları toplam atıf sayısına göre sıralarken, ikinci kısım yayın sayısına göre sıralamaktadır. Atıf sayısına göre sıralamada, Scheithauer, G. 431 atıf ile en üst sırada yer alırken, yayın sayısına göre sıralamada da 12 yayın ile liderliğini korumaktadır. Ancak, diğer yazarlar arasında sıralama değişiklik göstermektedir; örneğin, Arenales, M. N. atıf sıralamasında üst sıralarda yer alırken, yayın sayısına göre daha alt sıralarda bulunmaktadır. Benzer şekilde, Poldi, K. C. ve Cherri, A. C. gibi yazarlar yayın sayısında öne çıkarken, atıf sayısında daha geride kalabilmektedir. Bu durum, yayınların akademik etkisinin (atıf sayısı) ve üretkenliğin (yayın sayısı) her zaman paralel olmadığını göstermektedir. Ayrıca, "Diğer" kategorisi, toplam yayın ve atıf sayısında önemli bir paya sahiptir ve bu, bireysel yazarların dışında kalan geniş bir katkı grubunu işaret etmektedir.

Tablo 2.5. Atıf sayısı ve makale sayısına göre sıralı yazar isimleri.

Atıf sayısına göre sıralama			Yayın sayısına göre sıralama		
Yazar	**Yayın sayısı**	**Atıf Sayısı**	**Yazar**	**Yayın sayısı**	**Atıf Sayısı**
Scheithauer, G.	12	431	Scheithauer, G.	12	431
Arenales, M. N.	6	257	Poldi, K. C.	8	205
Belov, G	4	251	Cherri, A. C.	7	175
Poldi, K. C.	8	205	Arenales, M. N.	6	257
Wascher, G	2	201	Cui, Y.	6	158

Tablo 2.5. (Devamı) Atıf sayısı ve makale sayısına göre sıralı yazar isimleri.

Atıf sayısına göre sıralama			Yayın sayısına göre sıralama		
Yazar	**Yayın sayısı**	**Atıf Sayısı**	**Yazar**	**Yayın sayısı**	**Atıf Sayısı**
Gau, T	2	201	Terno, J	6	135
Cherri, A. C.	7	175	De Araujo, S. A.	6	67
Yanasse, H. H.	5	161	Yanasse, H. H.	5	161
Cui, Y.	6	158	Adriana C. C.	5	0
Terno, J	6	135	Belov, G	4	251
Diğer	360	4920	Diğer	372	4504
Toplam	**437**	**6344**		**437**	**6344**

2.2. Literatür Araştırması

Tek boyutlu kesim stok probleminin amaçları oldukça çeşitlidir. Bu amaçlar arasında fire minimizasyonu (Gilmore ve Gomory, 1963), birleşik fire ve desen minimizasyonu (Cui ve ark., 2015), fire minimizasyonu ile birlikte yeniden kullanılabilir artıkların maksimizasyonu (Cui ve ark., 2017), fire minimizasyonu ve çizelgeleme optimizasyonunun bir arada ele alınması (Wang ve ark., 2019), parti büyüklüğünün maksimizasyonu ile fireların minimize edilmesi (Andrade ve ark., 2021) ve hem fireların minimize edilmesi hem de açık yığın sayısının azaltılması (Arbib ve ark., 2016) yer almaktadır.

Desen minimizasyonlu tek boyutlu stok kesme problemi genellikle Haessler (1975) tarafından tanıtılan desen oluşturma yöntemlerine dayanan sezgisel yaklaşımlar veya yerel arama teknikleriyle birleştirilen meta-sezgisel yöntemler kullanılarak ele alınmaktadır. Ayrıca, küçük boyutlu problemlerin çözümü için sınırlı sayıda kesin yöntem çalışması da bulunmaktadır (Henn & Wascher, 2013).

Bu kısımda yer alan inceleme, yalnızca aynı anda fire ve desen minimizasyonu ile ilgili çalışmalara odaklanmaktadır.

2.2.1. Sezgisel yöntemler

Haessler (1975), belirli fire ve desen kullanım hedeflerini (istek seviyeleri) karşılayan kesim desenleri oluşturmak için sıralı sezgisel prosedür (SHP) yöntemini tanıtmıştır. Eğer uygun bir desen oluşturulamazsa, izin verilen fire miktarı artırılır ve istek seviyeleri gevşetilir. Bu yinelemeli süreç, tüm taleplerin kontrollü bir şekilde

karşılanana kadar devam eder. SHP, çeşitli kesim planlarını araştırır, fire ve istek seviyelerini ayarlar ve en uygun olanı seçer. Haessler'in çalışmasını takiben, birçok araştırmacı, uygun desenler oluşturma, desen sayısını minimize etme ve stok kullanımını optimize etme yöntemlerinde farklılık gösteren çeşitli desen oluşturma yaklaşımları geliştirmiştir.

Goulimis (1990), tüm potansiyel kesim desenlerini dikkate alan bir yöntem önermiş, ilişkili tamsayılı programı kesim düzlemleri ve dallanma-sınır teknikleriyle çözmüş ve bıçak kurulumlarını azaltmak için çözüm sonrası optimizasyonu çalışmıştır. Ancak, bu yaklaşım, yüksek boyutlu problemlerdeki yüksek hesaplama maliyetleri nedeniyle 'yumuşak' problemler (minimum kesim kaybı olan problemler) için daha az etkili olmuştur.

Vahrenkamp (1996), 200 rastgele öğe kombinasyonu kullanan bir sıralı sezgisel yöntem geliştirmiş ve genetik algoritmalardan çaprazlama tekniklerini kullanarak çeşitli çözümleri keşfetmiştir.

Foerster ve Wascher (2000), Diegel ve ark.'nın (1993) desen minimizasyonunu geliştirerek, desenleri birleştirip sonuç frekanslarını başlangıçta birleştirilen desenlerle hizalayan KOMBI yöntemini tanıtmıştır.

Yanasse ve Limeira (2006), üç aşamalı bir hibrit yöntem önermiştir. Bu yaklaşımın ilk aşaması, talebi karşılarken fireları en aza indiren verimli kesim desenlerinin oluşturulması ve seçilmesidir. İkinci aşamada, karşılanamayan talepler veya oluşan fazla firelar ele alınır. Son aşamada ise, kullanılan desen sayısı azaltılarak sistemin optimizasyonu sağlanır.

Dikili ve ark. (2007), çok sayıda kesim deseni oluşturmuş ve talepleri fazla üretim olmadan karşılamayı garanti ederek, fire minimizasyonu ve desen çok yönlülüğü gibi kriterlere dayalı en iyi deseni seçmiştir. Ancak, bu yöntem, başlangıçta tüm olası desenlerin oluşturulması gerektiğinden çok büyük problemlerde çalışmakta zorlanmıştır.

Renildo ve ark. (2009), öğeleri talebe dayalı iki gruba ayıran bir sezgisel yöntem tanıtmış, karşılanmayan talepleri ayrı olarak ele almış ve optimizasyon için bir desen azaltma tekniği uygulamıştır.

Cui ve Liu (2011), kesim desenlerini adım adım oluşturan bir sıralı sezgisel algoritma önermiş, maliyet etkin desenleri seçmek ve kalan öğelerden yeni desenler oluşturmak için ileriye dönük bir strateji kullanmıştır.

Cui (2012), belirli desenlerin birden çok kez kullanıldığı kesim planları oluşturmak için bir CAM sistemi tabanlı çözüm önermiş, kesim maliyetlerini minimize etmeyi ve aynı zamanda desen kullanımını ve artık malzemeyi azaltmayı hedeflemiştir.

Mobasher ve Ekici (2011), bir karışık tamsayılı doğrusal program formüle etmiş ve yerel arama algoritmaları ile bir sütun oluşturma sezgisel yöntemi tanıtmıştır. Değerlendirmeleri, sütun oluşturma algoritmasının düşük kurulum maliyetlerinde iyi performans gösterdiğini, yerel arama yöntemlerinin ise yüksek kurulum maliyetlerinde daha başarılı olduğunu göstermiştir.

Cui ve ark. (2008), SHPC algoritmasını tanıtarak, kesim desenleri oluşturmak için sınırlı bir sırt çantası problemi kullanmış, desen sıklığını optimize etmiş ve kalan öğe türlerinin sınırlı bir setini dikkate alarak fire kaybını azaltmıştır.

Cui ve ark. (2015), SHP'yi bir tamsayılı doğrusal modelle birleştirerek toplam malzeme ve kurulum maliyetlerini en aza indirmeyi amaçlamıştır.

Martin ve ark. (2018), Haessler'in prosedürünü uyarlayarak, sıralı arama yöntemini tamsayılı sınırlı sırt çantası problemiyle değiştirmiş, yaklaşımlarına Değiştirilmiş Sıralı Sezgisel Prosedür (MSHP) adını vermiş ve çözüm kalitesini artırmak için kesim planlarını bir tamsayılı programlama modeliyle iyileştirmiştir.

Martin ve ark. (2022), yalın bir çerçeveye entegre edilmiş, pseudo-polynomial karmaşıklığa sahip desen tabanlı bir ILP formülasyonu sunmuş ve optimal çözümler bulmak için bu yöntemi kullanmıştır. Deneyleri, orta büyüklükteki nesneler ve sınırlı desenler içeren problemler için yöntemin etkinliğini göstermiş ve standart bir ILP çözücüsü kullanarak pareto optimal çözümleri verimli olarak belirlemiştir.

Guimarães ve ark. (2025), temel hedef olan malzeme kaybını en aza indirmenin yanı sıra, kullanılan farklı kesim desenlerinin sayısını ve aynı anda açık olan istiflerin maksimum sayısını azaltma gibi ek hedefler üzerinde durmuştur. Kesim desenlerinin değişimi üretimde zaman ve maliyet artışına yol açarken, açık istif sayısının sınırlandırılması, kesim makinesinin yakınındaki alan veya otomfire boşaltma istasyonlarının kısıtlı olduğu durumlar için kritiktir. Sorunu çözmek için iki Tamsayılı Doğrusal Programlama (ILP) formülasyonu ve bir Kısıt Programlama (CP) modeli

önerilmiştir. Önerilen yaklaşımlar, küçük ve orta ölçekli problem örneklerinde etkin çözümler sunarak, minimum malzeme kaybı, az sayıda farklı kesim deseni ve düşük açık istif sayısı ile Pareto cephesini belirlemede başarılı olmuştur.

2.2.2. Meta-sezgisel yöntemler

Umetani ve ark. (2003), yerel aramayı doğrusal programlama ile birleştiren bir hibrit yöntem önermiş, komşu çözüm sayısını azaltmak için duyarlılık analizi kullanmıştır. Bu yaklaşım, taleplere bağlı olarak ortaya çıkan fazlalıkları ve eksiklikleri ele almıştır.

Umetani ve ark. (2006), kesim desen çeşitliliğini sınırlarken stok hammadde kullanımını minimize etmeyi amaçlayan desen kısıtlı problemi (PRP) üzerine odaklanmış, bir öğe ekleme ve kaydırma temelli komşuluklar ile yerel arama algoritması kullanmış ve doğrusal programlamayı komşuluk çözümlerini kolaylaştırmak için entegre etmiştir.

Lee (2007), tamsayılı doğrusal programlamayı yerel arama sezgisel yöntemiyle birleştiren bir yöntem tanıtmış, model iyileştirmelerinin pratik çözümlere dönüşmesini sağlamıştır.

Golfeto ve ark. (2009) ile Araujo ve ark. (2014), yerel arama ile geliştirilen evrimsel algoritmalar kullanan meta-sezgisel yöntemler geliştirmiştir. Golfeto ve ark. (2009), çok amaçlı zorluklar için nüfus çeşitliliğini teşvik eden bir seçim stratejisi kullanarak çözümleri pareto cephesine yönlendirmiştir. Araujo ve ark. (2014), maliyet, fire minimizasyonu ve üretim hızı gibi kriterlere dayalı olarak karar vericilerin seçebileceği baskın olmayan çözümler üreten çok amaçlı bir genetik algoritma önermiştir.

Aliano Filho ve ark. (2018), desen kullanım sıklığını ve çeşitliliğini en aza indirmeyi amaçlayan iki amaçlı kesim stoğu problemini incelemiş, Pareto-optimal çözümleri belirlemek için üç yöntem (ağırlıklı toplam, Chebyshev metriği ve ε-kısıt) kullanmış ve değiştirilmiş bir Chebyshev metriği önermiştir.

2.2.3. Kesin yöntemler

Vanderbeck (2000), 5 ila 32 arasında değişen öğe türleriyle 16 gerçek dünya vakasında test edilen bir kuadrfire tamsayılı programlama yöntemi tanıtmış, her vaka için 2 saatlik bir zaman sınırı nedeniyle yalnızca 12 örneği tamamen çözebilmiştir. Bu, yöntemin daha küçük senaryolar için uygun olduğunu göstermektedir.

Belov (2003), fire minimizasyonu için Gilmore-Gomory modeline dayalı bir karışık tamsayılı programlama modeli sunmuş ve bir dallanma ve fiyatlandırma algoritması geliştirmiştir. Sonuçları, minimum malzeme girişinin yüzde 0,2 oranında aşılmasına izin verilmesinin çözüm kalitesini önemli ölçüde artırabileceğini göstermiştir.

Alves ve Valério de Carvalho (2008), model kısıtlamalarından eşitsizlikler türetmek için çift uygunluk fonksiyonları kullanmış, bir dallanma-kesme-fiyatlandırma algoritmasında bir yay akış formülasyonu uygulamış ve dallanma kısıtlamalarını diğer model kısıtlamalarıyla entegre ederek daha güçlü kesimler sağlamıştır.

Alves ve Valério de Carvalho (2009), sütun oluşturma yoluyla çözülebilen alternatif bir tamsayılı programlama modeli önermiş, kısıt programlama tekniklerini entegre etme ve yeni değerli kısıtlamalar ekleme gibi çeşitli iyileştirmeler sunmuştur.

Ghafari ve ark. (2024) kesim desenlerini optimize ederek üretkenliği artırmayı amaçlayan çalışmalarında makine bıçak değişimlerinden kaynaklanan kurulum sayısını ve üretim ile talep arasındaki farkı en aza indiren ve fireyi kısıt olarak koyan bir karma tamsayılı doğrusal olmayan model önermiştir. Model doğrusallaştırıldıktan sonra, mevcut şirket yöntemine kıyasla hem fireta hemde desen sayısında iyileştirmeler sağlamıştır.

2.3. Çözüm Yöntemlerinin İncelenmesi

Bu bölüm, 1D-CSP için geliştirilmiş çeşitli çözüm yaklaşımlarını sistemfire olarak incelemekte, her bir yöntemin matematiksel temellerini, algoritmik ayrıntılarını, avantajlarını, dezavantajlarını ve pratik uygulama alanlarını derinlemesine analiz etmektedir. Bu çalışmanın amacı, endüstriyel uygulamalardan üretim planlamasına uzanan geniş bir yelpazede kullanılan yöntemleri; tam çözümler, sezgisel yaklaşımlar, metasezgisel algoritmalar, hibrit yöntemler ve yeni gelişen teknikler ekseninde sınıflandırarak kapsamlı bir karşılaştırma ortaya koymaktır.

Bu problem için geliştirilen başlıca matematiksel modellerden biri Gilmore-Gomory Modeli'dir. Bu modelde, kesme desenlerinin kaç kez kullanılacağını belirleyen tamsayı değişkenler ve her desende kesilen parça miktarlarını ifade eden parametreler kullanılır. Model, sütun üretimi tekniğiyle çözülür; bu teknik, LP gevşetmesini iteratif olarak çözerek her adımda yeni kesme desenleri ekler. Yeni desenler, çift değişkenler

kullanılarak bir sırtçantası problemi çözülerek bulunur. Bu yöntem, büyük ölçekli problemlerde etkili bir çözüm sunar.

Bir diğer model olan Küme Kapsama Formülasyonu, kesme desenlerinin seçilip seçilmediğini belirleyen ikili değişkenler kullanır ve dal ve fiyat algoritmasıyla çözülür. Bu algoritma, sütun üretimi ve dal ve sınır yöntemlerini birleştirerek optimal çözüme ulaşır. Ayrıca, yay akış ve dinamik programlama yaklaşımları, problemi daha küçük alt problemlere ayırarak çözmeyi hedefler. Bu yöntemler, küçük ve orta ölçekli problemler için etkili olsa da büyük ölçekli problemlerde durum uzayının büyümesi nedeniyle hesaplama karmaşıklığı artar. Bu modeller, 1D-CSP'nin farklı ölçek ve gereksinimlerine göre uygulanabilir.

1D-CSP, NP-tam sınıfında yer alan bir problem olup, çözüm yaklaşımları genellikle tam çözüm yöntemleri, sezgisel algoritmalar, metasezgisel teknikler ve hibrit modeller şeklinde sınıflandırılmaktadır. Tam çözüm yöntemleri (örneğin, sütun üretimi, dal ve fiyat, kesme düzlemi yöntemleri), optimal çözümler sunar ancak büyük ölçekli problemlerde hesaplama maliyeti oldukça yüksektir. Özellikle sütun üretimi ve dal ve fiyat yöntemleri, büyük ölçekli problemlerde etkili sonuçlar verebilirken, dinamik programlama gibi yöntemler küçük ve orta ölçekli problemler için uygundur. Bu yöntemler, optimal çözüm garantisi sunmalarına rağmen, zaman ve bellek gereksinimleri nedeniyle sınırlı uygulanabilirliğe sahiptir.

Sezgisel yöntemler (örneğin, açgözlü algoritmalar, dinamik programlama tabanlı sezgiseller) ve metasezgisel yöntemler (örneğin, genetik algoritmalar, benzetimli tavlama, tabu arama, parçacık sürü optimizasyonu), büyük ölçekli problemlerde daha hızlı ve pratik çözümler sunar. Sezgisel yöntemler, hızlı olmalarına rağmen optimal çözüm garantisi vermezken, metasezgisel yöntemler geniş çözüm uzayını keşfetme yeteneğiyle yerel optimumlardan kaçınabilir. Bu yöntemler, özellikle gerçek zamanlı uygulamalarda veya büyük ölçekli problemlerde tercih edilir. Metasezgiseller doğadan esinlenen algoritmalarla çözüm kalitesini artırırken, parametre ayarları gibi faktörler performansı etkileyebilir.

Hibrit yaklaşımlar, farklı yöntemlerin güçlü yönlerini birleştirerek daha iyi performans elde etmeyi amaçlar. Örneğin, metasezgisel algoritmalarla üretilen çözümler, LP tabanlı yöntemlerle iyileştirilebilir. Ayrıca, makine öğrenmesi ve derin pekiştirmeli öğrenme gibi gelişmekte olan yaklaşımlar, geçmiş verilerden öğrenerek veya karar süreçlerini optimize ederek çözüm kalitesini artırabilir. Tablo 2.6'da özetlenen bu

yöntemler, problemin ölçeğine ve çözüm gereksinimlerine göre seçilerek hem hesaplama süresi hem de çözüm kalitesi açısından dengeli bir yaklaşım sunar.

Tablo 2.6. Karşılaştırmalı çözüm yöntemleri.

Yöntem Kategorisi	Çalışma Prensibi	Avantajlar	Dezavantajlar
Tam çözüm yöntemleri	Problemin optimal çözümünü garanti eder. Sütun üretimi, dal ve fiyat, kesme düzlemi ve dinamik programlama gibi yöntemler kullanılır.	Optimal çözüm garantisi.	Büyük ölçekli problemlerde hesaplama süresi ve bellek gereksinimi artar.
Sezgisel yöntemler	En iyi çözümü her zaman bulmasa da hızlı bir şekilde tatmin edici ve yeterli sonuçlar sunar. Açgözlü yaklaşımlar, dinamik programlama tabanlı sezgiseller, ağaç tabanlı sezgiseller ve kalıntı yönetimi gibi yöntemler içerir.	Basit, hızlı ve büyük ölçekli problemlerde uygulanabilir.	Optimal çözümden uzak sonuçlar verebilir.
Metasezgisel yöntemler	Doğadan esinlenen algoritmalarla geniş çözüm uzayını keşfeder. Genetik algoritmalar, benzetimli tavlama, tabu arama ve parçacık sürü optimizasyonu gibi yöntemler içerir.	Yerel optimumlardan kaçınabilir, geniş çözüm uzayını keşfeder.	Parametre ayarları performansı etkiler, hesaplama süresi artabilir.
Hibrit yaklaşımlar	Farklı yöntemlerin güçlü yönlerini birleştirerek daha iyi performans elde etmeyi amaçlar. Metasezgisel ve LP entegrasyonu, genetik algoritmalar ile ince ayar ve derin pekiştirmeli öğrenme gibi yöntemler içerir.	Çözüm kalitesini ve hesaplama süresini iyileştirebilir.	Uygulama karmaşıktır, veri gereksinimi yüksek olabilir.
Gelişmekte olan yaklaşımlar	Makine öğrenmesi tabanlı yöntemler ve diğer yenilikçi yaklaşımlar. Geçmiş verilerden öğrenerek problem özelliklerine göre çözüm stratejileri geliştirir.	Gelecekteki problemlerde daha iyi kararlar alınabilir.	Veri gereksinimi yüksektir, model eğitimi zaman alabilir.

3. PROBLEM ÇÖZÜMÜ İÇİN YENİ YÖNTEMİN GELİŞTİRİLMESİ

OPTICUT algoritması, desen minimizasyonunu esas alan tek boyutlu stok kesme problemini sistemfire bir şekilde çözmek için üç aşamadan oluşan bir yapı şeklinde geliştirilmiştir. Bu yapı, algoritmanın hem çözüm alanını etkin keşfetmesini hem de hesaplama verimliliğini korumasını sağlamaktadır. Süreç aşağıda özetlenmektedir:

Algoritma, başlangıç çözümünü oluşturmak için sütun oluşturma (*column generation*) tekniğinden yararlanır. Bu adım, başlangıç çözümüne hızlı ve verimli bir şekilde ulaşılmasını sağlar. Sütun oluşturma yöntemi, büyük boyutlu kesme problemlerinde çözüm uzayını daraltarak hesaplama yükünü azaltır ve algoritmanın ILP aşamasında kullanmak üzere desenler üretir.

İkinci aşamada, algoritma farklı desen oluşturma yaklaşımlarını benimseyerek çeşitli aday çözümler üretir. Bu yinelemeli süreç, aşağıdaki alt adımları içerir:

- Farklı kesim desenlerinin sistemfire oluşturulması,
- Gelecekte kullanılmak üzere oluşturulan desenlerin bir havuzda saklanması,
- Her bir aday çözümün toplam maliyetinin hesaplanarak kaydedilmesi.

Bu aşama, çözüm uzayının geniş bir bölümünü keşfetmeyi ve daha iyi çözümler elde etme olasılığını artırmayı hedefler.

Son aşamada, önceki adımda oluşturulan ve saklanan desenler havuzuna tamsayılı doğrusal programlama (*Integer Linear Programming-ILP*) tekniği uygulanır. Bu işlem, aday çözümler setine ek bir çözüm kazandırmayı ve mevcut çözümleri iyileştirmeyi amaçlar. ILP'nin uygulanması, belirlenen zaman limit içerisinde mevcut desenler içerisinden optimal çözümün belirlenmesinde kesin bir yöntem sunar.

Bu üç aşamanın tamamlanmasının ardından, algoritma tüm aday çözümler arasından en düşük toplam maliyete sahip olanı seçerek optimal çözümü belirler. OPTICUT'ın bu çok aşamalı yapısı, farklı optimizasyon tekniklerinin güçlü yönlerinden

yararlanmasını mümkün kılarak, algoritmanın çeşitli problem örneklerinde daha iyi performans göstermesini sağlar.

OPTICUT algoritmasının metodolojisi, kombinatoryal optimizasyon alanındaki güncel eğilimlerle uyumlu olarak hibrit bir yaklaşım benimsemektedir. Tek bir yönteme dayalı algoritmalara kıyasla, hibrit yaklaşımlar genellikle daha üstün sonuçlar üretmektedir. Bu bağlamda, OPTICUT'ın çok aşamalı yapısı hem çözüm kalitesini artırmakta hem de hesaplama verimliliğini korumaktadır.

Bu tezde kullanılan notasyonların çoğu, Cui ve ark. (2015) ile uyumlu olarak tanımlanmıştır. Kullanılan notasyonlar şunlardır:

m: Sipariş boyutu sayısı

Q: Desen seti

Z: Mevcut kesim planının maliyeti

$Z_{en\ iyi}$: En iyi kesim planının maliyeti

G: Mevcut neslin kimliği (deneme sayısı)

P: Mevcut desen

R_i: i tipi siparişler için kalan miktar talebi

D_i: i tipi siparişlerin başlangıç miktar talebi

V_i: i tipi siparişin değeri

Pr_i: i tipi siparişin önceliği

l_i: i tipi siparişin uzunluğu

pc: Öncelik katsayıları seti

SL: Stok uzunluğu

X_i: Mevcut desendeki i tipi siparişler için kullanım miktarı

OPTICUT algoritması çalışma sırasında beş ana fonksiyon çağırır: `Column_Generation`, `Assign_Priority`, `Algorithm_A`, `Algorithm_B` ve `ILP`. Bu işlevlerin her biri sonraki bölümlerde ayrıntılı olarak açıklanmaktadır. Algoritmanın ana yapısı Sözdekod 1 ve Şekil 3.1' de özetlenmiştir. Sözdekod 1'de bahsedilen G simgesi deneme sayısını belirtmektedir. Algoritma A için 20 deneme,

Algoritma B için ise 50 deneme yapılmaktadır. Deneme sayıları test problemlerine göre optimizasyon için harcanan zaman ve çözüm kalitesi kriterleri dikkate alınarak deneysel olarak belirlenmiştir. Deneme sayılarının artmasının etkileri duyarlılık analizi bölümünde gösterilmiştir.

Column generation, Algoritma A ve Algoritma B fonksiyonları birbirlerinden bağımsız çalışmaktadır. Yazılımın kodlanmasında paralel işleme kullanılmadığından bu işlevler sırası ile çağrılmaktadır. Paralel işleme özelliği iyileştirme önerisi olarak sunulmuştur. ILP aşamasında ise Column generation, Algoritma A ve Algoritma B'de üretilen desenler kullanılır. Assign_Priority fonksiyonunda hesaplanan değerler Algoritma A ve Algoritma B'de kullanılmaktadır.

Sözdekod 1. OPTICUT algoritmasının ana yapısı.

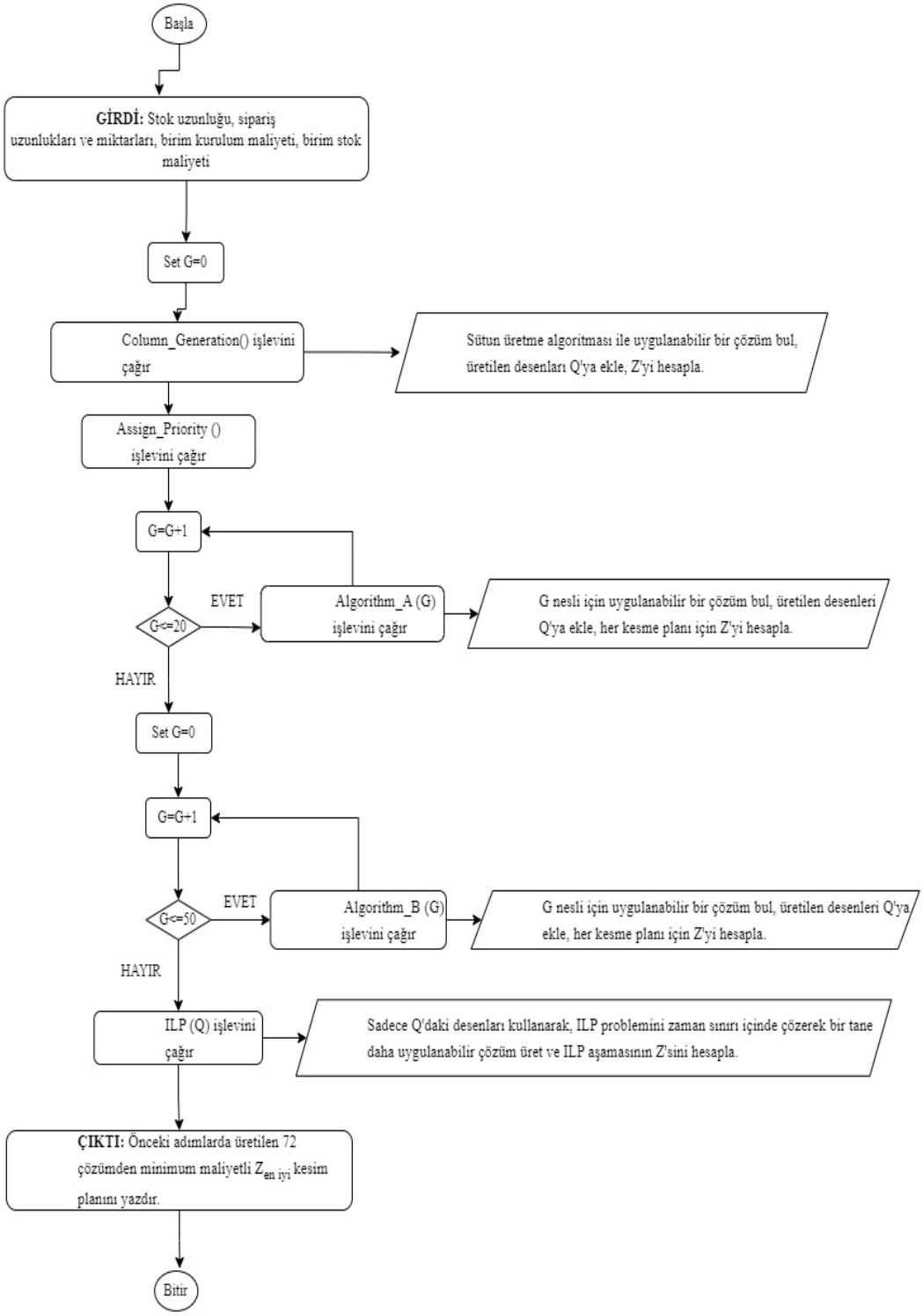
GİRDİ: Stok uzunluğu, sipariş uzunlukları ve miktarları, birim kurulum maliyeti, birim malzeme maliyeti

ÇIKTI: Bir dizi uygulanabilir çözüm ve en iyi çözüm

```
Siparişlerin uzunluğu ve miktarını içeren stok ve sipariş verilerini içe aktar,
    G=0 olarak ayarla.
Column_Generation() işlevini çağır.
    Sütun üretme algoritması ile bir çözüm bul, üretilen desenları Q'ya ekle,
    Z'yi hesapla.
Assign_Priority () işlevini çağır.
G <= 20 iken:
 Algorithm_A (G) işlevini çağır.
    G nesli için bir çözüm bul, üretilen desenleri Q'ya ekle, her kesme planı
    için Z'yi hesapla.
  G = G + 1 olarak ayarla.
G=1 olarak ayarla.
G <= 50 iken
 Algorithm_B (G) işlevini çağır.
    G nesli için bir çözüm bul, üretilen desenleri Q'ya ekle, her kesme planı
    için Z'yi hesapla.
  G = G + 1 olarak ayarla.
ILP (Q) işlevini çağır.
```

Sadece Q'daki desenleri kullanarak, ILP problemini zaman sınırı içinde çözerek bir tane daha uygulanabilir çözüm üret ve ILP aşamasının Z'sini hesapla.

Önceki adımlarda üretilen 72 adet çözüm arasından minimum maliyetli $Z_{en\ iyi}$ kesim planını yazdır.



Şekil 3.1. OPTICUT algoritmasının ana akış şeması.

3.1. Column_Generation Fonksiyonu

Algoritmada ilk kısımda sütün oluşturma algoritması kullanılmıştır. Gilmore ve Gomory (1963), tek boyutlu stok kesme problemlerinde kesim desenlerini sistematik olarak üretmek için doğrusal programlama gevşetmesi ile çözülen bir ön problem aracılığıyla çalışan sütun oluşturma (*column generation*) algoritmasını önermiştir. Bu yöntem, stok kesme problemlerinde fireyi en aza indirmeyi hedefleyen etkili bir yaklaşımdır. Algoritmanın temel mantığı, başlangıçta sınırlı sayıda kesim deseniyle başlamak ve daha sonra doğrusal programlama modeline yeni desenler ekleyerek çözümü kademeli olarak iyileştirmektir. Bu süreç, yalnızca çözümün iyileştirilmesine katkı sağlayacak desenlerin modele dahil edilmesini sağlar ve böylece gereksiz hesaplamaların önüne geçilir.

Sütun oluşturma algoritması, iki temel bileşenden oluşur: ana problem (*master problem*) ve alt problem (*subproblem*). Ana problem, mevcut kesim desenlerini kullanarak stok kesme problemini çözmeye çalışır. Alt problem ise, ana problemden elde edilen indirgenmiş maliyetler (*reduced costs*) yardımıyla yeni ve daha iyi kesim desenleri üretir. Bu iki bileşen, iteratif olarak birbirini besler ve algoritma, yeni bir kesim deseni bulunamadığında veya belirli bir durdurma kriteri karşılandığında sonlanır. Algoritmanın işleyişi aşağıda açıklanmaktadır:

Algoritma, başlangıçta genellikle basit ve sezgisel yöntemlerle oluşturulan bir dizi kesim deseniyle başlar. Bu desenler, stok kesme probleminin temel gereksinimlerini karşılayacak şekilde tasarlanır.

Mevcut kesim desenleri kullanılarak doğrusal programlama modeli çözülür. Bu model, her bir desenin ne kadar kullanılacağını belirler ve toplam malzeme kullanımını minimize etmeyi hedefler. Ancak, bu aşamada tamsayılı kısıtlar gevşetilir ve çözüm doğrusal programlama gevşetmesi olarak elde edilir.

Ana problemden elde edilen dual değerler kullanılarak alt problem formüle edilir. Alt problem, yeni bir kesim deseni oluşturmayı amaçlar ve genellikle bir sırt çantası problemi (*knapsack problem*) olarak modellenir. Bu problem, mevcut ikili *(dual)* değerler doğrultusunda negatif görece maliyete sahip bir desen bulmaya çalışır. Eğer böyle bir desen bulunursa, bu desen ana probleme eklenir.

Yeni bir desen bulunduğunda, ana problem tekrar çözülür ve süreç tekrarlanır. Bu yinelemeli süreç, yeni bir desen elde edilemediğinde ya da belirlenen bir durdurma koşulu sağlandığında sonlanır.

Algoritma, doğrusal programlama gevşetmesi ile çalıştığı için elde edilen çözüm genellikle kesirli değerler içerir. Bu nedenle, çözümün tamsayılı hale getirilmesi için bir yuvarlama işlemi uygulanır. Ancak, bu yuvarlama işlemi, genellikle aşırı üretime ve optimal olmayan çözümlere yol açabilir. Özellikle, sipariş boyutlarının çeşitliliği arttıkça, üretilen desenlerin sayısı da artar ve bu durum yuvarlama sürecini daha karmaşık hale getirir.

Sütun oluşturma algoritması, stok kesme problemlerinde fireyi minimize etme konusunda etkili bir yöntemdir. Fakat, sipariş çeşitliliğinin ve desen sayısının artması, algoritmanın performansında düşüşe yol açabilir. Bu nedenle, algoritmanın uygulanmasında, desen sayısını kontrol altında tutmak ve fireyi arttıran yuvarlama sürecini iyileştirmek için ek yöntemler veya kısıtlamalar kullanılmalıdır. Algoritmanın sözde kodu, Sözdekod 2'de sunulmaktadır.

Sözdekod 2. Sütun oluşturma algoritması.
GİRDİ: Stok uzunluğu, sipariş uzunlukları ve miktarları
ÇIKTI: Uygulanabilir bir çözüm

(1) Ana problemi bir başlangıç kesim modeliyle oluştur
(2) Tekrarla:
  (3) Optimal çözümü (LP) elde etmek için ana problemi çöz
  (4) Her bir kısıt için ikili fiyatları (gölge fiyatlar) çıkar
  (5) Alt problemde her bir sipariş türü için:
    (6) alt problemi çözerek yeni bir desen üretmenin indirgenmiş maliyetini hesapla
    (7) İndirgenmiş maliyet < 0 ise:
      (8) Yeni deseni ana probleme ekle
    (9) Negatif indirgenmiş maliyete sahip yeni bir desen bulunmazsa:
      (10) Dur (Optimal çözüm bulundu)
(11) Ana problemin çözümünden tam sayı kısıtıyla nihai kesim planını oluştur
(12) Optimum kesim modellerinin ve bunlara karşılık gelen miktarları göster

3.2. Assign_Priority İşlevi

Optimizasyon süreci, genellikle küçük boyutlu siparişlerin hızla işlenmesiyle başlar. Bu başlangıç aşamasında, açgözlü (*greedy*) bir yaklaşım benimsenir; bu da mevcut kaynakların hızlı tahsis edilerek kısa vadede çözüm elde edilmesini sağlar. Ancak, bu strateji, daha uzun siparişlerin işlenmesini geciktirebilir ve bu durum, optimizasyon sürecinin genel verimliliğini düşürerek fire miktarının artmasına neden olabilir. Bu yüzden, optimizasyon sürecinin başarısı açısından daha dengeli ve stratejik bir yaklaşım izlemek büyük önem taşımaktadır.

Bu süreçte, algoritmanın temel hedeflerinden biri, her bir sipariş türünün değerini (V_i) ve önceliğini (Pr_i) belirlemektir. Siparişlerin değerleri ve öncelikleri, optimizasyon sırasında hangi siparişlerin öncelikli olarak işleneceğini belirlemek için kullanılır. Özellikle, daha uzun siparişlere ve daha yüksek artık uzunluklara sahip olan siparişlere öncelik tanınır. Bu yaklaşım, genellikle daha fazla fire üreten siparişlere odaklanarak, genel optimizasyon sürecini iyileştirmeyi amaçlar. Assign priority fonksiyonunun işleyişi aşağıda detaylandırılmaktadır:

Her bir sipariş türü için bir değer (V_i) ve öncelik (Pr_i) hesaplanır. Bu değerler, siparişin uzunluğu, fire miktarına etkisi ve çözüm zorluğu gibi faktörlere dayalı olarak belirlenir. Daha uzun siparişler ve daha yüksek artık uzunluklara sahip olan siparişler, daha yüksek öncelik ve değer alır.

Optimizasyon sırasında, algoritma mevcut kesim desenleri arasından seçim yapar. Eğer iki kesim deseni aynı fire miktarına sahipse, algoritma, bu desenlerin birleşik öncelik ve değer toplamını dikkate alır. Daha yüksek toplam değere sahip olan desen, öncelikli olarak seçilir. Bu strateji, doğal olarak optimize edilmesi daha zor olan siparişlerin önceliklendirilmesini sağlar.

Algoritma, daha uzun siparişlere ve daha yüksek artık uzunluklara sahip olan siparişlere öncelik tanıyarak, çözümü daha karmaşık olan öğelere odaklanır. Bu yaklaşım, optimizasyon sürecinin genel etkinliğini artırır ve fire miktarını minimize etmeye yardımcı olur.

Açgözlü strateji, küçük boyutlu siparişlerin hızlı işlenmesini sağlasa da bu stratejinin olumsuz etkilerini dengelemek için daha uzun siparişlere öncelik verilmesi gereklidir. Bu dengeleme, optimizasyon sürecinin daha verimli ve etkili ilerlemesini sağlar.

Özetle bu yöntem, optimizasyon sürecinde daha zor işlenebilen siparişlere odaklanarak, genel çözüm kalitesini artırmayı hedefler. Siparişlerin değer ve önceliklerine dayalı bu stratejik yaklaşım, yalnızca fire miktarını azaltmakla kalmaz, aynı zamanda daha dengeli ve sürdürülebilir bir çözüm süreci sunar. Böylece, algoritma hem kısa vadeli hem de uzun vadeli hedefleri göz önünde bulundurarak etkin bir çözüm stratejisi benimser.

3.2.1. Değer parametresi

Bu fonksiyon her bir sipariş uzunluğu için değer parametresini hesaplar. Değer parametresinin amacı, daha uzun siparişlere daha fazla önem atamaktır. Fonksiyonun ana yapısı Sözdekod 3'te açıklanmıştır.

Sözdekod 3. Her bir siparişin değerini hesaplayan fonksiyon.

GİRDİ: stok uzunluğu (SL), sipariş uzunlukları l_i
ÇIKTI: Her bir öğe uzunluğu için değer parametresi (V_i)

```
Vi listesi = BOŞ_LISTE
HER bir li İÇİN siparis_uzunluklari İÇİNDE
  r = li / stok_uzunlugu
  EĞER r <= 0.25 İSE
    Vi = li
  YOKSA EĞER r > 0.25 VE r <= 0.50 İSE
    Vi = (1 + r * 0.2) * li
  YOKSA
    Vi = 2 * li
```

3.2.2. Öncelik parametresi

Bu fonksiyon, ortalama uzunluğu (*ortalama_uzunluk*) sipariş uzunluklarının ağırlıklı ortalaması olarak ve ortalama artık uzunluğu (*ortalama_artık*) her bir siparişin artık uzunluğunun ortalaması olarak hesaplar. Örneğin, stok uzunluğu 6800 mm ve bir sipariş uzunluğu 800 mm ise, bu sipariş için artık uzunluk, 6800 mm'lik stok uzunluğunun 800 mm'lik sipariş uzunluğuna bölünmesiyle elde edilir ve 400 mm'lik bir artık uzunluk ile sonuçlanır (800 × 8 + 400 = 6800). Benzer şekilde, tüm siparişler için artık uzunluklar hesaplanır ve son olarak ortalama artık uzunluk bulunur. Bu parametrenin amacı daha yüksek artık uzunluğa sahip olanlara öncelik vermektir.

Öncelik parametresi yalnızca A algoritmasında kullanılır. A algoritmasında tanıtılan öncelik katsayı kümesi (pc) elemanları ile çarpılarak öncelik parametresinin farklı değerleri keşfedilir. Fonksiyonun ana yapısı, Sözdekod 4'te açıklanmıştır.

Sözdekod 4. Her bir siparişin önceliğini hesaplayan fonksiyon.
GİRDİ: stok uzunluğu (SL), sipariş uzunlukları ($l_{(i)}$)
ÇIKTI: Her bir sipariş uzunluğu için değer parametresi (V_i)

Pr_i listesi = BOŞ_LISTE

HER bir l_i İÇİN siparis_uzunluklari İÇİNDE

$artik_i$ = stok_uzunlugu MOD l_i

EĞER l_i <= ortalama_uzunluk VE $artik_i$ <= ortalama_artik İSE

Pr_i = -2

YOKSA EĞER l_i <= ortalama_uzunluk VE $artik_i$ > ortalama_artik İSE

Pr_i = -1

YOKSA EĞER l_i > ortalama_uzunluk VE $artik_i$ <= ortalama_artik İSE

Pr_i = 1

YOKSA

Pr_i = 2

3.3. Algoritma A

Algoritma A'nın ana yapısı Sözde Kod 5'te açıklanmıştır.

Sözdekod 5. Algoritma A’nın ana yapısı.

GİRDİ:

- orijinal_sipariş_miktarları: Başlangıçtaki sipariş taleplerini içeren vektör
- G: Mevcut nesil numarası (deneme sayısı)

ÇIKTI:

- Stok kullanımı, desen sayısı, toplam maliyet ve kesim planını içeren uygulanabilir bir çözüm

1. Kalan sipariş miktarlarının başlatılması:

   kalan_sipariş_miktarları ← orijinal_sipariş_miktarları

2. Öncelik katsayılarının tanımlanması:

   pc ← {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 20, 30, 40, 50}

3. Ana döngü:

   İken ∑(kalan_sipariş_miktarları) > 0:

      4. LP modelinin oluşturulması ve çözümü:

         LP ← Create_and_Solve_LP()

      5. Tam sayı bölme sonucunun hesaplanması:

         quotients ← [kalan_sipariş_miktarları / lp_çözüm]

      6. Minimum değerin belirlenmesi:

         min_değer ← minimum(quotients)

      7. Kalan sipariş miktarlarının güncellenmesi:

         kalan_sipariş_miktarları ← kalan_sipariş_miktarları - min_değer × lp_çözüm

      8. Önceliklerin yeniden atanması:

         Assign_Priority() fonksiyonunu çağır

Algoritma, *kalan_sipariş_miktarları* değişkenini, orijinal sipariş taleplerini temsil eden *orjinal_sipariş_miktarları* değerleriyle aynı yapar. Bu, çözüm sürecinde kalan taleplerin izlenmesini sağlar. Öncelik katsayısı kümesi (pc), her biri farklı bir çözüm kalitesini üretmeyi hedefleyen 20 farklı değerden oluşan bir küme ile tanımlanır. Deneysel analizler, ilk 15 parametreden sonra çözüm kalitesinin genellikle kötüleştiğini ve desen çeşitliliğinin azaldığını göstermektedir. Bu nedenle, öncelik katsayılarının sayısı 20 ile sınırlandırılmıştır.

Algoritma, *kalan_sipariş_miktarları* toplamı sıfırdan büyük olduğu sürece, 4 ila 8 numaralı adımları yineleyen bir döngüye girer. Bu döngü, kalan talepler tamamen karşılanana kadar yeni desenler üretmeyi amaçlar. Her yinelemede aşağıdaki adımlar gerçekleştirilir:

Create_and_Solve_Lp fonksiyonu, G'ninci nesilde bir tamsayılı doğrusal program (*Integer Linear Programming- ILP*) modeli oluşturur ve çözer. Bu model, seçilen öncelik katsayısı (pc[G]) ile Assign_Priority fonksiyonu tarafından hesaplanan değerler (V_i) ve öncelikler (Pr_i) kullanılarak tanımlanır. Amaç fonksiyonu ve kısıtlar aşağıda ifade edilmiştir:

$$max(\sum_{i=1}^{m}((v_i + pc[G]Pr_i) * x_i)) \quad (3.1)$$

$$\sum_{i=1}^{m} l_i x_i \leq SL \quad (3.2)$$

$$x_i \leq r(i) \quad (3.3)$$

$$i=1...m$$

Denklem 3.1, değerler (V_i) ve önceliklerin (Pr_i) farklı katsayılarla ağırlıklandırılmış toplamını maksimize etmeyi hedefler. Kısıt 3.2, desene dahil edilen ürün taleplerinin toplam uzunluğunun stok uzunluğunu (SL) aşmamasını sağlar. Kısıt 3.3, her bir siparişin kesim miktarının, ilgili siparişin kalan talebini (r_i) geçmemesini garanti eder. Bulunan çözüm, *lp_solution* değişkeninde saklanır ve x_i her bir sipariş için kesim miktarlarını temsil eder.

Algoritma, *lp_solution* vektörünün eleman bazında tamsayılı bölümünü hesaplayarak bir bölümler listesi değişkeni oluşturur. Bu liste, her siparişin desen içindeki potansiyel kullanım oranlarını gösterir. Daha sonra, bu listedeki minimum değer (*min_değer*), desenin frekansını (tekrar sayısını) belirlemek için kullanılır. Ardından, *kalan_sipariş_miktarları* aşağıdaki formulle güncellenir:

*kalan_siparis_miktarları=kalan_siparis,_miktarları−min_değer*lp_çözüm*

Bu işlem, kalan taleplerin, üretilen desenin frekansı kadar azaltılmasını sağlar. Güncellemenin ardından, önceliklerin kalan taleplerde yeniden hesaplanması için Assign_Priority fonksiyonu tekrar çağrılır.

kalan_sipariş_miktarları değişkeni toplamı sıfıra ulaştığında, döngü sona erer. Bu noktada, 4. adımda üretilen desenler Q kümesine eklenir ve mevcut çözümün toplam maliyeti (Z) hesaplanır. Bu maliyet, stok kullanımı ve desen sayısına dayalı olarak değerlendirilir.

Yukarıdaki süreç, her biri farklı bir öncelik katsayısı (pc[G]) kullanan 20 nesil boyunca tekrarlanır. Bu yinelemeler, farklı ağırlıklandırma stratejileriyle çeşitli ve kaliteli

çözümler üretilmesini sağlar. Her nesilde elde edilen desenler ve maliyetler, nihai çözümün optimizasyonu için bir temel oluşturur.

3.4. Algoritma B

B algoritmasının ana yapısı Sözdekod 6'da açıklanmıştır.

Sözdekod 6. Algoritma B'nin ana yapısı.

GİRDİ:

- fire_artış: Fire artış katsayısı
- G: Mevcut nesil numarası
- orijinal_sipariş_miktarları: Başlangıçtaki sipariş talepleri vektörü
- stok_uzunluğu: Kullanılabilir stok uzunluğu (SL)
- sipariş_uzunlukları: Her siparişin uzunluklarını içeren vektör

ÇIKTI:

Stok kullanımı, desen sayısı, toplam maliyet ve kesim planını içeren uygulanabilir bir çözüm

1. Başlangıç parametrelerinin ayarlanması:

  Eğer G < 30 ise:

    min_desen ← 2

    fire_izin ← (G - 1) × fire_artış

  Aksi takdirde (G ≥ 30):

    min_desen ← 1

    fire_izin ← (G - 31) × fire_artış

2. Kalan sipariş miktarlarının başlatılması:

  kalan_sipariş_miktarları ← orijinal_sipariş_miktarları

3. Ana döngü:

  İken ∑(kalan_sipariş_miktarları) > 0:

    4. max_talep ← maksimum(kalan_sipariş_miktarları)

    5. İç döngü:

Her m ∈ {max_talep, max_talep - 1,..., min_desen} için:

6. revize_kalan_sipariş_miktarları ← ⌊kalan_sipariş_miktarları / m⌋

7. Koşul kontrolü:

Eğer ∑(revize_kalan_sipariş_miktarları × sipariş_uzunlukları) ≥ stok_uzunluğu - fire_izin ise:

LP ← Create_and_Solve_LP()

8. LP çözüm değerlendirmesi:

Eğer LP optimal bir çözüm sunuyorsa:

kalan_sipariş_miktarları ← kalan_sipariş_miktarları - m × lp_çözüm

Eğer G < 30 ise:

İç döngüden çık

9. Parametre güncellemesi:

Eğer ∑(revize_kalan_sipariş_miktarları × sipariş_uzunlukları) < stok_uzunluğu - fire_izin

ve (m = min_desen veya m = 1) ise:

min_desen ← 1

fire_izin ← fire_izin + fire_artış

Aşağıda, Algoritma B'nin işleyişi adım adım açıklanmıştır:

Algoritma B, mevcut nesil sayısını temsil eden G değerini kontrol ederek başlar. Bu kontrol, iki farklı senaryoya göre parametrelerin tanımlanmasını sağlar:

Eğer G<30 ise, minimum desen sayısı (*min_desen*) 2 olarak ayarlanır ve izin verilen fire miktarı (*fire_izin*) aşağıdaki formulle hesaplanır:

$$fire\_izin=(G-1)x\ fire\_artış$$

Eğer G≥30 ise, minimum desen sayısı (*min_desen*) 1 olarak belirlenir ve izin verilen fire miktarı fire_izin aşağıdaki formulle güncellenir:

$$fire\_izin=(G-31)x\ fire\_artış$$

Ardından, *kalan_sipariş_miktarları* değişkeni, orijinal sipariş taleplerini temsil eden *orjinal_sipariş_miktarları* değerleriyle eşitlenir.

Algoritma, *kalan_sipariş_miktarları* toplamı sıfırdan büyük olduğu sürece devam eden bir ana döngüye girer. Bu döngü, kalan taleplerin tamamı karşılanana kadar desen üretimini sürdürmeyi sağlar. Döngü içindeki adımlar aşağıda açıklanmaktadır:

Ana döngü içinde, *kalan_sipariş_miktarları* listesinden maksimum talep (*max_talep*) belirlenir. Daha sonra, bu değerden (*min_desen - 1*)'e kadar olan m değerleri üzerinde bir iç döngü başlatılır. İç döngüde, *kalan_sipariş_miktarları* vektörüne eleman bazında taban bölme işlemi uygulanarak revize edilmiş kalan sipariş miktarları (*revize_kalan_sipariş_miktarları*) hesaplanır:

$$revize\_kalan\_siparis\_miktarları = kalan\_siparis,\_miktarları / m$$

Hesaplanan talepler ve bunların uzunlukları, stok uzunluğu (SL) ve izin verilen fire (*fire_izin*) ile ilgili bir uygunluk koşuluna tabi tutulur. Bu koşul, desenin fiziksel ve operasyonel sınırlar içinde kalmasını sağlar. Koşul karşılanırsa, Create_and_Solve_Lp fonksiyonu çağrılarak bir tamsayılı doğrusal program (ILP) modeli oluşturulur ve çözülür. Model aşağıda tanımlanmaktadır:

$$max(\sum_{i=1}^{m}(v_i * x_i)) \quad (3.4)$$

$$\sum_{i=1}^{m} l_i x_i \leq SL \quad (3.5)$$

$$x_i \leq r_{(i)} \quad (3.6)$$

$$i = 1 \ldots m$$

Denklem 3.4, her siparişin değerlerini (V_i) maksimize etmeyi amaçlar. Kısıt 3.5, desendeki toplam uzunluğun stok uzunluğunu aşmamasını sağlar. Kısıt 3.6, her bir siparişin kesim miktarının (x_i) kalan talebi (r_i) geçmemesini garanti eder. Çözüm, *lp_solution* değişkeninde saklanır.

- Eğer *lp_solution* optimal bir çözüm sunarsa, *kalan_sipariş_miktarları* bu çözüme göre güncellenir ve G<30 koşulu sağlanıyorsa iç döngü sonlandırılır. Bu, erken nesillerde hızlı ilerlemeyi teşvik eder.
- Eğer çözüm optimal değilse ve 7. maddede belirtilen koşul karşılanmıyorsa, *min_desen* ve *fire_izin* değerleri hesaplanan sonuçlara göre yeniden ayarlanır. Bu uyarlama, algoritmanın esnekliğini artırır.

Döngü sürecinde *kalan_sipariş_miktarları* toplamı sıfıra ulaştığında, ana döngü sona erer. Bu aşamada, üretilen desenler Q kümesine eklenir ve mevcut çözümün toplam maliyeti (Z) hesaplanır.

Algoritma B'nin tüm süreci, her biri farklı bir fire_izin değeriyle çalışan 50 nesil boyunca tekrarlanır. İterasyon sayısının 50 olarak belirlenmesi, çözüm etkinliği ile hesaplama süresi arasında optimal bir denge sağladığı için tercih edilmiştir. Deneysel analizler, iterasyon sayısının bir birim artmasının işlem süresini 0,5 ile 3 saniye arasında arttırdığını göstermektedir. Ayrıca, *fire_artış* parametresi, stok başına beklenen fire uzunluğu ve toplam deneme sayısına bağlı olarak dinamik olarak hesaplanır. Bu dinamik yapı, algoritmanın kullanıcı ihtiyaçlarına uygun farklı problem senaryolarına uyum sağlama yeteneğini güçlendirir.

3.5. ILP

Bu aşamada sütün üretimi, Algoritma A ve Algoritma B adımlarında üretilen farklı desenler kullanılarak ek bir uygulanabilir çözüm üretilir. Aşağıda verilen tamsayılı doğrusal programlama (*Integer Linear Programming - ILP*) formülasyonu, belirlenen 45 saniyelik süre içerisinde çözülür. 45 saniye süre sınırı test verisinin özelliklerine göre deneysel olarak belirlenmiştir. Diğer problem tiplerinde kullanıcı bu süreyi arttırıp azaltabilir. Burdaki kısaltmalar, Cui ve ark. (2015) tarafından kullanılan notasyonlarla uyumlu olarak tanımlanmıştır.

Z_{ILP}: Amaç değeri

Q: Farklı desenlerin kümesi, $Q=\{Q_1,...,Q_N\}$

N: Q'daki desen sayısı

u_B: Stok başına maliyet

u_S: Kurulum başına maliyet

x_J: Desenin frekansı Q_j

ε_J: $Q_{(j)}$'nin kullanılıp kullanılmadığını ($\varepsilon_{(j)}=1$) veya kullanılmadığını ($\varepsilon_{(j)}=0$) gösteren ikili değişken

$q_{(iJ)}$): qj desenindeki i tipi öğelerin miktarı

M: Bir modelin maksimum frekans sınırı

I: Negatif olmayan tam sayılar kümesi

$$Z_{ilp} = min(u_b \sum_{j=1}^{N} x_j + u_s \sum_{j=1}^{N} \varepsilon_j) \quad (3.7)$$

$$\sum_{j=1}^{N} q_{ij} x_j \geq d_i \quad (3.8)$$

$$x_j \leq M\varepsilon_j \quad (3.9)$$

$$x_j \in I \quad (3.10)$$

$$\varepsilon_j \in \{0,1\}, j = 1...N$$

Denklem 3.7, malzeme ve kurulumla ilgili toplam maliyetleri en aza indirmeyi amaçlayan amaç fonksiyonudur. Kısıt 3.8, ürün taleplerinin karşılanması gerekliliğini sağlar. Kısıt 3.9, Q_j olarak ifade edilen modele yalnızca ε_j 1'e eşit olduğunda izin verilmesini sağlar. Aksi belirtilmedikçe kıyaslama hesaplamalarında malzeme maliyeti malzeme uzunluğuna eşit olarak (u_b = SL) ve desen maliyeti is 100 olarak (u_s =100) kullanılmıştır.

4. KARŞILAŞTIRMA DENEYLERİ SONUÇLARI

Karşılaştırma deneyleri, Intel Core i7-1165G7 işlemci (2.80 GHz), 16 GB RAM ve Windows işletim sistemi ile donatılmış bir dizüstü bilgisayar üzerinde gerçekleştirilmiştir. Uygulama, Python programlama dili kullanılarak geliştirilmiş ve PULP-CBC-CMD çözücü motoru ile çalıştırılmıştır. Deneyler sırasında, ortalama hesaplama süresi 165 saniye olarak ölçülmüş, örnek başına maksimum süre ise 230 saniye olarak kaydedilmiştir.

OPTICUT'ın performansı, CUTGEN1 (Gau & Wäscher, 1995) ve Umetani (2008) tarafından sağlanan toplam 1840 problem örneği üzerinden karşılaştırılmıştır. Ayrıca, algoritmanın etkinliğini daha iyi göstermek amacıyla üç örnek uygulama sunulmuştur. Bu deneysel veriler, OPTICUT'ın geniş bir problem seti üzerinde değerlendirilmesini ve algoritmanın çeşitli koşullar altında ne kadar verimli çalıştığını ortaya koymaktadır.

4.1. Örnek Uygulama 1

Bir ambalaj firması, çeşitli boyutlarda depoladığı yarı mamul stokları, müşterilerden gelen belirli boyutlardaki siparişleri karşılamak amacıyla kesmektedir. Bu durum, Dyckhoff (1990) tarafından önerilen stok kesim problemleri sınıflandırmasına göre 1/V/I/M tipinde bir problem olarak tanımlanabilir. Bu sınıflandırma aşağıda açıklanmaktadır:

- 1 (tek boyutlu): Kesim işlemi yalnızca bir boyutta gerçekleştirilir (örneğin, uzunluk).
- V (tüm siparişlerin karşılanması): Mevcut kesim desenleri, tüm müşteri taleplerini tam olarak karşılamak zorundadır; kısmi karşılamalar veya eksik teslimatlar kabul edilmez.
- I (birden fazla aynı ebatta stok malzemesi): Kullanılabilir stoklar, aynı boyutta birden fazla birim içermektedir. Bu, çözümde stok miktarının da dikkate alınması gerektiği anlamına gelir.

- M (siparişlerin benzer boyutlarda olması): Müşteri talepleri, birbirine yakın ölçülerde olup, bu da desen oluşturma sürecinde çözümün karmaşıklığını artırabilir.

Bu kapsamda, stok verileri Tablo 4.1'de ilgili sipariş verileri ise Tablo 4.2'de sunulmuştur.

Tablo 4.1. Örnek uygulama 1 sipariş uzunluk ve miktarları.

Stok boyu	Stok adeti
6800	220

Tablo 4.2. Örnek uygulama 1 stok uzunluk ve miktarları.

Sipariş boyu	Sipariş adeti
1250	130
980	90
800	75
1500	25
685	110
1155	145
1000	150
1400	18
750	245
650	110
1365	30
995	60
890	110

Bu kesim problemi hem üretim planlaması hem de maliyet yönetimi açısından optimize edilmesi gereken çok boyutlu bir karar problemidir. Problemin temel kısıtları ve hedefi aşağıda detaylandırılmıştır.

Tüm müşteri siparişleri, eksiksiz olarak karşılanmalıdır. Kısmi teslimat veya eksik üretim kabul edilemez. Fazla üretim yapılabilir, fakat bu fire olarak kabul edilir.

Farklı bir desene geçiş yapıldığında, dilme makinesi durdurularak ayar yapılmalıdır. Bu durum her desen değişikliğinde sabit bir kurulum maliyeti doğurur.

Amaç, toplam maliyeti en aza indirmektir. Toplam maliyet iki bileşenden oluşur: malzeme maliyeti ve kurulum maliyeti.

Kullanılan her 1 mm malzeme için 1 TL ödenir. Her farklı desen için 1000 TL kurulum maliyeti oluşur. Aynı desen tekrar kullanıldığında bu maliyet oluşmaz. Toplam maliyet aşağıdaki formulle ifade edilebilir:

$$Toplam\ Maliyet = Malzeme\ Maliyeti + Kurulum\ Maliyeti$$

OPTICUT algoritması tarafından üretilen kesim planları Tablo 4.3'te verilmiştir. Tablo şu bilgileri içermektedir:

- Desen numarası
- Kullanılan stok boyu
- Desen başına oluşan fire miktarı
- Her desenin kaç kez kullanıldığı
- Net kullanılan uzunluk
- Hangi siparişlerin hangi stoklarda kesildiği

Elde edilen bu plan, siparişleri eksiksiz karşılarken, malzeme israfını azaltmakta, kurulum maliyetini en aza indirmekte ve stoğa dönüştürülebilir fireları verimli biçimde yöneterek toplam üretim maliyetini minimize etmektedir.

Tablo 4.3. Örnek uygulama 1 kesim planları.

Desen	Stok boyu	Fire	Kullanım adeti	Net kullanım	Desen
1	6800	0	60	6800	['1*1250', '1*980', '1*685', '2*1000', '1*995', '1*890']
2	6800	0	50	6800	['1*800', '2*1155', '2*750', '2*650', '1*890']
3	6800	0	30	6800	['2*1250', '1*685', '3*750', '1*1365']
4	6800	0	10	6800	['2*980', '1*685', '1*1155', '3*1000']
5	6800	0	10	6800	['1*1250', '1*685', '3*1155', '1*1400']
6	6800	0	10	6800	['1*800', '2*1500', '4*750']
7	6800	0	5	6800	['3*800', '1*1500', '1*1400', '2*750']
8	6800	5	3	6800	['3*980', '1*1155', '1*1400', '2*650']
9	6800	45	1	6800	['1*1155', '4*750', '4*650']
10	6800	3915	1	6800	['1*980', '1*1155', '1*750']

Tablo 4.4'te, her bir algoritma için kombinasyon istatistikleri ve maliyetler gösterilmektedir.

- Malzeme maliyeti, tüm paternler için aşağıdaki formül kullanılarak hesaplanmıştır:

$$Malzeme\ Maliyeti = Birim\ Stok\ Boyu \times Stok\ Adeti \times Birim\ Malzeme\ Maliyeti$$

- Patern geçişi sırasındaki hazırlık maliyeti, 10 farklı patern için her biri 1000 TL olmak üzere aşağıdaki formül ile hesaplanmıştır:

$$Hazırlık\ Maliyeti = 1000\ TL/patern \times 10\ patern = 10{,}000\ TL$$

- Toplam maliyet, aşağıdaki formül ile hesaplanmıştır:

$$Toplam\ Maliyet = Malzeme\ Maliyeti + Kurulum\ Maliyeti$$

Tablo 4.4. Örnek uygulama 1 kombinasyon istatistikleri ve maliyetler.

	OPTICUT	**COLUMN GENERATION**	**REAL CUT 1D**
Toplam fire (mm)	3,975	10,775	3,975
Desen Sayısı (adet)	10	16	19
Fire %	0.325	0.876	0.325
Malzeme maliyeti (TL)	816,000	822,800	816,000
Hazırlık Maliyeti (TL)	10,000	16,000	19,000
Toplam maliyet (TL)	826,000	838,800	835,000

OPTICUT algoritmasının performansı, Column Generation algoritması ve RealCut 1D yazılımı ile karşılaştırılmıştır.

- OPTICUT, %0.325'lik toplam fire oranı ile malzeme kullanımında yüksek etkinlik sağlarken, yalnızca 10 adet desen kullanarak hazırlık maliyetini minimize etmektedir.
- Column Generation algoritması, %0.876'lık yüksek toplam fire oranı ve 16 desenlik konfigürasyonu nedeniyle toplam maliyeti artırmaktadır.
- RealCut 1D, 19 desenle en yüksek operasyonel karmaşıklığa sahip olup, malzeme kullanımında %0.325 fire vermiştir.

Bu bulgular, endüstriyel kesim süreçlerinde OPTICUT algoritmasının ekonomik performans açısından üstünlüğünü açıkça ortaya koymaktadır.

4.2. Örnek Uygulama 2

Bu örnek uygulama, Wikipedia (2025) sayfasından alınmıştır. Bir kâğıt makinesi, her biri 5600 mm genişliğinde sınırsız sayıda ana rulo üretebilmektedir. Tablo 4.5'te belirtilen 13 farklı boya sahip siparişler, bu rulolardan kesilmelidir. Problem, talebi karşılayacak ve fireyi en aza indirecek şekilde, ana rulodan sipariş rulolarını üretmek için optimum bir kesim deseni seti bulmaktır. Örnek 1'de olduğu gibi: 1 mm malzeme maliyeti 1 TL'dir. Her bir farklı patern, 1000 TL kurulum maliyeti oluşturmaktadır.

Tablo 4.5. Örnek uygulama 2 sipariş uzunluk ve miktarları.

Sipariş boyu	Sipariş adeti	Sipariş boyu	Sipariş adeti
1380	22	2000	10
1520	25	2050	12
1560	12	2100	14
1710	14	2140	16
1820	18	2150	18
1880	18	2200	20
1930	20		

Opticut tarafından bulunan kesim planları Tablo 4.6'de sunulmuştur.

Tablo 4.6. Örnek uygulama 2 için OPTICUT kesim planları.

Desen	Stok boyu	Fire	Kullanım adeti	Net kullanım	Desen
1	5600	0	18	5600	['1*1520', '1*1930', '1*2150']
2	5600	0	5	5600	['1*1520', '1*1880', '1*2200']
3	5600	10	10	5590	['1*1710', '1*1880', '1*2000']
4	5600	20	12	5580	['1*1560', '1*1820', '1*2200']
5	5600	140	2	5460	['3*1820']
6	5600	20	7	5580	['1*1380', '2*2100']
7	5600	30	12	5570	['1*1380', '1*2050', '1*2140']
8	5600	10	2	5590	['1*1520', '1*1930', '1*2140']
9	5600	40	2	5560	['2*1710', '1*2140']
10	5600	140	3	5460	['1*1380', '1*1880', '1*2200']

Tablo 4.7, OPTICUT, Column Generation ve RealCut 1D yöntemlerinin kesim optimizasyonu performansını ve maliyetlerini karşılaştırmaktadır. Toplam fire, her üç

yöntemde de 1640 mm ile sabitken, desen sayısı OPTICUT'ta 10, Column Generation'da 13 ve RealCut 1D'de 14 olarak değişmektedir.

Tablo 4.7. Örnek uygulama 2 kombinasyon istatistikleri ve maliyetler.

	OPTICUT	COLUMN GENERATION	REAL CUT 1D
Toplam fire (mm)	1,640	1,640	1,640
Desen sayısı (adet)	10	13	14
Fire %	0.401	0.401	0.401
Malzeme maliyeti (TL)	408,800	408,800	408,800
Hazırlık maliyeti (TL)	10,000	13,000	14,000
Toplam maliyet (TL)	418,800	421,800	422,800

Fire yüzdesi (%0.401) ve malzeme maliyeti (408,800 TL) tüm yöntemlerde aynı kalırken, hazırlık maliyeti sırasıyla: OPTICUT için 10,000 TL, Column Generation için 13,000 TL, RealCut 1D için 14,000 TL olarak artmaktadır.

Bu durum, toplam maliyeti OPTICUT için 418,800 TL, Column Generation için 421,800 TL, RealCut 1D için 422,800 TL olarak hesaplanmasına neden olmaktadır. Sonuç olarak, OPTICUT algoritması, en düşük toplam maliyeti sunarak maliyet etkinliği açısından avantajlıdır.

4.3. Örnek Uygulama 3

Umetani ve diğerleri (2003) tarafından ortaya konan "Fiber29_9080" örnek uygulaması, literatürde önemli bir referans olarak yer almakta olup, Cui ve diğerleri (2015) tarafından da incelenmiştir. Tablo 4.8, toplamda 381 adet olan 20 farklı ürünün talep miktarlarını ve bu ürünlerin uzunluklarını detaylı bir şekilde listelemektedir. Söz konusu problemde, stok uzunluğu 9080 mm olarak belirlenmiştir. Çalışmanın temel amacı, makul bir işlem süresi içerisinde malzeme maliyetini, toplam maliyeti ve kullanılan desen sayısını en aza indirmek için etkin bir optimizasyon yaklaşımı geliştirmektir. Bu bağlamda, problem hem malzeme kullanım verimliliğini artırmayı hem de desen sayısını azaltmayı hedefleyen çok boyutlu bir optimizasyon meselesi olarak ele alınmaktadır.

Tablo 4.8. Örnek uygulama 3 sipariş uzunluk ve talep miktarları.

Uzunluk	Talep	Uzunluk	Talep
500	93	1020	30
550	12	1030	10
560	60	1040	6
680	10	1100	14
920	18	1120	5
940	2	1130	5
1000	50	1150	2
1160	6	1250	12
1210	24	1350	2
1220	12	1360	8

Toplam ürün miktarı, ana stok boyutuna bölünerek ana stok kullanımı için temel bir alt sınır elde edilir. İhtiyaç duyulan kümülatif ürün, aşağıdaki formulle hesaplanır:

$$500x93 + 550 \text{ x } 12 + \cdots + 1350 \text{ x } 2 + 1360 \text{ x } 8 = 317{,}250 \text{ mm}$$

Her ana stok 9080 mm ölçüsündedir ve minimum 34,94 stok gerekir (317,250/9,080) bu nedenle en az 35 stok gereklidir. Beklenen toplam fire ise aşağıdaki formulle hesaplanır:

$$(35-34{,}94) \text{ x } 9080 = 550 \text{ mm}$$

OPTICUT'ta fire artış parametresi 2 mm olarak ayarlanmıştır. Tablo 4.9, OPTICUT tarafından sağlanan optimum kesim planındaki her bir desenin desen sayısını, desen miktarını, kesim planını ve birim fireyi göstermektedir. OPTICUT, 35 adet stok kullanmış ve optimizasyon %0.173 fire oranıyla sonuçlanmıştır. Hesaplama 70 saniye sürmüştür. Çözüm, stok kullanımı açısından optimaldir çünkü daha önce hesaplandığı gibi stokun alt sınırını kullanmaktadır.

Desen sayısının optimalliği belirsizliğini korurken, bir hipoteze göre eşitlik kısıtlamaları olan tek boyutlu bir örnekte, fireyi en aza indiren bir çözüm n + 1'den fazla desen gerektirmez (burada n öğe sayısıdır). OPTICUT, Tablo 4.9'da gösterilen 6 desenli bir çözüm bulmuştur, bu da potansiyel olarak 21'den az desenli bir minimum fire çözümüne işaret etmektedir.

Tablo 4.9. Örnek uygulama 3 için OPTICUT çözümü.

Desen	Frekans	Kesim planı	Fire
1	15	[4*500, 2*560, 1*920, 3*1000, 2*1020]	0
2	10	[3*500, 1*560, 1*1030, 1*1100, 2*1210, 1*1220, 1*1250]	0
3	4	[2*560, 1*1000, 1*1040, 1*1100, 1*1120, 1*1130, 1*1210, 1*1360]	0
4	3	[2*550, 4*560, 3*680, 2*1160, 1*1360]	20
5	2	[1*500, 3*550, 1*920, 1*1040, 1*1150, 1*1220, 1*1250, 1*1350]	0
6	1	[1*500, 1*680, 1*920, 2*940, 1*1000, 1*1120, 1*1130, 1*1360]	490
Toplam	***35***		***550***

Tablo 4.10, mevcut probleme uygulanan CUI algoritmasının sonuçlarını göstermektedir. Aynı fire seviyesini 7 farklı desen kullanarak tespit etmiştir. Sonuç olarak, OPTICUT aynı fire seviyesinde 6 desenli bir çözüm elde ederek desen sayısını azaltmıştır.

Tablo 4.10. Örnek uygulama 3 için CUI çözümü.

Desen	Frekans	Kesim planı	Fire
1	12	[1*500, 2*560, 3*1000, 2*1020, 2*1210]	0
2	10	[5*500, 3*560, 1*680, 1*920, 1*1000,1*1030, 1*1250]	200
3	6	[3*500, 1*560, 1*1020, 1*1040, 1*1160,2*1220,	0
4	3	[1*500, 1*920, 1*1000, 4*1100, 1*1120,1*1130]	30
5	2	[2*500, 2*550, 1*920, 1*1100,1*1120, 1*1130, 1*1350, 1*1360]	0
6	1	[4*500, 5*550, 1*920, 1*1000,1*1150, 1*1250]	10
7	1	[2*500, 3*550, 2*920, 2*940,1*1150, 1*1250]	310
Toplam	***35***		***550***

4.5. Cutgen1 Kıyaslama Örnekleri

Gau ve Wäscher (1995) tarafından tek boyutlu stok kesme problemleri için örnek problemler üreten CUTGEN1 adlı bir yazılım geliştirilmiştir. Literatürde karşılaştırmalar genellikle bu yazılım tarafından oluşturulan 18 örnek problem seti kullanılarak yapılmıştır. Her bir problem seti m, r_1, r_2, d_{av} ve SL değişkenleri ile karakterize edilen 100 problemden oluşmaktadır. Karakteristiklerin açıklamaları aşağıdaki gibidir:

- m: Siparişlerdeki farklı uzunlukların sayısı.
- r_1: Sipariş uzunluklarının stok uzunluğuna minimum oranı.
- r_2: Sipariş uzunluklarının stok uzunluğuna maksimum oranı.
- d_{av}: Ortalama sipariş miktarı.
- SL: Stok uzunluğu.

Problem sınıfları üç gruba ayrılmaktadır: Grup 1, Grup 2 ve Grup 3. Grup 1, r_1 (0.01) ve r_2 (0.20) parametreleri için küçük değerlere sahip altı ürün sınıfını içermektedir. Grup 2, r_1 (0.01) ve r_2 (0.8) ile çeşitli özelliklere sahip altı ürün sınıfını içermektedir. Grup 3, özellikle r_1 (0.2) ve r_2 (0.8) olmak üzere daha büyük parametre değerlerine sahip altı ürün sınıfı içermektedir.

Tüm durumlarda, SL=1000 stok uzunluğu kullanılmış ve problemler 1995 tohum numarası ile üretilmiştir. Örneğin, Grup 1'de sipariş uzunlukları 10 (1000x0,01) ile 200 (1000x0,2) arasında eşit olarak dağıtılmıştır.

Tablo 4.11, CUTGEN1'in rastgele örnekleri için parametreleri göstermektedir.

Tablo 4.11. CUTGEN1'in rastgele örnekleri için parametreler.

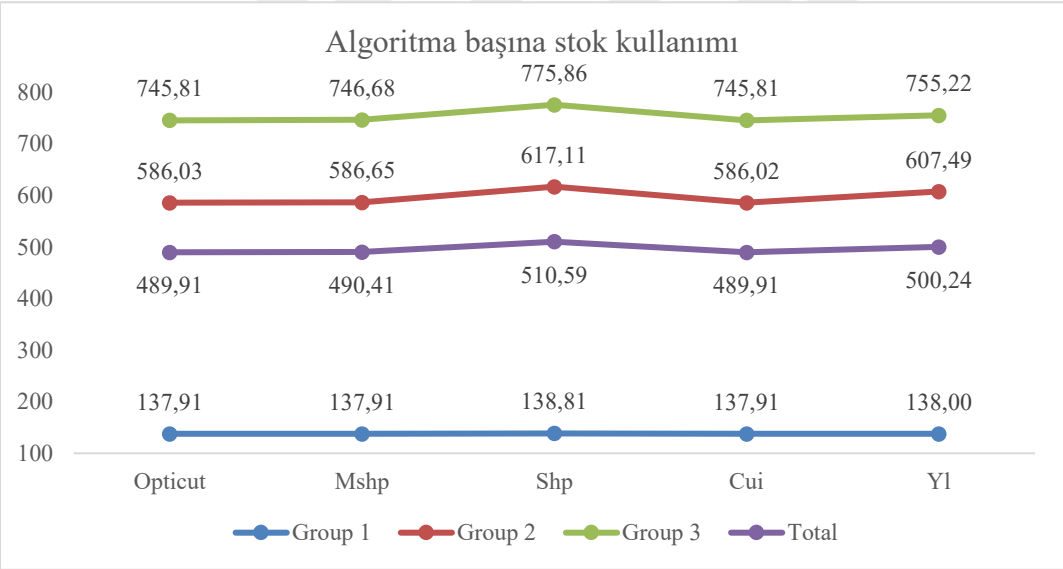
Sınıf	**m**	**r_1, r_2**	**d_{av}**	**Sınıf**	**m**	**r_1, r_2**	**d_{av}**	**Sınıf**	**m**	**r_1, r_2**	**d_{av}**
1	10		10	*7*	10		10	*13*	10		10
2	10		100	*8*	10		100	*14*	10		100
3	20	[0.01,0.2]	10	*9*	20	[0.01,0.8]	10	*15*	20	[0.2,0.8]	10
4	20		100	*10*	20		100	*16*	20		100
5	40		10	*11*	40		10	*17*	40		10
6	40		100	*12*	40		100	*18*	40		100
	Grup 1				**Grup 2**				**Grup 3**		

Her bir problem sınıfı için OPTICUT sonuçları, MSHP (Martin ve ark., 2018), SHP (Haessler, 1975), CUI (Cui ve ark., 2015) ve YL (Yanesse, 2006) algoritmaları ile karşılaştırılmıştır.

Ayrıca, Cui (2012), De Araujo ve ark. (2014), Renildo ve ark. (2009), Golfeto ve ark. (2009), Cui ve Liu (2011), Umetani ve ark. (2003), Foerster ve Wäscher (2000) de çalışmalarında aynı problemleri kullanmışlardır. Bu çalışmalara göre, MSHP ve CUI algoritmaları daha iyi performans gösterdikleri için karşılaştırma tablosuna dahil edilmemişlerdir.

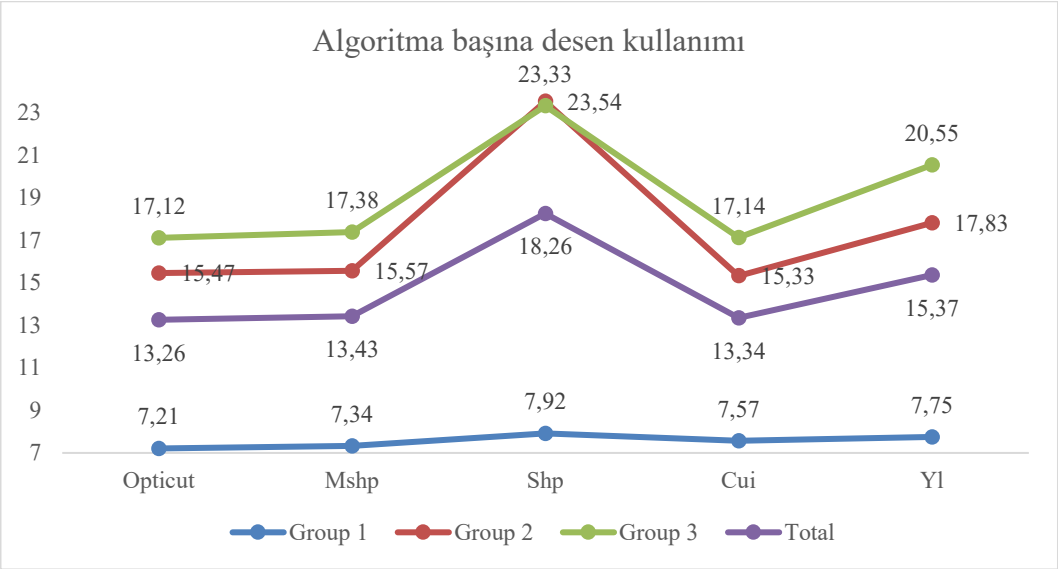
Şekil 4.1, Şekil 4.2 ve Şekil 4.3, sırasıyla algoritma ve grup başına ortalama stok kullanımını, desen kullanımını, toplam ortalama maliyeti göstermektedir.

Şekil 4.1'e göre stok kullanım açısından OPTICUT ve CUI için değerler oldukça yakındır. CUI en düşük stok kullanımına sahipken, OPTICUT stok kullanımı açısından ikinci sırada yer almaktadır.



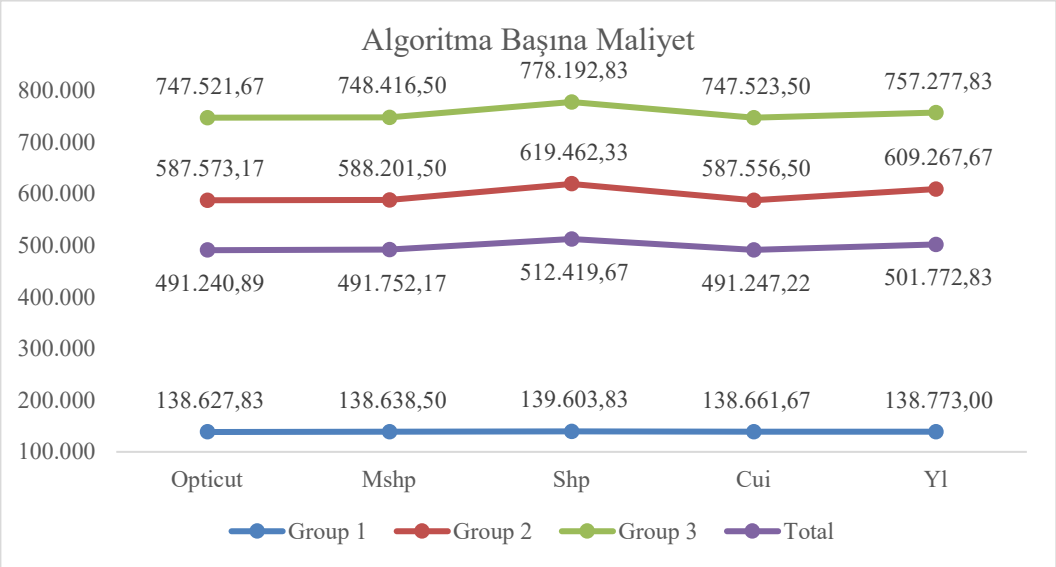
Şekil 4.1. Algoritma başına stok kullanımı.

Şekil 4.2'ye göre OPTICUT en düşük desen kullanımına sahipken, CUI desen kullanımı açısından ikinci sırada yer almaktadır.



Şekil 4.2. Algoritma başına desen kullanımı.

Algoritma başın amaliyetlerde Şekil 4.3' te yine OPTICUT en düşük maliyete sahipken, CUI maliyet açısından ikinci sırada yer almaktadır.



Şekil 4.3. Algoritma başına maliyet.

Tablo 4.12 ve Tablo 4.13, OPTICUT algoritmasının hesaplama sonuçlarını dört alternatif algoritmayla karşılaştırmalı olarak sunmaktadır. Diğer algoritmalar için sonuç değerleri, Martin ve ark. (2018)'den alınmıştır.

Tabloda 4.12'de, stok kullanımı, desen sayısı ve toplam maliyet için ortalama değerler sunulmaktadır. Bu sütunlar sırasıyla stok, desen ve maliyet olarak gösterilmektedir. Birim malzeme maliyeti 1000 ve desen maliyeti 100 olarak varsayılmıştır. Toplam

maliyet, malzeme maliyeti ve desen maliyetinin toplamı olarak hesaplanmıştır. Performans değerlendirmesi bağlamında, koyu yazılar üstün sonuçları gösteren baskın değerleri vurgular. Her sütundaki baskınlık (üstünlük) , her bir problem sınıfı için stok kullanımı, desen sayısı veya toplam maliyet arasından minimum değer seçilerek belirlenir. Baskınlık yüzdesi, her sütundaki baskın hücre sayısının toplam sınıf sayısına (bu durumda 18) bölünmesiyle hesaplanır. Bu, stok kullanımı, desen sayısı ve genel maliyet verimliliği açısından algoritmik üstünlüğün nicel bir ölçüsünü sağlar.

OPTICUT, ele alınan sınıflar arasında stok kullanımında %83 ve desen sayısında %50'lik bir üstünlük göstermektedir. Toplam maliyet açısından, OPTICUT sınıfların %67'sinde üstünlük sağlamaktadır. CUI, sınıfların %94'ünde stok kullanımında baskındır ve sınıfların %39'unda maliyet açısından verimlidir.

Özetle, OPTICUT hem desen sayısı hem de toplam maliyet ölçütlerinde üstünlük sağlayarak genel avantajlı performansının altını çizmektedir. Buna karşılık, CUI, özellikle stok kullanımı konusunda güçlü yönlerini sergilemektedir. CUI, stok kullanımında daha yüksek baskınlık gösterse de OPTICUT'ın desen sayısındaki üstünlüğü, toplam maliyeti düşürmede belirleyici olmuştur.

Tablo 4.13, gruplara göre kategorize edilmiş hesaplama sonuçlarını sunmakta ve OPTICUT'ın Grup 1, Grup 3 ve genel toplamlar açısından malzeme maliyeti, toplam maliyet ve desen sayısındaki üstünlüğünü göstermektedir.

Buna karşılık, CUI, genel toplamlar açısından minimum stok kullanımı sağlarken, Grup 2'de malzeme maliyeti, desen sayısı ve toplam maliyette üstün performans göstermektedir.

CUI ve OPTICUT arasındaki karşılaştırmalı analiz, OPTICUT'ın verimliliğini vurgulamaktadır. Bu analiz, desen kullanımında %0.6'lık bir azalma ve toplam maliyette %0.0013'lük bir düşüş ile sonuçlanmaktadır. Ancak, bu verimliliğe CUI'ye kıyasla stok kullanımında %0.0003'lük bir artış eşlik etmektedir.

Tablo 4.12. Her algoritma için CUTGEN1 sınıflarının hesaplama sonuçları.

	OPTICUT			MSHP			SHP			CUI			YL			
Sınıf	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**	**Grup**
1	11.49	3.46	**11.836**	**11,48**	3,61	11.841	11,56	4,49	12.009	11,49	3,73	11.863	11.56	**3.31**	11.891	
2	**110.25**	6.27	110.877	110,26	**6,13**	**110.873**	110,91	6,46	111.556	**110,25**	6,42	110.892	110.40	6.95	111.095	
3	**22.13**	5.15	**22.645**	**22,13**	5,16	22.646	22,18	5,90	22.770	**22,13**	5,42	22.672	22.17	**4.96**	22.666	
4	**215.93**	**8.25**	**216.755**	**215,93**	8,49	216.779	217,39	8,64	218.254	**215,93**	8,84	216.814	215.98	10.32	217.012	Grup 1
5	**42.95**	7.84	**43.734**	**42,95**	8,14	43.764	43,26	9,03	44.163	**42,95**	8,01	43.751	42.99	**7.63**	43.753	
6	424.69	**12.3**	**425.920**	**424,68**	12,48	425.928	427,57	13,01	428.871	**424,68**	12,98	425.978	424.89	13.31	426.221	
7	**50.24**	**6.4**	**50.880**	50,27	6,44	50.914	51,85	10,79	52.929	**50,24**	6,40	**50.240**	51.69	7.66	52.456	
8	**499.62**	7.05	500.325	499,64	7,19	500.359	518,39	11,48	519.538	**499,62**	7,03	**500.323**	502.23	9.62	503.192	
9	**93.65**	12.35	94.885	93,86	12,61	95.121	97,61	19,62	99.572	**93,65**	12,26	**94.876**	99.49	13.64	100.854	
10	**932.26**	**13.89**	**933.649**	932,53	14,27	933.957	976,07	21,54	978.224	**932,26**	14,03	933.663	948.41	18.21	950.231	Grup 2
1	176.93	25.11	179.441	177,64	24,77	180.117	186,99	37,28	190.718	**176,91**	24,21	**179.331**	195.67	24.60	198.130	
2	**1763.4**	**27.99**	**1.766.259**	1765,93	28,11	1.768.741	1871,74	40,53	1.875.793	**1763,46**	28,06	1.766.266	1847.42	33.23	1.850.743	
13	**63.47**	7.53	64.223	63,48	7,55	64.235	64,71	10,25	65.735	**63,47**	**7,52**	**64.222**	64.20	8.93	65.093	
14	**632.36**	8.16	633.176	**632,36**	8,16	633.176	646,82	10,76	647.896	**632,36**	**8,10**	**633.170**	633.26	10.51	634.311	
15	**119.59**	**14.15**	**121.005**	119,80	14,22	121.222	122,64	19,99	124.639	**119,59**	**14,15**	**121.005**	123.90	16.28	125.528	
16	**1192.0**	**15.62**	**1.193.562**	1192,47	15,99	1.194.069	1227,43	20,85	1.229.515	**1192,00**	15,66	1.193.566	1197.66	19.89	1.199.649	Grup 3
17	**224.85**	**27.08**	**227.558**	225,80	27,29	228.529	235,49	38,20	239.310	**224,85**	27,12	227.562	244.02	29.76	246.996	
18	**2242.5**	**30.16**	**2.245.606**	2246,16	31,08	2.249.268	2358,07	39,92	2.362.062	**2242,59**	30,26	2.245.616	2268.30	37.90	2.272.090	
Baskınlık	83%	**50%**	**67%**	33%	6%	6%	0%	0%	0%	**94%**	39%	39%	0%	17%	0%	

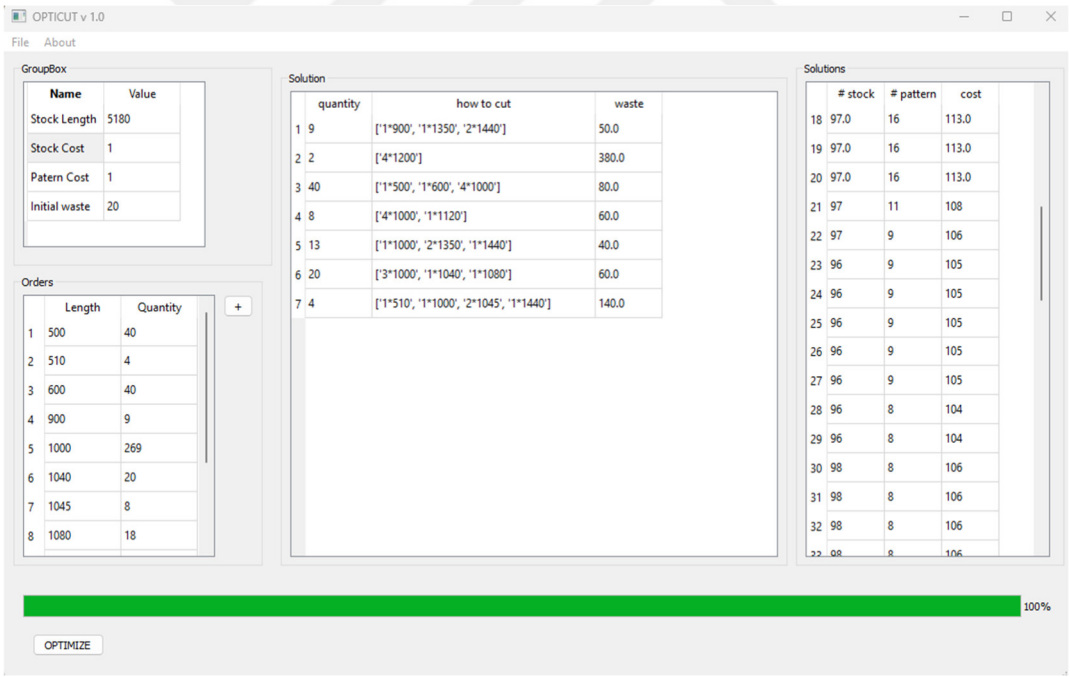
Tablo 4.13. Her algoritma için CUTGEN1 gruplarının hesaplama sonuçları.

	OPTICUT			MSHP			SHP			CUI			YL		
	Stok	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**
Grup 1	**137.91**	7.21	**138.627,83**	**137.91**	7.34	138.638,50	138.81	7.92	139.603,83	**137.91**	7.57	138.661,67	138.00	7.75	138.773,00
Grup 2	586.03	15.47	587.573,17	586.65	15.57	588.201,50	617.11	23.54	619.462,33	**586.02**	**15.33**	**587.556,50**	607.49	17.83	609.267,67
Grup 3	**745.81**	17.12	**747.521,67**	746.68	17.38	748.416,50	775.86	23.33	778.192,83	**745.81**	17.14	747.523,50	755.22	20.55	757.277,83
Ortalama	**489.91**	**13.26**	491.240,89	490.41	13.43	491.752,17	510.59	18.26	512.419,67	**489.91**	13.34	491.247,22	500.24	15.37	501.772,83
Toplam	881846	**23876**	**884.233.600**	882737	24169	885.153.900	919068	32874	922.355.400	**881843**	24020	884.245.000	900424	27671	903.191.100

4.6. Umetani kıyaslama örnekleri

Umetani ve ark. (2003), kimyasal elyaf endüstrisinden elde edilen 40 gerçek örnekten oluşan bir veri seti sunmuştur. Bu veri setine, Umetani (2018) çalışması aracılığıyla erişim sağlanabilmektedir.

Şekil 4.4'te fiber18 örneğine ilişkin OPTICUT algoritmasıyla elde edilen çözüm yer almaktadır. Örnekte kullanılan fiber18 ifadesi, m = 18 (ürün sayısı) ve SL (stok uzunluğu) parametresinin 5180 ya da 9080 olarak belirlendiğini ifade etmektedir. Siparişler bölümü, her bir siparişin uzunluk ve miktar bilgilerini içerecek şekilde detaylandırılmıştır. Çözümler bölümü ise, stok kullanımı, desen sayısı ve her bir çözümle ilişkili toplam maliyet gibi temel performans göstergelerini sunmaktadır. Bu bölümde, özellikle en düşük maliyetli çözüm gösterilmiştir. Söz konusu en iyi çözümde; 7 farklı desen kullanılmış, 96 birimlik stok tüketilmiş ve toplam maliyet 103 birim olarak hesaplanmıştır.



Şekil 4.4. Fiber18 örneği çözüm sonuçları (SL=5180, c_s=s_p=1).

OPTICUT, CUI ve MSHP stok uzunluğuna bağlı iki farklı parametre kümesi üzerinden karşılaştırmalı hesaplamalar yapmak amacıyla kullanılmıştır. Bu deneysel analizlerin bulguları, Martin ve ark. (2018) tarafından raporlanan diğer algoritmaların sonuçlarıyla birlikte Tablo 4.14 ve Tablo 4.15' te ayrıntılı olarak sunulmuştur. Bu tablolarda, üstün performans sergileyen değerler koyu biçimde vurgulanmıştır.

Tablo 4.14. Umetani örnekleri için algoritmaların hesaplama sonuçları (SL=5180, malzeme maliyeti=1, desen maliyeti=1).

	Opticut			Mshp			Cui		
Problem	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**
fiber06	34	**4**	**38**	**33**	5	**38**	34	**4**	**38**
fiber07	**33**	**3**	**36**	**33**	4	37	34	**3**	37
fiber08	**86**	**4**	**90**	**86**	**4**	**90**	**86**	6	92
fiber09	55	**4**	**59**	**54**	5	**59**	55	5	60
fiber10	**70**	**4**	**74**	**70**	**4**	**74**	**70**	6	76
fiber11	**67**	**5**	**72**	**67**	6	73	**67**	7	74
fiber13a	57	**5**	**62**	57	**5**	**62**	**56**	7	63
fiber13b	**28**	**4**	**32**	**28**	**4**	**32**	**28**	5	33
fiber14	49	**4**	**53**	49	**4**	**53**	**48**	6	54
fiber15	**57**	5	**62**	58	**4**	**62**	**57**	7	64
fiber16	**83**	**7**	**90**	84	**7**	91	**83**	8	91
fiber17	**83**	7	**90**	85	**5**	**90**	**83**	8	91
fiber18	**96**	**7**	**103**	**96**	8	104	**96**	11	107
fiber19	134	**7**	**141**	134	**7**	**141**	**133**	9	142
fiber20	**32**	**7**	**39**	**32**	8	40	**32**	8	40
fiber23	**142**	10	**152**	144	**8**	**152**	**142**	13	155
fiber26	**191**	**8**	**199**	**191**	9	200	**191**	12	203
fiber28a	**84**	9	**93**	86	**8**	94	**84**	11	95
fiber28b	**119**	**10**	**129**	**119**	**10**	**129**	**119**	12	131
fiber29	**62**	**8**	**70**	**62**	**8**	**70**	**62**	11	73
Toplam	1562	**122**	**1684**	1568	123	1691	**1560**	159	1719
Baskınlık	75%	**80%**	**100%**	60%	65%	70%	**85%**	10%	5%

Tablo 4.14, Stok ve desen maliyeti (c_s=s_p=1) 1 olarak belirlendiğinde, her bir algoritmanın Umetani veri seti üzerindeki hesaplama performanslarını sunmaktadır. Bu bağlamda, stok uzunluğu (SL) 5180 olarak alındığında elde edilen bulgular aşağıdaki şekilde özetlenebilir:

- OPTICUT algoritması, desen sayısı bakımından örneklerin %80'inde üstünlük sağlarken, toplam maliyet açısından tüm örneklerde (%100) en iyi performansı göstermektedir. OPTICUT hem en düşük desen kullanımı hem de en düşük toplam maliyet ile öne çıkmaktadır.
- CUI algoritması, stok kullanımında örneklerin %85'inde üstünlük sağlayarak bu metrikte en yüksek değeri elde etmektedir. Ancak, desen sayısı ve toplam maliyet açısından karşılaştırılan algoritmalar arasında en zayıf performansı sergilemektedir.

- OPTICUT, MSHP algoritmasına kıyasla %0.42 oranında daha düşük toplam maliyet üretmiştir. Bu oran aşağıdaki şekilde hesaplanmaktadır:

$$(1691-1684)\ /1684*100 = \%0.42$$

Bu sonuçlar, OPTICUT'ın özellikle toplam maliyet açısından yüksek etkinlik sağladığını ve stok kesim problemlerinde tercih edilebilirliğini desteklemektedir.

Tablo 4.15. Umetani örnekleri için algoritmaların hesaplama sonuçları (SL=9080, Malzeme maliyeti=1, desen maliyeti=1).

	Opticut			**Mshp**			**Cui**		
Problem	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**
fiber06	**19**	**3**	**22**	**19**	4	23	**19**	4	23
fiber07	**19**	**2**	**21**	**19**	3	22	**19**	4	23
fiber08	**48**	**3**	**51**	**48**	**3**	**51**	**48**	4	52
fiber09	**29**	**4**	**33**	**29**	**4**	**33**	30	4	34
fiber10	**39**	**4**	**43**	**39**	**4**	**43**	**39**	5	44
fiber11	**38**	**4**	**42**	**38**	**4**	**42**	**38**	5	43
fiber13a	**32**	**4**	**36**	**32**	5	37	**32**	5	37
fiber13b	**16**	**3**	**19**	**16**	**3**	**19**	**16**	4	20
fiber14	**27**	**4**	**31**	**27**	**4**	**31**	**27**	5	32
fiber15	**32**	**4**	**36**	**32**	**4**	**36**	**32**	5	37
fiber16	**47**	**6**	**53**	**47**	**6**	**53**	**47**	**6**	**53**
fiber17	**47**	**5**	**52**	**47**	**5**	**52**	**47**	**5**	**52**
fiber18	**54**	6	**60**	55	**5**	**60**	**54**	7	61
fiber19	**74**	**5**	**79**	**74**	**5**	**79**	**74**	7	81
fiber20	**19**	**5**	**24**	**19**	**5**	**24**	**19**	6	25
fiber23	81	**7**	88	**80**	**7**	**87**	81	8	89
fiber26	**107**	**6**	**113**	**107**	7	114	**107**	9	116
fiber28a	**48**	6	**54**	49	**5**	**54**	**48**	8	56
fiber28b	**67**	**7**	**74**	**67**	8	75	**67**	8	75
fiber29	**35**	**6**	**41**	**35**	7	42	**35**	7	42
Toplam	**878**	**94**	**972**	879	98	977	879	116	995
Baskınlık	**95%**	**90%**	**95%**	90%	70%	70%	90%	10%	10%

Tablo 4.15'te gösterilen stok uzunluğunun (SL) 9080 olarak belirlendiği durumda, OPTICUT algoritması tüm performans ölçütleri bakımından üstünlük sağlamaktadır. Bu kapsamda elde edilen bulgular aşağıda özetlenmektedir:

- Stok kullanımı açısından, değerlendirilen sınıfların %95'inde OPTICUT en iyi sonucu elde etmektedir.

- Desen sayısı bakımından, örneklerin %90'ında üstün performans sergilemektedir.
- Toplam maliyet açısından ise, sınıfların %95'inde en düşük değeri üretmiştir.

Bu bulgular, OPTICUT'ın yalnızca bir ölçüt üzerinde değil, tüm temel performans göstergelerinde baskınlık sağladığını ortaya koymaktadır. Algoritma, aynı anda en düşük stok kullanımı, en az desen kullanımı ve en düşük toplam maliyet ile öne çıkmaktadır. Ayrıca, OPTICUT algoritması, MSHP algoritmasına göre %0.51 oranında daha düşük toplam maliyet üretmiştir. Bu fark aşağıdaki gibi hesaplanmaktadır:

$$((977-972) / 972)*100 = \%0.51$$

Bu sonuçlar, OPTICUT'ın yüksek stok uzunluklarında da tutarlı ve etkili bir performans sergilediğini göstermektedir.

Tablo 4.16 ve Tablo 4.17, malzeme maliyetinin stok uzunluğuna eşit (c_s=SL) ve desen maliyetinin ise 100 (s_p=100) olduğu durumlarda, her bir algoritma için Umetani örneklerinin hesaplama sonuçlarını sunmaktadır. Bu veriler, aşağıdaki gibi özetlenebilir:

- Stok uzunluğu (SL) 5180 olduğunda:
    - OPTICUT, değerlendirilen sınıfların %80'inde desen sayısı bakımından, %100'ünde stok kullanımı açısından ve %95'inde toplam maliyet açısından üstünlük sağlamaktadır.
    - OPTICUT, en düşük stok kullanımı ve toplam maliyetiyle öne çıkmaktadır.
    - CUI, OPTICUT gibi minimum stok kullanımına sahiptir, ancak desen sayısı ve toplam maliyet açısından daha düşük performans sergilemiştir.
    - OPTICUT, CUI'ye kıyasla %0.05 oranında daha düşük toplam maliyet elde etmiştir. Bu fark aşağıdaki gibi hesaplanmaktadır:

$$((8,039,020-8,043,220) / 8,043,220)*100 = \%0.05$$

- Stok uzunluğu (SL) 9080 olduğunda:
    - OPTICUT, tüm performans ölçütlerinde üstünlük sağlamaktadır.

- Sınıfların %100'ü için stok kullanımı açısından, %85'i için desen sayısı açısından ve %85'i için toplam maliyet açısından üstünlük göstermektedir.
- OPTICUT, en düşük stok kullanımı, desen kullanımı ve toplam maliyetiyle öne çıkmaktadır.
- MSHP'ye kıyasla, %0.001 oranında daha düşük toplam maliyet elde edilmiştir. Bu fark aşağıdaki şekilde hesaplanmaktadır:

*((7.964.480-7.964.580) / 7.964.580)*100 = %0.001*

Bu bulgular, yine OPTICUT algoritmasının hem stok kullanımı hem de maliyet açısından diğer algoritmalara kıyasla sürekli olarak üstün performans gösterdiğini ve stok uzunluğu değişse bile etkili bir çözüm sunduğunu göstermektedir.

Tablo 4.16. Umetani örnekleri için algoritmaların hesaplama sonuçları (SL=5180, malzeme maliyeti=SL, desen maliyeti=100).

	Opticut			**Mshp**			**Cui**		
Problem	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**
fiber06	**33**	**5**	**171,440**	**33**	**5**	**171,440**	**33**	**5**	**171,440**
fiber07	**33**	**4**	**171,340**	**33**	**4**	**171,340**	**33**	**4**	**171,340**
fiber08	**86**	**4**	**445,880**	**86**	**4**	**445,880**	**86**	6	446,080
fiber09	**53**	**7**	**275,240**	**53**	**7**	**275,240**	**53**	**7**	**275,240**
fiber10	**69**	6	358,020	**69**	**5**	**357,920**	**69**	7	358,120
fiber11	**67**	**5**	**347,560**	**67**	**5**	**347,560**	**67**	7	347,760
fiber13a	**56**	**6**	**290,680**	**56**	7	290,780	**56**	7	290,780
fiber13b	**28**	**4**	**145,440**	**28**	**4**	**145,440**	**28**	5	145,540
fiber14	**47**	8	**244,260**	48	**5**	249,140	**47**	10	244,460
fiber15	**57**	**5**	**295,760**	**57**	**5**	**295,760**	**57**	8	296,060
fiber16	**82**	**9**	**425,660**	**82**	10	425,760	**82**	11	425,860
fiber17	**83**	**7**	**430,640**	**83**	8	430,740	**83**	8	430,740
fiber18	**96**	**8**	**498,080**	**96**	**8**	**498,080**	**96**	10	498,280
fiber19	**133**	10	689,940	**133**	**9**	**689,840**	**133**	**9**	**689,840**
fiber20	**32**	**7**	**166,460**	**32**	8	166,560	**32**	8	166,560
fiber23	**141**	**13**	**731,680**	**141**	**13**	**731,680**	**141**	18	732,180
fiber26	**190**	**12**	**985,400**	**190**	**12**	**985,400**	**190**	13	985,500
fiber28a	**83**	12	**431,140**	84	**9**	436,020	**83**	27	432,640
fiber28b	**118**	**12**	**612,440**	**118**	**12**	**612,440**	**118**	13	612,540
fiber29	**62**	**8**	**321,960**	**62**	**8**	**321,960**	**62**	11	322,260
Toplam	**1549**	152	**8,039,020**	1551	**148**	8,048,980	**1549**	194	8,043,220
Baskınlık	**100%**	**80%**	**90%**	90%	**80%**	70%	**100%**	20%	20%

Tablo 4.17. Umetani örnekleri için algoritmaların hesaplama sonuçları (SL=9080, malzeme maliyeti=SL, desen maliyeti=100).

	Opticut			Mshp			Cui		
Problem	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**	**Stok**	**Desen**	**Maliyet**
fiber06	**19**	4	172,920	**19**	4	172,920	**19**	**3**	**172,820**
fiber07	**19**	**2**	**172,720**	**19**	3	172,820	**19**	4	172,920
fiber08	**48**	**3**	**436,140**	**48**	**3**	**436,140**	**48**	4	436,240
fiber09	**29**	**4**	**263,720**	**29**	**4**	**263,720**	**29**	5	263,820
fiber10	**39**	**4**	**354,520**	**39**	**4**	**354,520**	**39**	5	354,620
fiber11	**38**	**4**	**345,440**	**38**	**4**	**345,440**	**38**	6	345,640
fiber13a	**32**	**4**	**290,960**	**32**	5	291,060	**32**	5	291,060
fiber13b	**16**	**3**	**145,580**	**16**	**3**	**145,580**	**16**	4	145,680
fiber14	**27**	**4**	**245,560**	**27**	**4**	**245,560**	**27**	5	245,660
fiber15	32	4	290,960	32	4	290,960	32	5	291,060
fiber16	**47**	**6**	**427,360**	**47**	**6**	**427,360**	**47**	**6**	**427,360**
fiber17	**47**	**5**	**427,260**	**47**	**5**	**427,260**	**47**	6	427,360
fiber18	**54**	**6**	**490,920**	**54**	**6**	**490,920**	**54**	7	491,020
fiber19	**73**	**12**	**664,040**	**73**	**10**	**663,840**	**73**	11	663,940
fiber20	**19**	**5**	**173,020**	**19**	**5**	**173,020**	**19**	6	173,120
fiber23	**80**	**9**	**727,300**	**80**	**7**	**727,100**	**80**	9	727,300
fiber26	**107**	**6**	**972,160**	**107**	7	972,260	**107**	9	972,460
fiber28a	**48**	**6**	**436,440**	**48**	**6**	**436,440**	**48**	7	436,540
fiber28b	**67**	**7**	**609,060**	**67**	8	609,160	**67**	8	609,160
fiber29	**35**	**6**	**318,400**	**35**	7	318,500	**35**	7	318,500
Toplam	**876**	**104**	**7,964,480**	**876**	105	7,964,580	**876**	122	7,966,280
Baskınlık	**100%**	**85%**	**85%**	**100%**	70%	70%	**100%**	10%	10%

4.7. Duyarlılık Analizi

OPTICUT algoritması, parametre değişimlerine karşı bir miktar hassasiyet göstermektedir. Bu bölüm, deneysel analiz yoluyla bu iddiaya kanıt sağlamaktadır. Çalışmada, CUTGEN1'den üç grup kullanılmış ve her gruptan 50 rastgele örnek seçilerek toplamda 150 örnek kullanılmıştır. Birim malzeme maliyeti 1000 ve birim desen maliyeti 100 olarak varsayılmıştır.

OPTICUT, üç ana parametre kullanmaktadır:

1. Değer (V_i): Daha uzun öğelere öncelik vermek için kullanılır.
2. Öncelik (Pr_i): Optimize edilmesi zor öğelere öncelik vermek için kullanılır.
3. Fire artışı: İzin verilen fire miktarı dahilinde uygulanabilir bir çözüm aramak için iterasyonlarda kullanılır.

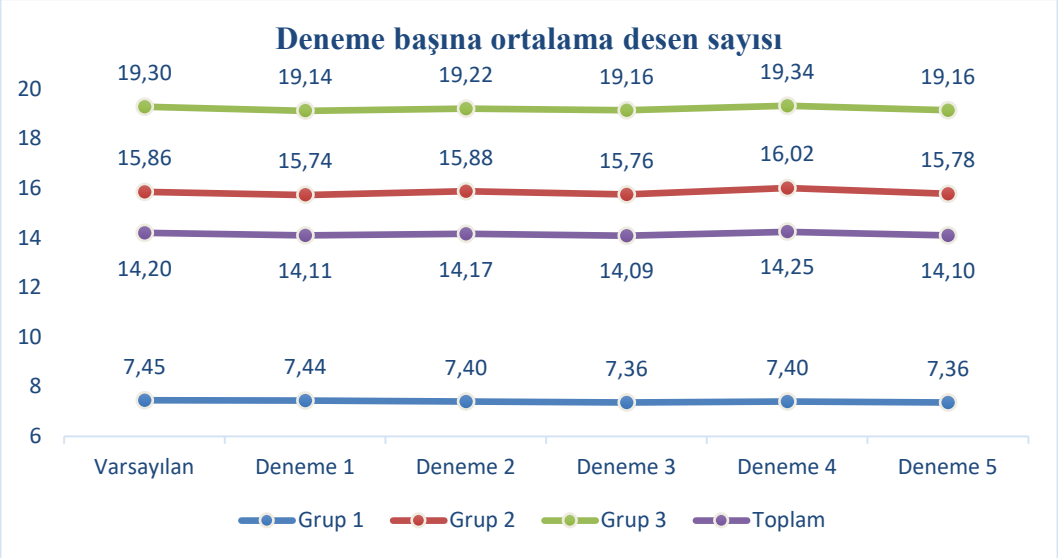
Varsayılan senaryoda değer parametresi 0.2, öncelik parametresi 20 farklı öncelik değeri, fire artış parametresi 50 farklı değere ayarlanmıştır. Deneysel senaryolar ise aşağıda tanımlanmıştır:

- Deneme 1: Değerler 0.2, 30 ve 150 olarak ayarlanmıştır.
- Deneme 2: Değerler 0.4, 30 ve 150 olarak ayarlanmıştır.
- Deneme 3: Değerler 0.6, 30 ve 150 olarak belirlenmiştir.
- Deneme 4: Değerler 1, 30 ve 150 olmuştur.
- Deneme 5: Değerler sırasıyla 0, 30 ve 150 olarak ayarlanmıştır.

Tablo 4.18 ve Şekil 4.5, bu deneylerin sonuçlarını göstermektedir. Tablo 4.18'de yer alan sonuçlara göre, desen sayısı değişirken stok kullanımı tüm denemelerde aynı kalmaktadır. Tipik olarak, deneme sayısı arttıkça desen kullanımının verimliliği de artmaktadır.

Tablo 4.18. Her deneme senaryosu için OPTICUT'ın hesaplama sonuçları.

	Varsayılan		**Deneme 1**		**Deneme 2**	
Grup	**Stok**	**Desen**	**Stok**	**Desen**	**Stok**	**Desen**
1	132.70	7.45	132.70	7.44	132.70	7.40
2	734.32	15.86	734.32	15.74	734.32	15.88
3	798.20	19.30	798.20	19.14	798.20	19.22
Ort.	555.07	14.20	555.07	14.11	555.07	14.17
	Deneme 3		**Deneme 4**		**Deneme 5**	
Grup	**Stok**	**Desen**	**Stok**	**Desen**	**Stok**	**Desen**
1	132.70	7.36	132.70	7.40	132.70	7.36
2	734.32	15.76	734.32	16.02	734.32	15.78
3	798.20	19.16	798.20	19.34	798.20	19.16
Ort.	555.07	14.09	555.07	14.25	555.07	14.10



Şekil 4.5. Deneme başına ortalama desen sayısı.

Parametre aralıklarını genişletmenin, desen sayısını azaltmada faydalı olabileceği, ancak etkisinin sınırlı olduğu açıktır. Hızlı yanıtın gerekli olmadığı senaryolarda, daha geniş bir parametre aralığı kullanılabilir. Her bir ilave deneme için hesaplama süresine ortalama 0,5 saniye eklenmektedir.

Stok başına maliyet (u_B) ve desen başına maliyet (u_S), kullanıcı tarafından belirlenir ve sürecin kendisine bağlıdır. Bu nedenle, bu maliyetlerin etkisi burada incelenmemektedir. Ancak, maliyetlerin farklı değerleri, farklı çözümler üretebilir. Aynı problemler için farklı maliyet değerlerinin etkisi, Bölüm 4.5. Umetani kıyaslama örnekleri bölümünde incelenmiştir.

5. SONUÇLAR VE GELECEKTEKİ ÇALIŞMALAR

5.1. Bulguların Özeti ve Katkılar

OPTICUT algoritması, tek boyutlu stok kesme problemini desen minimizasyonu ile birleştiren yenilikçi ve etkili bir sezgisel yaklaşım sunmaktadır. Algoritmanın performansı, hem sentetik (CUTGEN1) hem de gerçek dünya (Umetani) veri setlerinden elde edilen toplam 1.840 kıyaslama örneği üzerinden kapsamlı olarak değerlendirilmiştir.

Bu değerlendirmeler, OPTICUT'ın stok uzunluğuna bağlı olarak küçük ve büyük sipariş boyutlarını içeren çeşitli senaryolarda (CUTGEN1 grup 1 ve grup 3) ve kimyasal elyaf endüstrisinden alınan büyük stoklu örneklerde, mevcut en gelişmiş yöntemlere kıyasla daha üstün sonuçlar ürettiğini ortaya koymaktadır. Özellikle, toplam maliyetin CUTGEN1 örneklerinde %0.0013, Umetani örneklerinde ise %0.01 ile %0,51 oranında azaltılması, desen sayısının CUTGEN1'de %0.6, Umetani'de ise %0,82 oranında düşürülmesi, algoritmanın etkinliğini kanıtlamaktadır.

OPTICUT'ın başarısı, dinamik parametrelerin her nesilde evrilerek yüksek kaliteli kesim desenleri üretmesine ve bu desenlerin tamsayılı doğrusal programlama (ILP) ile nihai optimizasyona tabi tutulmasına dayanmaktadır. Bu hibrit yaklaşım, literatürdeki yöntemlerden daha iyi çözümler sağlayarak algoritmanın hem teorik hem de pratik açıdan değerini yükseltmektedir.

Ayrıca, OPTICUT'ın esnekliği, kullanıcıların stok ve kurulum maliyetlerini özelleştirmesine olanak tanıyarak farklı endüstriyel gereksinimlere uyarlanabilirliğini güçlendirmektedir. Makul hesaplama süreleri (örnek başına ortalama 165 saniye) ile gerçek dünya uygulamalarına uygunluğu da kanıtlanmıştır. Bu özellikler, OPTICUT algoritmasının farklı uygulama alanlarında etkili ve verimli bir şekilde kullanılabildiğini ortaya koymaktadır. Tablo 5.1'de OPTICUT algoritmasının diğer algoritmalara karşı avantaj özeti sunulmuştur.

Tablo 5.1. OPTICUT algoritmasının diğer algoritmalara karşı avantaj özeti.

Avantaj	Açıklama
Üstün performans ve literatürü aşan çözümler	Stok boyutuna göre hem küçük hem de büyük sipariş boyutları dahil olmak üzere çeşitli gerçek dünya ve sentetik senaryolarda mevcut metodolojilerinden daha iyi çözümler bulmaktadır.
Maliyet azaltma	Stok maliyetlerinin azaltılmasın ile toplam maliyet düşüşü sağlar.
Desen ve stok minimizasyonu	Gerekli kesim desenlerinin ve stokların sayısını en aza indirir.
Yüksek kaliteli desenler	Dinamik değişkenler ve farklı desen oluşturma algoritmaları sayesinde yüksek kaliteli kesim desenleri oluşturur.
Esneklik	Kullanıcıların ihtiyaçlarına uygun optimizasyon sonuçları elde etmek için birim stok ve kurulum maliyetlerinin yanı sıra deneme sayısı da dahil olmak üzere parametreleri değiştirmelerine olanak tanır.
Uygulama kolaylığı	Pratik kullanım ve kodlama için kolay uygulanabilen bir yaklaşım sunar.
Gerçek dünyada uygulanabilirlik	Makul zaman dilimleri içinde gerçek dünyadaki kesim problemlerini çözebilir.
Yenilikçi özellikler	Her nesil boyunca değişen ve çözüm uzayını genişleten dinamik değer, öncelik ve izin verilen fire değişkenlerini kullanır. Aynı anda sütün üretimi, desen üretimi ve doğrusal programlama yöntemlerini başarılı bir şekilde kullanır.

5.2. Gelecekteki Araştırma Yönelimleri

OPTICUT, desen minimizasyonu, fire minimizasyonu ve maliyet optimizasyonu açısından önemli avantajlar sunmasına rağmen, mevcut haliyle bazı sınırlamalar taşımaktadır. Bu eksikleri gidermek ve algoritmanın performansını daha da artırmak için aşağıdaki iyileştirme önerileri uygulanabilir:

5.2.1. Ticari Çözücü Entegrasyonu

Algoritmanın PULP-CBC-CMD çözücüsüne bağımlılığı, büyük ölçekli problemlerde hesaplama süresini ve çözüm kalitesini sınırlayabilir. ILP optimizasyon aşamasında, Gurobi veya CPLEX gibi ticari çözücülerin kullanımı hem hesaplama süresini kısaltabilir hem de çözüm kalitesini artırabilir. Özellikle büyük ölçekli problemlerde, bu çözücülerin gelişmiş dallanma ve kesme teknikleri, PULP-CBC-CMD'ye kıyasla daha verimli sonuçlar üretebilir.

5.2.2. Paralel Hesaplama Uygulaması

Algoritma A ve Algoritma B'nin sıralı çalışması, işlem süresini uzatmakta ve paralel hesaplama potansiyelini kullanamamaktadır. Algoritma A ve Algoritma B'nin bağımsız doğası, paralel hesaplama tekniklerinden yararlanmaya uygundur. Bu iki algoritmanın eşzamanlı çalıştırılması, toplam işlem süresini yaklaşık %50 oranında azaltabilir ve özellikle zaman kritik uygulamalarda OPTICUT'ın pratikliğini artırabilir.

5.2.3. Çoklu Stok Uzunluklarını İşleyebilme

OPTICUT, şu anda yalnızca tek bir stok uzunluğunu işleyebilecek olarak tasarlanmıştır. Bu durum, birden fazla stok uzunluğunun yaygın olduğu endüstriyel senaryolarda uygulanabilirliğini kısıtlamaktadır.

Algoritmanın birden fazla stok uzunluğunu aynı anda işleyebilecek olarak genişletilmesi, endüstriyel uygulanabilirliğini önemli ölçüde artıracaktır. Bu amaçla, sütun üretimi ve desen optimizasyon aşamalarına, çoklu stok uzunluklarını dikkate alan bir kısıt matrisi eklenebilir. Böyle bir geliştirme, kâğıt ve çelik gibi farklı stok boyutlarının bir arada kullanıldığı sektörlerde OPTICUT'ın kullanım alanını genişletebilir.

5.2.4. Nihai Çözüm Sonrası Desen Azaltma Mekanizması

Mevcut desen havuzuna, fire seviyesini korurken desen sayısını daha da azaltacak bir post-processing algoritması entegre edilebilir. Örneğin, benzer desenlerin birleştirilmesi, düşük frekanslı desenlerin elimine edilmesi gibi teknikler, OPTICUT'ın desen minimizasyon kapasitesini daha da güçlendirebilir.

5.2.5. Parametre Otomasyonu

OPTICUT'ın dinamik parametreleri şu anda sabit olarak ayarlanmıştır. Bu parametrelerin problem özelliklerine (örneğin, sipariş çeşitliliği veya stok uzunluğu) göre otomfire olarak uyarlanmasını sağlayacak bir makine öğrenimi modeli entegre edilmesi, algoritmanın kullanıcı dostu olmasını, daha hızlı çalışmasını, daha iyi sonuçlar üretmesini sağlayabilir.

5.2.6. Farklı Hedeflerle Entegrasyon

Algoritmaya fire vermek yerine çıkan artık (kullanılabilir fire) miktarının arttırılması, müşterilerin sipariş önceliklerinin dikkate alınması gibi farklı hedefler de entegre edilebilir.

OPTICUT, desen minimizasyonuna odaklanan 1DCSP varyantı için sağlam bir temel sağlasa da tespit edilen eksiklikler ve önerilen geliştirmeler sayesinde algoritmanın hem akademik hem de endüstriyel alanlarda daha verimli bir şekilde kullanılabilmesi mümkün olacaktır. Bu iyileştirmeler, OPTICUT'ın daha hızlı, esnek ve kapsamlı bir optimizasyon aracı olarak öne çıkmasına katkı sağlayacak; böylece ileride yapılacak araştırmalar ve uygulamalar için güçlü bir altyapı sunacaktır.

KAYNAKLAR

Aliano Filho, A., Moretti, A. C., & Pato, M. V. (2018). A comparative study of exact methods for the bi-objective integer one-dimensional cutting stock problem. *Journal of the Operational Research Society, 69*(1), 91–107. https://doi.org/10.1057/s41274-017-0214-7

Alves, C., Macedo, R., & Valério de Carvalho, J. (2009). New lower bounds based on column generation and constraint programming for the pattern minimization problem. *Computers & Operations Research, 36*(12), 2944–2954. https://doi.org/10.1016/j.cor.2009.01.008

Alves, C., & Valério de Carvalho, J. M. (2008). A branch-and-price-and-cut algorithm for the pattern minimization problem. *RAIRO - Operations Research, 42*(4), 435–453. https://doi.org/10.1051/ro:2008027

Andrade, P. R. de L., de Araujo, S. A., Cherri, A. C., & Lemos, F. K. (2021). The integrated lot sizing and cutting stock problem in an automotive spring factory. *Applied Mathematical Modelling, 91*, 1023–1036. https://doi.org/10.1016/j.apm.2020.10.033

Arbib, C., Marinelli, F., & Ventura, P. (2016). One-dimensional cutting stock with a limited number of open stacks: Bounds and solutions from a new integer linear programming model. *International Transactions in Operational Research, 23*(1), 47–63. https://doi.org/10.1111/itor.12134

Belov, G., & Scheithauer, G. (2003). The number of setups (different patterns) in one-dimensional stock cutting. *Technical Report, Institute for Numerical Mathematics, Dresden University*, MATH-NM-15-2003.

Chen, Z., An, Y., Wang, Z., Wu, H., Qin, X., Eynard, B., & Zhang, Y. (2022). Hybrid offline programming method for robotic welding systems. *Robotics and Computer-Integrated Manufacturing, 73*, 102238. https://doi.org/10.1016/j.rcim.2021.102238

Cui, Y., & Liu, Z. (2011). C-Sets-based sequential heuristic procedure for the one-dimensional cutting stock problem with pattern reduction. *Optimization Methods & Software, 26*(2), 155–167. https://doi.org/10.1080/10556780903420531

Cui, Y. (2012). A CAM system for one-dimensional stock cutting. *Advances in Engineering Software, 47*, 7–16. https://doi.org/10.1016/j.advengsoft.2011.12.004

Cui, Y., Zhong, C., & Yao, Y. (2015). Pattern-set generation algorithm for the one-dimensional cutting stock problem with setup cost. *European Journal of Operational Research, 243*(1), 315–322. https://doi.org/10.1016/j.ejor.2014.12.015

Cui, Y., Song, X., Chen, Y., & Cui, Y.-P. (2017). New model and heuristic solution approach for one-dimensional cutting stock problem with usable leftovers. *Journal of the Operational Research Society, 68*(3), 269–280. https://doi.org/10.1057/s41274-016-0098-y

Dantzig, G. B. (1947). Maximization of a linear function of variables subject to linear inequalities. In T. C. Koopmans (Ed.), *Activity analysis of production and allocation* (pp. 339–347). Wiley.

De Araujo, S., Poldi, K., & Smith, J. (2014). A genetic algorithm for the one-dimensional cutting stock problem with setups. *Pesquisa Operacional, 34*(2), 165–187. https://doi.org/10.1590/0101-7438.2014.034.02.0165

Diegel, A., Chetty, M., Van Schalkwyck, S., & Naidoo, S. (1993). Setup combining in the trim loss problem: 3-to-2 & 2-to-1. *Technical Report, Business Administration, University of Natal, Durban*. (First Draft)

Dikili, A., Narli Sarıöz, E., & Pek, N. (2007). A successive elimination method for one-dimensional stock cutting problems in ship production. *Ocean Engineering, 34*(13–14), 1841–1849. https://doi.org/10.1016/j.oceaneng.2006.11.008

Dyckhoff, H. (1990). A typology of cutting and packing problems. *European Journal of Operational Research, 44*(2), 145–159. https://doi.org/10.1016/0377-2217(90)90350-K

Foerster, H., & Wäscher, G. (2000). Pattern reduction in one-dimensional cutting stock problems. *International Journal of Production Research, 38*(7), 1657–1676. https://doi.org/10.1080/002075400188780

Gau, T., & Wäscher, G. (1995). CUTGEN1: A problem generator for the standard one-dimensional cutting stock problem. *European Journal of Operational Research, 84*(3), 572–579. https://doi.org/10.1016/0377-2217(95)00023-J

Ghafari, F., Asadi, Y., Salajegheh, A., & Valipour, E. (2024). Increasing productivity by generating optimal cutting patterns in the rolling mill of Shahid Bahonar Copper Industries Corporation. *International Journal of Production Research*, 1–28. https://doi.org/10.1080/00207543.2024.2435596

Gilmore, P. C., & Gomory, R. E. (1961). A linear programming approach to the cutting stock problem: Part I. *Operations Research, 9*(6), 849–859. https://doi.org/10.1287/opre.9.6.849

Gilmore, P. C., & Gomory, R. E. (1963). A linear programming approach to the cutting stock problem: Part II. *Operations Research, 11*(6), 863–888. https://doi.org/10.1287/opre.11.6.863

Golfeto, R., Leduino, L., & Salles-Neto, L. L. (2009). A genetic symbiotic algorithm applied to the cutting stock problem with multiple objectives. *Proceedings of the 11th International Conference on Computational Science and Its Applications (ICCSA)*, 11, 1–10.

Goulimis, C. (1990). Optimal solutions to the cutting stock problem. *European Journal of Operational Research, 44*(2), 197–208. https://doi.org/10.1016/0377-2217(90)90355-F

Goulimis, C. (1990). *The cutting stock problem revisited* (Doctoral dissertation). Imperial College of Science, Technology and Medicine, London.

Gradišar, M., Resinovič, G., & Kljajić, M. (2002). Evaluation of algorithms for one-dimensional cutting. *Computers & Operations Research, 29*(9), 1207–1220. https://doi.org/10.1016/S0305-0548(01)00025-9

Guimarães, G.G., Poldi, K.C. & Martin, M. Mathematical models for the one-dimensional cutting stock problem with setups and open stacks. *Journal of Combinational Optimization* 49, 43 (2025). https://doi.org/10.1007/s10878-025-01276-5

Haessler, R. W. (1975). Controlling cutting pattern changes in one-dimensional trim problems. *Operations Research, 23*(3), 483–493. https://doi.org/10.1287/opre.23.3.483

Henn, S., & Wäscher, G. (2013). Extensions of cutting problems: Setups. *Pesquisa Operacional, 33*(2), 133–162. https://doi.org/10.1590/S0101-74382013000200001

Kantorovich, L. V. (1939). *Mathematical methods of organizing and planning production*. Leningrad: Leningrad State University Press.

Kantorovich, L. V., & Zalgaller, V. A. (1951). *Calculation of rational cuts of materials*. Leningrad: Leningrad State University Press.

Lee, J. (2007). In situ column generation for a cutting-stock problem. *Computers & Operations Research, 34*(8), 2345–2358. https://doi.org/10.1016/j.cor.2005.09.007

Martin, M., Moretti, A. C., Gomes-Ruggiero, M. A., & Salles-Neto, L. L. (2018). Modification of Haessler's sequential heuristic procedure for the one-dimensional cutting stock problem with setup cost. *Production, 28*, e20170105. https://doi.org/10.1590/0103-6513.20170105

Martin, M., Yanasse, H. H., & Salles-Neto, L. L. (2022). Pattern-based ILP models for the one-dimensional cutting stock problem with setup cost. *Journal of Combinatorial Optimization, 44*(3), 557–582. https://doi.org/10.1007/s10878-022-00848-z

McDiarmid, C. (1999). Pattern minimization in cutting stock problems. *Discrete Applied Mathematics, 98*(1–2), 121–130. https://doi.org/10.1016/S0166-218X(99)00112-2

Mobasher, A., & Ekici, A. (2011). Cutting stock problem with setup considerations. *61st Annual IIE Conference and Expo Proceedings*.

Mobasher, A., & Ekici, A. (2013). Solution approaches for the cutting stock problem with setup cost. *Computers & Operations Research, 40*(1), 225–235. https://doi.org/10.1016/j.cor.2012.06.007

Renildo, G., Cerqueira Lima, G., & Yanasse, H. H. (2009). A pattern reduction procedure in a one-dimensional cutting stock problem by grouping items according to their demands. *Journal of Computational Interdisciplinary Sciences, 1*(2), 159–164. https://doi.org/10.6062/jcis.2009.01.02.0019

Sun, G., Xu, Z., Yu, H., Chen, X., Chang, V., & Vasilakos, A. V. (2020). Low-latency and resource-efficient service function chaining orchestration in network function virtualization. *IEEE Internet of Things Journal, 7*(7), 5760–5772. https://doi.org/10.1109/JIOT.2019.2937110

Trkman, P., & Gradišar, M. (2007). One-dimensional cutting stock optimization in consecutive time periods. *European Journal of Operational Research, 179*(2), 291–301. https://doi.org/10.1016/j.ejor.2006.03.027

Umetani, S., Yagiura, M., & Ibaraki, T. (2003). One-dimensional cutting stock problem to minimize the number of different patterns. *European Journal of Operational Research, 146*(3), 388–402. https://doi.org/10.1016/S0377-2217(02)00239-4

Umetani, S. (2018). *Benchmark instances for the one-dimensional cutting stock problem*. Osaka: Osaka University. Retrieved on November 5, 2017, from https://sites.google.com/site/shunjiumetani/benchmark

Umetani, S., Yagiura, M., & Ibaraki, T. (2006). One-dimensional cutting stock problem with a given number of setups: A hybrid approach of metaheuristics and linear programming. *Journal of Mathematical Modelling and Algorithms, 5*(1), 43–64. https://doi.org/10.1007/s10852-005-9031-0

Vanderbeck, F. (2000). Exact algorithm for minimizing the number of setups in the one-dimensional cutting stock problem. *Operations Research, 48*(6), 915–926. https://doi.org/10.1287/opre.48.6.915.12386

Vahrenkamp, R. (1996). Random search in the one-dimensional cutting stock problem. *European Journal of Operational Research, 95*(1), 191–200. https://doi.org/10.1016/0377-2217(95)00198-0

Wang, W., Shi, Z., Shi, L., & Zhao, Q. (2019). Integrated optimisation on flow-shop production with cutting stock. *International Journal of Production Research, 57*(19), 5996–6012. https://doi.org/10.1080/00207543.2018.1556823

Wäscher, G., Haußner, H., & Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research, 183*(3), 1109–1130. https://doi.org/10.1016/j.ejor.2005.12.047

Wikipedia. (2025). *Cutting stock problem*. Retrieved March 24, 2025, from https://en.wikipedia.org/wiki/Cutting_stock_problem

Yanasse, H. H., & Limeira, M. S. (2006). A hybrid heuristic to reduce the number of different patterns in cutting stock problems. *Computers & Operations Research, 33*(9), 2744–2756. https://doi.org/10.1016/j.cor.2005.02.026

ÖZGEÇMİŞ

Ad-Soyad : Nahsen KAYHAN

ÖĞRENİM DURUMU:

- **Lisans :** 2002, Marmara Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği
- **Yüksek lisans :** 2005, Marmara Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği
- **Doktora :** 2025, Sakarya Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği

MESLEKİ DENEYİM VE ÖDÜLLER:

- 2017 - : Sunparadise Alüminyum, Bilgi İşlem Müdürü
- 2014 - 2016 : Akınal Sentetik, Erp Yöneticisi
- 2008 - 2014 : Süperfilm A.Ş., Planlama Mühendisi / Üretim Yöneticisi
- 2002 - 2005 : Merlin Bilgisayar, Yazılım Geliştirme Mühendisi

TEZDEN TÜRETİLEN ESERLER:

Kayhan, N., & Tekez, E. K. (2025). OPTICUT: a new heuristic algorithm for the one-dimensional cutting stock problem with pattern minimization. *Engineering Optimization*, 1–23. https://doi.org/10.1080/0305215X.2025.2480864