ARDIŞIK MAKİNELERDE ÇOKLU OPERASYONA SAHİP İŞLERİN EŞ ZAMANLI ÇİZELGELENMESİ

Burcu ÇAĞLAR GENÇOSMAN

T.C.
ULUDAĞ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ARDIŞIK MAKİNELERDE ÇOKLU OPERASYONA SAHİP İŞLERİN EŞ ZAMANLI ÇİZELGELENMESİ

Burcu ÇAĞLAR GENÇOSMAN

Prof. Dr. H. Cenk ÖZMUTLU
(Danışman)

Yrd. Doç. Dr. Mehmet A. BEĞEN
(İkinci Danışman)
(Western Üniversitesi)

DOKTORA TEZİ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

BURSA-2014

TEZ ONAYI

Burcu ÇAĞLAR GENÇOSMAN tarafından hazırlanan "Kısıt Programlama Yönteminin Performansının Arttırılması" adlı tez çalışması aşağıdaki jüri tarafından oy birliği/oy çokluğu ile Uludağ Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalı'nda **DOKTORA TEZİ** olarak kabul edilmiştir.

Danışman	:Prof. Dr. H. Cenk ÖZMUTLU	
İkinci Danışman	:Yrd. Doç. Dr. Mehmet A. BEĞEN	
Başkan	:Prof. Dr. H. Cenk ÖZMUTLU Uludağ Üniversitesi Mühendislik Fakültesi Endüstri Mühendisliği Anabilim Dalı	İmza
Üye	:Prof. Dr. Erdal EMEL Uludağ Üniversitesi Mühendislik Fakültesi Endüstri Mühendisliği Anabilim Dalı	İmza
Üye	:Doç. Dr. M. Arslan ÖRNEK İzmir Ekonomi Üniversitesi Mühendislik ve Bilgisayar Bilimleri Fakültesi Endüstri Mühendisliği Anabilim Dalı	İmza
Üye	:Yrd. Doç. Dr. Mehmet A. BEĞEN Western Üniversitesi İşletme Fakültesi Yönetim Bilimi Anabilim Dalı	İmza
Üye	:Yrd. Doç. Dr. Mehmet AKANSEL Uludağ Üniversitesi Mühendislik Fakültesi Endüstri Mühendisliği Anabilim Dalı	İmza

Yukarıdaki sonucu onaylarım

Prof.Dr.Ali Osman DEMİR
Enstitü Müdürü
10/06/2014

U.Ü. Fen Bilimleri Enstitüsü, tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- tez içindeki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi,
- görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu,
- başkalarının eserlerinden yararlanılması durumunda ilgili eserlere bilimsel normlara uygun olarak atıfta bulunduğumu,
- atıfta bulunduğum eserlerin tümünü kaynak olarak gösterdiğimi,
- kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- ve bu tezin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez çalışması olarak sunmadığımı

beyan ederim.

10/06/2014

Burcu ÇAĞLAR GENÇOSMAN

ÖZET

Doktora Tezi

ARDIŞIK MAKİNELERDE ÇOKLU OPERASYONA SAHİP İŞLERİN EŞ ZAMANLI ÇİZELGELENMESİ

Burcu ÇAĞLAR GENÇOSMAN

Uludağ Üniversitesi
Fen Bilimleri Enstitüsü
Endüstri Mühendisliği Anabilim Dalı

Danışman: Prof.Dr. H. Cenk ÖZMUTLU

İkinci Danışman: Yrd.Doç.Dr. Mehmet A. BEĞEN (Western Üniversitesi)

Bu doktora çalışmasının amacı; makine elverişlilik ve kaynak kısıtları altında ardışık makinelerde işlenmesi gereken farklı sayıda operasyona sahip işlerin bağımsız paralel makine ortamında eş zamanlı olarak çizelgelenmesi problemi için etkili çözüm yöntemleri geliştirmektir.

Problemin çözümü için ilk olarak bir karışık tamsayılı programlama modeli (KTP1) geliştirilmiştir. Bu model teorik olarak problemi temsil etmekle birlikte, gerçek boyutlu problemlerin çözümünde yetersiz kalmıştır. Sonrasında, kısıt programlamanın özel kısıt tanımlamaları kullanılarak bir kısıt programlama modeli (KP1) geliştirilmiştir. KTP1 ve KP1 çözümleri karşılaştırılmış ve çeşitli iyileştirme çalışmaları gerçekleştirilmiştir. KP1 modelinin atama prosedürleri incelenerek blok atama yapabilen bir tamsayılı programlama modeli (TP1) geliştirilmiştir. Bu model de gerçek boyutlu problemlerle test edilmiş ve diğer yöntemlerden iyi olmasına rağmen optimal çözümlere istenen hızda ulaşamadığı görülmüştür. Bu problemi basitleştirmek amacıyla mantık-tabanlı Benders ayrıştırma tekniği (MTBA) kullanılmış ve MTBA1 ve MTBA2 olarak adlandırılan iki farklı algoritma geliştirilmiştir. Algoritmalarda kullanılan kesimler çeşitlendirilerek iyileştirme çalışmaları gerçekleştirilmiştir. Son olarak, MTBA algoritmasından esinlenilerek TP2/TP1 algoritması geliştirilmiş ve yöntemin optimum sonucu garantilediği ispatlanmıştır. Yapılan karşılaştırmalar sonucunda, TP2/TP1 algoritması ile dakikalar içinde gerçek boyutlu problemlerin optimal çözümlerine ulaşılabildiği görülmüştür.

Anahtar Kelimeler: bağımsız paralel makine çizelgeleme, makine elverişlilik ve kaynak kısıtları, karışık tamsayılı programlama, tamsayılı programlama, kısıt programlama, mantık-tabanlı Benders ayrıştırma.
2014, x + 126 sayfa.

ABSTRACT

PhD Thesis

SIMULTANEOUSLY SCHEDULING OF JOBS WITH MULTIPLE OPERATIONS IN CONSECUTIVE MACHINES

Burcu ÇAĞLAR GENÇOSMAN

Uludağ University
Graduate School of Natural and Applied Sciences
Department of Industrial Engineering

Supervisor: Prof.Dr. H. Cenk ÖZMUTLU

Second Supervisor: Asist.Prof.Dr. Mehmet A. BEĞEN (Western University)

The aim of this PhD study is to develop effective solution approaches for a scheduling problem which have jobs with different number of operations that must be processed on consecutive machines regarding to the machine eligibility and resource restrictions in an unrelated parallel machine environment.

Firstly, a mixed integer programming (MIP1) model is proposed for the solution of problem. Although MIP1 represents the problem in a theoretic way, it is insufficient to solve the real-world problems. Therefore, a constraint programming (CP1) model is developed by using special constraint definitions of constraint programming. MIP1 and CP1 results are compared with randomly generated data, and some improvement studies are performed. Considering the block scheduling behavior of the CP1 model, an integer programming (IP1) model is developed. The comparison results prove that IP1 generates better schedules than previous models, but it is not an efficient method to reach the optimal solutions of real-world problems. In order to simplify the complex scheduling problem, the logic-based Benders decomposition (LBBD) technique is used, and two different algorithms are proposed: LBBD1 and LBBD2. To improve the solutions, the cuts in the algorithms are diversified. Finally, a new algorithm as IP2/IP1 is developed by inspiring from LBBD, and it is proved that IP2/IP1 guarantees the optimal solution of problems. The comparison results show that the IP2/IP1 algorithm reaches the optimal solution of real-world problems within minutes.

Keywords: unrelated parallel machine scheduling, machine eligibility and resource restrictions, mixed integer programming, integer programming, constraint programming, logic-based Benders decomposition.
2014, x + 126 pages.

TEŞEKKÜR

Tez çalışmasının oluşturulması ve yürütülmesi aşamalarında her zaman yol gösterici yaklaşımıyla bana destek olan çok değerli hocam Prof. Dr. H. Cenk ÖZMUTLU'ya, yurt dışında saygınlığı ile bilinen Western Üniversitesi *Ivey* İşletme Bölümü'nde bana araştırma imkanı sunan ve değerli bilgileri ile çalışmalarıma katkıda bulunan hocam Yrd. Doç. Dr. Mehmet A. BEĞEN'e, İzmir Ekonomi Üniversitesi'nde *Kısıt Programlama* dersi almamı sağlayarak teorik altyapımın gelişmesini sağlayan ve önerileri ile tez çalışmalarıma katkıda bulunan Doç. Dr. M. Arslan ÖRNEK'e, tez izleme çalışmalarında beni dinleyen ve her konudaki tecrübelerini aktarak ufkumu açan değerli hocam Prof. Dr. Erdal EMEL'e teşekkürlerimi sunarım.

Destekleri ile hep yanımda olan, tez çalışmamın her döneminde ihtiyacım olduğu anda yardımıma koşan ve manevi destekleriyle bu dönemi güzel şekilde geçirmeme katkıda bulunan aileme, doktora öğrenimim süresince her zorlukta yanımda olan ve her türlü desteği sağlayan, tez çalışmalarım için yurtdışına gitmemi destekleyen ve bu konuda fedakarlık gösteren, olgunluğunu her zaman takdir ettiğim sevgili eşim Fatih GENÇOSMAN'a teşekkürü bir borç bilirim.

Doktora öğrenimim süresince maddi desteklerini esirgemeyen TÜBİTAK'a ve yeterlilik sonrası doktora araştırma bursu ile yurtdışında çalışmalarımı devam ettirmemi sağlayan YÖK'e teşekkür ederim.

Burcu Çağlar Gençosman

10/06/2014

İÇİNDEKİLER

SİMGELER ve KISALTMALAR DİZİNİ

Simgeler	Açıklama
α	Makine ortamı
β	İş karakteristiği
γ	Amaç fonksiyonu
batch(b)	Yığın işleme
block	Engelleme
brkdwn	Arızalar
C_j	*j* işinin sistemi terketmesi
C_{max}	Toplam tamamlanma zamanı
d_j	*j* işinin tamamlanması gereken zaman
E_{max}	Maksimum erken bitirme
f_j	*j* işinin maliyet fonksiyonu
F_{max}	Maksimum akış zamanı
Fm	Akış tipi atölye
FFc	Esnek akış tipi atölye
FJc	Esnek atölye tipi
fmls	İş aileleri
I_{max}	Maksimum boş makine zamanı
Jm	Atölye tipi
L_j	*j* işinin gecikmesi
L_{max}	Maksimum gecikme
m_j	*j* işinin operasyon sayısı
M_j	Makine elverişlilik sınırlamaları
n_j	İşlerin operasyon sayısında sınırlamalar
nbr	İşlerin sayısında sınırlamalar
nwt	Beklemesiz
Om	Açık tip atölye
p_{ij}	*j* işinin *i* makinesinde işlenme süresi
Pm	Özdeş paralel makineler
prec	Öncelik kısıtları
prmp	Bölünebilme
prmu	Permütasyon
Qm	Farklı hızlara sahip paralel makineler
r_j	*j* işinin işlem için uygun hale gelme zamanı
Rm	Bağımsız paralel makineler
res	Kaynaklar
rcrc	Yeniden dolaşım
s_{jk}	Sıra bağımlı hazırlık süreleri
T_j	*j* işinin pozitif gecikmesi
T_{max}	Maksimum pozitif gecikme
w_j	*j* işinin ağırlığı

Kısaltmalar	**Açıklama**
%EAOS	En iyi amacın ortalama yüzde sapması
KP	Kısıt programlama
KSP	Kısıt sağlama problemi
KTP	Karışık tamsayılı programlama
MTBA	Mantık-tabanlı Benders ayrıştırma tekniği
TP	Tamsayılı programlama

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

1. GİRİŞ

1.1. Çalışmanın Amacı

Günümüzde kullanımı hızla artan iletişim teknolojileri ile üretim/hizmet sektörleri için küreselleşme kaçınılmaz olmuş ve bu gelişim, rekabetçi üretim/hizmet ortamını da beraberinde getirmiştir. Bu süreçte işletmelerden beklenen, müşteri taleplerini istenen miktar ve kalitede en az maliyetle karşılayabilmeleridir. Bu sebeple maliyetlerini düşürme çabasına giren işletmeler, azalan maliyetlerle ürün/hizmet fiyatlarını rekabet edilebilir seviyede tutmayı hedeflemişlerdir. İyileştirilen fiyat politikalarının yanında işletmelerin değişen müşteri taleplerine de çok hızlı şekilde cevap verebilmeleri gerekmektedir. Stoklarını ve üretim hazırlık maliyetlerini artırmadan bu değişken taleplere cevap verebilmek ve hatta yüksek verimlilikte bir üretim ile bu değişkenliği yönetmek, işletmelerin karşılaştığı en büyük zorluklardan birisi olmuştur. Sektörde yerini korumak isteyen işletmeler Ar-Ge faaliyetlerine daha fazla önem vererek modern üretim teknikleri ile üretim süreçlerini iyileştirme çalışmalarına hız vermiştir.

Bu çalışma alanlarından biri olan çizelgeleme değişken taleplere, değişkenlik gösteren üretim ortamlarında etkin şekilde cevap verebilmek adına önemli bir rol oynamaktadır. Bunun farkına varan işletmeler çizelgeleme konusunda ayrı bir hassasiyet göstermişlerdir. Üretim ortamının ve taleplerin değişken olması, süreci stokastik hale getirmiş ve bu tür stokastik değişimlere tatmin edici sürede uygun çizelgeler üretebilecek yapılara ihtiyaç duyulmuştur. Bu tarz değişiklikler iki şekilde ele alınabilir: Bunlardan ilki tüm stokastik süreçlerin algoritma dahilinde değerlendirilmesidir. İkinci olarak ve çalışmada tercih edilen yöntem; planlama anında her şeyin deterministik olduğunun varsayılması ve herhangi bir değişiklikte algoritmanın tekrar çalıştırılmasıdır. Bu sayede değişkenliklere kısa sürede cevap verme mümkün hale gelmiştir. Firmaların değişken ortamlardaki çizelgeleme ihtiyaçlarını giderebilmek amacıyla gerçekleştirilen çalışmalar sürekli yeni algoritmalarla desteklenmektedir.

Çizelgeleme 1900'lerin başında Henry Gannt (1919) tarafından gerçekleştirilen çalışmalarla ilk kez ciddi bir şekilde ele alınmış, ancak ilk yayınların literatürde yer alması uzun sürmüştür. İlk yayınlardan bazıları 1950'lerde Naval araştırma merkezindeki çalışmalarla ortaya çıkmış (Jackson 1956, Wagner 1959) ve 1960'larda çizelgeleme problemlerinin dinamik programlama ve tamsayılı programlama ile modellenmesi çalışmaları hızla artış göstermiştir (Held ve Karp 1962). Richard Karp'ın (1972) karmaşıklık teorisi üzerine olan yayınından sonra 1970'lerdeki araştırmaların odak noktası, çizelgeleme problemlerinin karmaşıklık hiyerarşisi olmuştur. 1980'lerde bir çok alanda hızla artan şekilde bilim alanında ve endüstride uygulamalı çalışmalar gerçekleştirilmiştir. 1990'larda literatür taramalarıyla zenginleştirilen çalışmalar, sezgisel yöntemlerin çizelgeleme uygulamaları ile arttırılmıştır. 2000'li yıllardan itibaren çizelgeleme çalışmaları büyük bir hız kazanmıştır. Bu süreçte kişisel bilgisayarların kullanımının yaygınlaşması da süreçlerin modellenmesi aşamasında çizelgeleme çalışmalarına önemli katkıda bulunmuştur (Pinedo 2012). Sonuç olarak, çizelgeleme problemleri altmış yıldan fazla süredir çalışılmaktadır ve yeni geliştirilen algoritmalarla desteklenen bu çalışmalar günümüzde de devam etmektedir.

1.2. Kullanılan Metodoloji

Bu çalışmada, NP-zor sınıfı bir bağımsız paralel makine çizelgeleme problemi ele alınmıştır. Bu problemde, makine elverişlilik ve kaynak kısıtları altında birbirlerinden farklı operasyonları barındıran işler vardır. Bu işler bağımsız paralel makinelerde eş zamanlı olarak işlenebilir. Ayrıca işlerin operasyonları ardışık makinelerde işlenmelidir. Bu sebeple bağımsız paralel makine ortamı, işlerin başlangıç makinelere atanmasından sonra ardışıklık/zincir özelliğinden dolayı akış tipi çizelgeleme problemine dönüşmektedir. Dolayısıyla, 3-alanlı $\alpha|\beta|\gamma$ sınıflandırması kullanılarak problem $R_m|M_i; res; chains|C_{max}$ şeklinde tanımlanabilir.

Problemin çözümü için ilk olarak karışık tamsayılı programlama modeli (KTP1) geliştirilmiştir. Bu model teorik olarak problemi temsil etmekle birlikte gerçek boyutlu problemlerin çözümünde yetersiz kalmıştır. Sonrasında, kısıt programlamanın özel

kısıt tanımlamaları kullanılarak kısıt programlama modeli (KP1) geliştirilmiştir. KTP1 ve KP1 çözümleri karşılaştırılarak çeşitli iyileştirme çalışmaları gerçekleştirilmiştir. KP1 modelinin atama prosedürleri incelenerek blok atama yapabilen bir tamsayılı programlama modeli (TP1) geliştirilmiştir. Bu model de gerçek boyutlu problemlerle test edilmiş ve diğer yöntemlerden iyi olmasına rağmen optimal çözümlere istenen sürede ulaşamadığı görülmüştür.

Problemin NP-zor yapısı çözümünün de güç olduğunu göstermektedir. Bu problemi basitleştirmek adına karmaşık olan problem mantık-tabanlı Benders ayrıştırma tekniği (MTBA) kullanılarak atama ve sıralamadan oluşan iki ayrı probleme (*ana-problem* ve *alt-problem*) dönüştürülmüştür. Ana-problem atama problemi olarak tanımlanmış ve işlerin makinelere atanmasını gerçekleştirmiştir. Alt-problem ise işlerin atandıkları makineler üzerinde çizelgelenmesini kapsamıştır. Ana-problem çözümü için, işlerin sadece makine elverişlilik kısıtlarına göre makinelere atanmasını sağlayan bir tamsayılı programlama modeli (TP2) geliştirilmiştir. Ayrıca, TP2 modeline Benders kesim denklemleri eklenmiş ve KP1 modeline de yapılan kısmi atamaların sabitlendiği bir denklem eklenerek, MTBA1 algoritmasının temelleri oluşturulmuştur. MTBA1 algoritması ilk iterasyonda işlerin makinelere atanması problemini kesim denklemleri olmadan çözmüş ve ilk kısmi atama bulunmuştur. Sonrasında, KP1 modeline bu atamalar sabitlenerek gönderilmiş ve işlerin ilgili makineler üzerindeki sırası, bir diğer ifade ile başlangıç zamanları KP1 tarafından belirlenmiştir. Durdurma kriteri sağlanmışsa, elde edilen en iyi çözüm sonuç olarak belirlenmiş ve algoritma sonlandırılmıştır. Eğer durdurma kriteri sağlanmamışsa, mevcut atamanın çözüm uzayından çıkarılması için Benders kesim denklemi kullanılmış ve bu atama kesim denkleminde kullanılmak üzere TP2 modeline gönderilmiştir. Böylece ikinci iterasyonda TP2 modelinin, ilk iterasyondan farklı atamalara ulaşması sağlanmıştır. Bu şekilde algoritma, en iyi çözüme ulaşana kadar veya ana-problemin bütün çözüm uzayı taranana kadar devam etmiştir. Bölüm 5'de detaylandırılan MTBA1 algoritmasında, alt-problem çözümünde kullanılan KP1'in optimali ispat edememesi bir dezavantaja dönüşmüş ve bu algoritmayı hızlandırabilmek için farklı kesim denklemleri denenmiştir. Bu denemelerden sonra tamsayılı programlamanın optimalite bilgisinden yararlanmanın daha başarılı sonuçlar

üreteceği düşünülmüş ve MTBA2 algoritması geliştirilmiştir.

MTBA2 algoritmasında ana-problem ve alt-problem çözümleri için tamsayılı matematiksel modeller kullanılmıştır. Dolayısıyla, çözümlerin optimalliği veya optimalden uzaklığı ile ilgili bilgilere erişilebilmiştir. MTBA2'de TP2 ve TP1 modelleri mantık-tabanlı Benders ayrıştırma için uygun hale getirilmiş ve MTBA1 ile benzer durdurma kriterleri kullanılmıştır. Bölüm 5'de ayrıntılandırılan algoritmanın, gerçek boyutlu problemlerde optimal çözüme ulaşmada zaman sınırlamasına takıldığı görülmüştür. Son olarak, TP2 ve TP1'in amaç fonksiyonları iyileştirilmiş ve aynı amaç fonksiyonunun hesaplanması sağlanmıştır. TP2 ve TP1 modelleri Benders kesimleri olmadan kullanılmış ve sadece atama problemi sonucunun sıralama problemine gönderildiği denklem TP1'e eklenmiş, böylece TP2/TP1 algoritması geliştirilmiştir. Geliştirilen algoritma ile deneyler gerçekleştirildiğinde bütün problemlerde iki model ile de aynı amaç fonksiyonu değerlerinin bulunduğu ve bu değerlerin KTP2 ile bulunan optimal değerlere eşit olduğu görülmüştür. Sonrasında yapılan ispatla TP2/TP1 algoritmasının NP-zor olan bir problemin optimum sonucunu garantilediği gösterilmiştir. TP2/TP1 algoritması ile dakikalar içinde gerçek boyutlu problemlerin optimal çözümlerine ulaşılabilmiştir. Algoritmanın hızından yararlanarak yeniden çizelgeleme yöntemi geliştirilmiş ve makine arızaları durumunda kalan üretim için yeniden çizelgeleme yapılarak uygun olmayan makinelerdeki atamalar güncellenmiştir. Ek olarak, geliştirilen yöntemin mevcut probleme katkılarının incelenmesi için kapasite artış oranları araştırılmıştır. Son olarak, duruşların istatistiksel analizi yapılmış ve farklı amaç fonksiyonları modellerde kullanılarak elde edilen çizelgeler karşılaştırılmıştır. Geliştirilen bütün yöntemler Çizelge 1.1'de özetlenmiştir.

1.3. Literatüre Yapılan Katkılar

Tez çalışması kapsamında, makine elverişlilik ve kaynak kısıtları altında ardışık makinelerde işlenmesi gereken farklı sayıda operasyona sahip işlerin bağımsız paralel makine ortamında eş zamanlı olarak çizelgelenmesi; $R_m|M_i; res; chains|C_{max}$ problemi ele alınmıştır. Bağımsız paralel makine ortamı şeklinde başlayan çizelgeleme

Çizelge 1.1. Problem çözümü için geliştirilen yöntemler

Geliştirilen Yöntemler	Özellikleri
KTP1	- Karışık tamsayılı programlama kullanılmıştır. - Problem varsayımlarını test etmek amacıyla kullanılmıştır. - Gerçek boyutlu problemlerin optimal çözümlerinde yeterli olamamıştır.
KP1	- Kısıt programlama kullanılmıştır. - Kısa sürede başarılı çizelgelere ulaşılmıştır. - Gerçek boyutlu problemlerin optimal çözümlerinde yeterli olamamıştır.
TP1	- Tamsayılı programlama kullanılmıştır. - KTP1 ve KP1 modellerinden daha başarılı olmuştur. - Gerçek boyutlu problemlerin optimal çözümlerinde yeterli olamamıştır.
TP2	- Tamsayılı programlama kullanılmıştır. - İşlerin makinelere atanması problemini çözmek için geliştirilmiştir. - Ayrıştırma modellerinde kullanılmıştır.
MTBA1	- Mantık-tabanlı Benders ayrıştırma yöntemi kullanılmıştır. - Ana-problem çözümü için TP2 ve alt-problem çözümü için KP1 kullanılmıştır. - Gerçek boyutlu problemlerin optimal çözümlerinde yeterli olamamıştır.
MTBA2	- Mantık-tabanlı Benders ayrıştırma yöntemi kullanılmıştır. - Ana-problem çözümü için TP2 ve alt-problem çözümü için TP1 kullanılmıştır. - Gerçek boyutlu problemlerin optimal çözümlerinde yeterli olamamıştır.
TP2/TP1	- Ayrıştırma yöntemlerinden esinlenilerek geliştirilmiştir. - TP2 ve TP1 modelleri problem çözümünde iteratif olarak kullanılmıştır. - Gerçek boyutlu problemlerin optimal çözümlerinde başarılı olmuştur.

probleminde, işlerin makinelere atanması sonrasında işlerin operasyonlarının ardışık makinelerde işlenmesi zorunluluğu, atama sonrasında makine ortamını akış tipine çevirmektedir. Parça üretimi için gerekli makinelerin tümü sipariş miktarı tamamlanana kadar ilgili üretime ayrılmaktadır, dolayısıyla işlerin bölünmesi söz konusu değildir. Bilindiği kadarıyla bu problem literatürde açık bir problemdir ve henüz incelenmemiştir. İlk defa bu problemin tanımı yapılarak literatürdeki bağımsız paralel makine çizelgeleme problemlerine bir yenisi eklenmiştir.

Problem tanımı sonrasında problemin çözümü için ilk karışık tamsayılı model, ilk kısıt programlama modeli ve ilk mantık-tabanlı Benders ayrıştırma modelleri geliştirilmiştir. Ayrıca, ayrıştırma tekniklerinden esinlenerek yeni bir algoritma oluşturulmuştur. Çalışmada geliştirilen bütün yöntemler özgündür ve literatüre katkıdır.

Son olarak, uygulamada bir gerçek hayat problemi ele alınmıştır. NP-zor olarak sınıflandırılan bu bağımsız paralel makine çizelgeleme problemi çözümü için farklı yaklaşımlar denenmiş, son olarak TP2/TP1 algoritması geliştirilmiştir. Optimum sonucu garantilediği ispatlanan TP2/TP1 algoritması, gerçek boyutlu verilerle test edilmiş ve

problemlerin hızlı bir şekilde optimal çözümleri elde edilebilmiştir. Bu başarının bir devamı olarak, bağımsız paralel makine çizelgeleme problemlerine benzer farklı çizelgeleme problemleri için TP2/TP1 algoritması güncellenebilir ve farklı sınıfta problemlere uygulanarak bu problemlerin de optimal çözümlerine ulaşılabilir. Sonuç olarak, tez çalışması kapsamında geliştirilen ve sezgisel yerine optimali sağlayan bir algoritma, literatürdeki benzer yapıdaki NP-zor problemlere uygulanarak, henüz çözümü sağlanamamış problemler için bir çözüm yöntemi olarak kullanılabilir.

1.4. Tezin Organizasyonu

Tez çalışması altı bölümden oluşmaktadır. Bölüm 1'de çizelgeleme problemlerinin gerçek hayattaki önemine değinilmiş ve çalışma kapsamında ele alınan problem tanımı verilerek, geliştirilen çözüm yöntemleri ve yapılan ilave çalışmalar özetlenmiştir. Bölüm 2'de üretim çizelgeleme kavramı ele alınmış, çizelgeleme problemlerinin literatürde tanımlanan sınıflandırma sistematiği, karmaşıklık sınıfları ve çalışmada ele alınan çizelgeleme problemi hakkında genel bilgiler verilmiştir. Bölüm 3'de bağımsız paralel çizelgeleme problemleri ile ilgili literatür çalışmalarına yer verilmiş, sonrasında geliştirilen çözüm yöntemleri dikkate alınarak literatürdeki çalışmalar tartışılmıştır. Bölüm 4'de ele alınan problem çözümünde kullanılacak yöntemler hakkında genel bilgi verilmiştir. Bölüm 5'de çizelgeleme problemi gerçek hayattaki şekli ile ortaya konmuş ve uygulamada karşılaşılan zorluklardan bahsedilmiştir. Sonrasında, problem çözümü için geliştirilen karışık tamsayılı doğrusal programlama, tamsayılı doğrusal programlama ve kısıt programlama modelleri ile mantık-tabanlı Benders ayrıştırma tekniğiyle geliştirilen algoritmalar detaylandırılmıştır. Son olarak, iki matematiksel modelin ardışık olarak çalışmasıyla geliştirilen TP2/TP1 algoritması detaylandırılmış ve yöntemin optimum çözümü garantilediği ispatlanmıştır. Ayrıca, geçmiş sipariş ve üretim verileri analiz edilmiş ve bu veriler yöntemlerde kullanılabilir hale getirilmiştir. Geliştirilen yöntemlerle elde edilen çizelgeler ve gerçek hayatta uygulanmış olan çizelgeler karşılaştırılmıştır. Sonrasında, pratikte karşılaşılan aksaklıkları geliştirilen yöntemlere yansıtabilmek için duruşlar dikkate alınmış ve mevcut işleyen çizelgeler, duruş sonrasında kalan üretim adetleri için yeniden üretilmiştir. Yeniden çizelgeleme metodolojisi ile yöntemlerin

aksaklıklar karşısındaki davranışları ve üretilen çizelgeler incelenmiştir. Bölüm 6'da geliştirilen yöntemlerle elde edilen çizelgelerin üretime katkıları incelenmiş ve kapasite artışına etkileri detaylandırılmıştır. Uygulamada meydana gelen arızalar ve duruşlar istatistiksel olarak incelenmiş ve bu duruşların haftalık çizelgelere yansıtılması çalışmaları gerçekleştirilmiştir. Ayrıca, farklı amaç fonksiyonları dikkkate alınmış ve bu amaç fonksiyonları ile elde edilen çizelgeler karşılaştırılmıştır. Son bölümde, değerlendirme ve sonuca yer verilmiş, tez kapsamındaki çalışmalar özetlenerek literatüre yapılan katkı ortaya koyulmuştur.

2. ÇİZELGELEME PROBLEMLERİNE GENEL BAKIŞ

Çizelgeleme, performans ölçütlerinin en iyilenmesi amacıyla kıt *kaynakların aktivitelere* atanması olarak tanımlanabilir (Leung 2004). *Kaynaklar* ve *aktiviteler* çok farklı formlarda olabilir. Örnek olarak, bir montaj hattındaki makineler, bilgisayar işlemcileri ve havaalanındaki uçak pistleri *kaynaklar* olarak sınıflandırılabilirken; üretim sürecindeki operasyonlar, bilgisayar programları veya havaalanındaki iniş ve kalkışlar *aktiviteler* olabilir. Benzer şekilde, en iyilenmek istenen performans ölçütleri; toplam tamamlanma zamanının minimizasyonu veya geç kalan işlerin minimizasyonu gibi çeşitlilik gösterebilir (Leung 2004). Bu çeşitliliğin yanında altmış yıldan fazla bir geçmişe sahip olmak, çizelgeleme alanının inanılmaz derecede geniş olmasını ve binlerce çalışmayı barındırmasını sağlamıştır.

Modern çizelgeleme teorisinin temellerini oluşturan modellerin ve optimizasyon tekniklerinin gelişimi II. Dünya Savaşı yıllarına uzanmaktadır. (Baker ve Trietsch 2009). 1950'lerde bu çalışmaların bir kısmı resmi olarak yayınlanmış (Flood 1954, Vazsonyi 1955) ve 1960'larda ilk ders kitapları basılmıştır (Conway ve Maxwell 1967). Bu tarihlerde ele alınan problemler nispeten basit olduklarından, etkin algoritmalarla optimal çözümler elde edilebilmiştir. Ancak 1970'lerde karmaşıklık teorisinin gelişmesiyle birçok çizelgeleme probleminin NP-zor olduğu gösterilmiştir. 1980'lerde problem zorluk dereceleri ve çeşitliliği giderek artmakla birlikte akademi ve endüstri farklı yönlere doğru ilerlemişlerdir. Bir taraf yakınsama algoritmalarının gelişimi üzerinde dururken, diğer taraf stokastik çizelgeleme problemlerine ilgi göstermeye başlamıştır. Bu tarihten itibaren çizelgeleme teorisi çok hızlı bir şekilde gelişme göstermiştir. Geçen altmış yıldan sonra çizelgeleme alanında inanılmaz büyük bir bilgi havuzu oluşmuştur ve her geçen gün yapılan yeni çalışmalarla bu havuz genişlemeye devam etmektedir (Leung 2004).

Bu bilgi yığınında ortak tanımları koruyabilmek amacıyla çizelgeleme notasyonu yıllar içinde geliştirilmiştir. Geliştirilen ortak terminoloji ile, farklı çalışmaların birbirleri ile karşılaştırılabilmesi ve benzer çalışmaların sınıflandırılabilmesi mümkün kılınmış ve çalışmalar arası süreklilik sağlanabilmiştir.

2.1. Çizelgeleme Problemleri Sınıflandırması

Çizelgeleme alanındaki ilk çalışmaların bir çoğu üretim problemlerini hedef almıştır (Baker ve Trietsch 2009). Bu sebeple, üretim sistemleri dışında da kullanılmasına rağmen, genelde çizelgeleme problemleri tanımında üretim terminolojisi kullanılmaktadır. Kaynaklar *makine*, aktiviteler *iş* olarak tanımlanmaktadır. İşlerin birden fazla görevden oluşması durumunda *operasyonlar* devreye girmekte ve çizelgeleme problemi ortamına *atölye* denilmektedir (Baker ve Trietsch 2009).

Bütün çizelgeleme problemlerinde ele alınan makine ve işlerin sonlu olduğu kabul edilir (Pinedo 2012). İşlerin sayısı *n* ile ve makinelerin sayısı *m* ile gösterilir. İşler *j* ile ve makineler *i* indisi ile temsil edilir. Eğer bir iş birden fazla işlem gerektiriyorsa veya birden fazla operasyondan oluşuyorsa (i,j) çifti *j* işinin *i* makinesideki işlemini veya operasyonunu ifade eder (Pinedo 2012). Sınıflandırmada, her makinenin aynı anda bir iş işleyebileceği ve bir işin aynı anda en fazla bir makinede işlenebileceği kabul edilmiştir (Graham ve ark. 1979). Graham ve ark. (1979) çizelgeleme problemleri sınıflandırmasında 3-alanlı $\alpha|\beta|\gamma$ notasyonunu tanımlamışlardır; α alanı makine ortamını temsil eder, β alanı işlemlerin özellikleri ile ve kısıtlarla ilgili bilgi içerir ve γ alanı optimize edilecek amaç fonksiyonunu tanımlar. Bu notasyonun anlatımında Graham ve ark. (1979), Blazewicz ve ark. (1983) ve Pinedo'nun (2012) çalışmalarından yararlanılmıştır. İngilizce çizelgeleme terimlerinin Türkçe karşılıkları için Çetinkaya'nın (2009) çalışması dikkate alınmıştır.

2.1.1. İş verisi

Aşağıdaki veriler *j* işi ile ilgilidir:

Operasyon Sayısı (m_j): *j* işinin operasyon sayısıdır.

İşlem Süresi (p_{ij}): *j* işinin *i* makinesinde işlenme süresidir. Eğer işlem süresi makineye bağlı değilse *i* indisi kullanılmadan p_j ile de gösterilebilir.

Serbest Kalış Zamanı (r_j): *j* işinin işlem için uygun hale gelme zamanıdır. Hazır olma zamanı olarak da tanımlanabilir.

Teslim Zamanı (d_j): *j* işinin tamamlanması için gereken zamandır.

Ağırlık (w_j): *j* işinin ağırlığı bir öncelik faktörüdür ve diğer işler arasında *j* işinin önemini temsil eder.
Maliyet Fonksiyonu (f_j): *j* işi *t* zamanında tamamlandığında ölçülen $f_j(t)$ maliyetidir. Genelde bu veriler tamsayıdır (Graham ve ark. 1979).

2.1.2. Makine ortamı

Makine ortamını gösteren α=$\alpha_1\alpha_2$ alanı iki parametreden oluşur. $\alpha_1\epsilon\{\emptyset, P, Q, R, PD, F, FFc, J, FJ, O\}$ parametresi makine tipini gösterir ve $\alpha_2\epsilon\{\emptyset, m\}$ parametresi problemdeki makine adedini temsil eder. Bu iki parametre birlikte kullanılarak olası makine ortamları tanımlanmıştır.

- **Tek makine (α=1):** Bu durumda α_1=$\emptyset$ ve α_2=1'dir. Tek makinenin olduğu çalışma ortamı diğer makine ortamları içinde en basit olanıdır.
- **Özdeş paralel makineler (α=Pm):** α_1=P ve α_2=m'dir. Paralel *m* makinenin olduğu çalışma ortamında *j* işi tek operasyon gerektirir ve herhangi bir *m* makine veya bir makine alt kümesinde işlenebilir. Eğer *j* işi belli bir M_j alt kümesinde işlenebiliyorsa, bu durum β alanında gösterilir.
- **Farklı hızlara sahip paralel makineler(α=Qm):** Paralel *m* makine farklı hızlara sahiptir. *i* makinesinin hızı v_i ile tanımlanır ve *j* işinin *i* makinesinde harcadığı p_{ij} zamanı p_j/v_i'ye eşittir (bütün işlemin *i* makinesinde yapıldığı varsayılır). Bu ortama birbiçimli paralel makineler (uniform parallel machines) de denir. Eğer bütün makineler aynı hıza sahipse, örnek olarak $v_i = 1$ ise, bütün p_{ij} değerleri p_j'ye eşittir ve bir önceki durumdaki gibi özdeş paralel makine (identical parallel machines) ortamı oluşur.
- **Bağımsız paralel makineler (α=Rm):** Önceki durumun genellemesidir. Paralel *m* farklı makine vardır. *i* makinesi *j* işini v_{ij} hızında işlemektedir. *j* işinin *i* makinesi üzerinde harcadığı zaman p_j/v_i'ye eşittir (bütün işlemin *i* makinesinde yapıldığı varsayılır). Makinelerin hızları işlerden bağımsız ise, yani her *i* ve *j* için v_{ij}=v_i ise bir önceki durumdaki gibi özdeş paralel makine ortamı oluşur.
- **Akış tipi atölye (α=Fm):** Seri halde *m* makine vardır. Her iş her bir *m* makinede işlem görür. Bütün işler aynı rotayı takip etmelidir. Örnek olarak, işler ilk önce

1.makinede sonra 2.makinede sonra 3.makinede..vb. şeklinde makineleri sırayla ziyaret ederler. Bir iş bir makinede tamamlandığında diğer makinenin sırasına girer. Genelde bütün kuyruklar *İlk Giren İlk Çıkar (İGİÇ)* kuralıyla işlem yaparlar. İGİÇ disiplinin etkin olduğu akış tipi atölyesi *permütasyon akış tipi atölye* olarak tanımlanır ve β alanında *prmu* girdisi eklenir.

- **Esnek akış tipi atölye (α=FFc):** Akış tipi atölye ve paralel makine ortamlarının genelleştirilmiş halidir. Seri halde *m* makine yerine *c* adet aşama vardır ve her aşamada belli sayıda özdeş paralel makine vardır. Her iş her aşamada işlenmelidir. Örnek olarak bir iş önce 1.aşamada sonra 2.aşamada..vb. işlenmelidir. Her aşamada *j* işi herhangi bir makinede işlenir. Aşamalar arası kuyruk kuralı *İlk Gelen İlk Hizmet Alır (İGİHA)* olabilir.
- **Atölye tipi (α=Jm):** *m* makineye sahip atölye tipinde her işin önceden belirlenmiş takip etmesi gereken rotası vardır. Atölye tipi makine yerleşiminin alternatifleri; her işin bir makineyi sadece bir kere ziyaret etmesi ve her makineyi birden fazla ziyaret etmesidir. Makineler arası yeniden dolaşımı (recirculation) temsil etmek için β alanına *rcrc* girdisi eklenir.
- **Esnek atölye tipi (α=FJc):** Atölye tipi ve paralel makine ortamlarının genelleştirilmiş halidir. *m* makine yerine *c* adet iş merkezi ve bu iş merkezlerinde belli sayıda özdeş paralel makineler vardır. Her işin atölyede takip ettiği rotası vardır. *j* işi her iş merkezinde sadece bir (herhangi) makinede işlenebilir. Eğer bir iş rota gereği bir iş merkezini birden fazla ziyaret ederse β alanına *rcrc* girdisi eklenir.
- **Açık tip atölye (α=Om):** *m* adet makine vardır ve her iş *m* adet makinenin herbirinde işlenmelidir. Ancak işlem zamanlarından bazıları sıfır olabilir. Makine ortamında işlerin rotaları ile ilgili sınırlama yoktur. Her iş için bir rota belirlenebilir ve farklı işlerin farklı rotaları olabilir.

2.1.3. İş karakteristiği

İş karakteristiklerinin gösterildiği $\beta \subset \{\beta_1, \beta_2, ...vb.\}$ alanı birden fazla girdiye sahip olabilir. $\beta \epsilon \{prmp, res, res1, prec, chains, tree, intree, outtree, s_{ij}, fmls, batch(b), brkdwn, M_j, prmu, block, nwt, rcrc, nbr, n_j, p_j, r_j, d_j, \emptyset\}$ girdileri bu alanda yer alabilir.

- **Bölünebilme (prmp):** İşin bölünmesine izin verilir; herhangi bir operasyon tamamlanmadan kesintiye uğrayabilir ve kalan kısım sonra tamamlanabilir. Dolayısıyla bölünme sonrası işlem süresi hesabı kalan kısma bağlıdır. Bölünmeye izin verildiği durumda β alanına *prmp* ifadesi eklenir.
- **Kaynaklar (res):** *res* ile *s* adet sınırlı kaynağın $R_h(h = 1, ..., s)$ olduğu durum temsil edilir. Her *j* işi R_h kaynağından r_{hj} birime ihtiyaç duyar. Hiçbir zaman kaynağın %100'den fazlası kullanılamaz. Eğer sadece bir kaynak varsa β alanına *res1* eklenir.
- **Öncelik kısıtları (prec):** Öncelik kısıtları tek makine veya paralel makine ortamlarında görülebilir ve bir işin başlaması için bir veya daha fazla işin bitmesini gerektirir. *prec* durumu yönlü çevrimsiz {1,...,n} düğüm kümesini barındıran *G* çizgesinden türetilir. Eğer *G*, *j*'den *k*'ya yönlü bir patikaya sahipse, *k* başlamadan önce *j* tamamlanmalıdır denir. Her işin en fazla bir öncül faaliyeti varsa ve en fazla bir ardıl faaliyeti varsa bu durum, *chains* ile temsil edilir. Eğer her işin en fazla bir öncül faaliyeti varsa *intree*, eğer her işin en fazla bir ardıl faaliyeti varsa *outtree* ile temsil edilir. Eğer β alanında *prec* ifadesi yoksa herhangi bir öncelik kısıtı da yoktur.
- **Sıra bağımlı hazırlık süreleri (s_{jk}):** *j* işinin işlenmesinden sonra *k* işinin işlenmesi için gereken sıra bağımlı hazırlık süresi s_{jk} ile temsil edilir. Eğer *k* işi ilk iş ise hazırlık süresi s_{0k} ile ve eğer *j* işi son iş ise temizlik süresi s_{j0} ile temsil edilir (s_{0k} ve s_{j0} sıfır olabilir). Eğer *j* ve *k* işleri arasındaki hazırlık süresi makineye bağlıysa *i* indisi eklenir ve s_{ijk} ile temsil edilir. Eğer β alanında s_{jk} ifadesi yoksa bütün hazırlık sürelerinin sıfır olduğu veya sıra bağımsız olduğu ve işlem sürelerine eklendiği varsayılır.
- **İş aileleri (fmls):** Bu durumda *n* adet iş *F* adet farklı iş ailesine aittir. Aynı ailedeki işlerin işlem süreleri farklı olabilir, ancak bunlar bir makinede hazırlık süresi olmadan arka arkaya işlenebilir. Eğer makine aileler arasında değişiklik yaparsa, örneğin *g* ailesinden *h* ailesine geçerse hazırlık gerekli olur. Eğer hazırlık süresi aileler arasındaki sıraya bağımlıysa s_{gh} ile, eğer sadece başlayan aileye bağımlıysa s_h ile gösterilir. Eğer herhangi bir aileye bağımlı değilse *s* ile temsil edilir.

- **Yığın işleme (batch(b)):** Bu durumda bir makine *b* adet işi eş zamanlı olarak işleyebilir (*b* adet üründen oluşan yığın). Yığındaki işlerin işlem zamanları aynı olmayabilir, ancak işler eş zamanlı işlendiklerinden *b* adet işin tamamlanma zamanı, en uzun süreli işin tamamlanma zamanıdır. Eğer $b = 1$ ise problem geleneksel çizelgeleme ortamına dönüşür. Bir diğer özel durum olarak, $b = \infty$ ise herhangi bir zamanda makinenin işleyebileceği işlerin sayısında herhangi bir sınır yoktur.
- **Arızalar (brkdwn):** Makine arızaları, makinelerin her zaman işlemeye uygun olmadıklarını gösterir. Eğer ortamda özdeş paralel makineler varsa herhangi bir zamanda uygun makine sayısı *m(t)* gibi zamanın bir fonksiyonudur. Makine arızaları aynı zamanda makine uygunluk kısıtları ile de temsil edilir.
- **Makine elverişlilik sınırlamaları (M_j):** Makine ortamında *m* adet paralel makine (*Pm*) olduğunda β alanında M_j ifadesi görülebilir. Bu ifade varsa, her *m* makinesi *j* işini işleyememektedir. M_j kümesi *j* işini işleyebilen makineleri barındırır. Eğer β alanında M_j ifadesi yoksa, her *j* işi her *m* makinesinde işlenebilir.
- **Permütasyon (prmu):** Akış tipi atölyede görülen bu durumda, her makine önündeki kuyruk *İlk Giren İlk Çıkar* (İGİÇ) disiplinine göre ilerler, böylece işlerin makinelerdeki sırası sistem tarafından sağlanır.
- **Engelleme (block):** Akış tipi atölyelerde görülen bir durumdur. Eğer akış tipi atölyede ardışık makineler arası tampon (buffer) sınırlandırılmışsa ve buradaki iş sayısı üst limite gelmişse, önceki makinede tamamlanmış işin serbest kalması engellenir. Bu engelleme ile önceki makinede tamamlanmış iş kalır ve makinenin yeni bir iş işlemesi engellenir. Eğer iki ardışık makine arasında sıfır tampon varsa sonraki makinedeki iş tamamlanmadan, önceki makinedeki iş serbest bırakılmaz; bu şekilde önceki makinenin yeni işi işlemesi de engellenmiş olur. Engelleme durumunda makinelerin İGİÇ prensibine göre işlem yaptıkları varsayılır ve *block* ifadesi aslında *prmu* ifadesini de barındırır.
- **Beklemesiz (nwt):** Akış tipi atölyelerde görülen bir durumdur. İki ardışık makine arasında işlerin beklemesine izin verilmez. Bu durumda, eğer akış tipinde makinelerde beklemeler gerçekleşecekse, ilk makinede işin işlenmesi geciktirilir, çünkü makineler arasında beklemeden ilerlemesi gereklidir. Bu durumda makineler İGİÇ prensibiyle çalışır.

- **Yeniden dolaşım (rcrc):** Bir işin bir makineyi veya iş merkezini birden fazla ziyaret etmesi durumunda iş atölyesinde veya esnek iş atölyesinde görülen bir durumdur.
- **İşlerin sayısında sınırlamalar (nbr):** Eğer β alanında *nbr* ifadesi varsa işlenecek işlerin sayısı sınırlanmıştır. Örnek olarak $nbr = 5$ ise en fazla 5 adet iş işlenebilir. Eğer β alanında *nbr* ifadesi yoksa işlerin sayısı sınırlanmamıştır ve girdi parametresi *n* ile sisteme verilir.
- **İşlerin operasyon sayısında sınırlamalar (n_j):** Bu durum iş atölyelerinde ortaya çıkabilir. Eğer β alanında n_j ifadesi varsa işlenecek işlerin operasyon sayıları sınırlanmıştır. Eğer β alanında n_j ifadesi yoksa operasyon sayılarında bir sınırlama yoktur.
- **İşlem sürelerinde sınırlamalar (p_j):** Eğer β alanında p_j ifadesi varsa işlerin işlem süreleri sınırlanmıştır. Eğer $p_j = p$ ise her işin işlem süresi *p* birimdir. Eğer β alanında p_j ifadesi yoksa işlem süreleri sınırlanmamıştır.
- **Serbest kalma zamanı (r_j):** Eğer bu ifade β alanına eklenmişse *j* işi başlamak için r_j zamanını beklemelidir. Eğer β alanında r_j yoksa *j* işi herhangi bir zamanda başlayabilir. Serbest kalma zamanı aksine d_j teslim zamanı (due date) problemde dikkate alınsa bile β alanına eklenmesine gerek yoktur. Teslim zamanının ortamda dikkate alındığı, amaç fonksiyonundan anlaşılır.

2.1.4. Amaç fonksiyonu

Çizelgelerin değerlendirilmesinde farklı optimallik kriterleri kullanılabilir. Amaç fonksiyonu olarak adlandırılan bu kriter, son çizelgede arzu edileni yansıtır ve tamamlanma zamanını, teslim zamanını veya maliyeti dikkate alabilir. Çizelgeleme problemleri sınıflandırmasında amaç fonksiyonları γ alanında gösterilir.

En çok çalışılan amaç fonksiyonu bütün çizelgenin tamamlanma zamanının C_{max} (makespan) minimizasyonudur (Shahzad 2010). *j* işinin *i* makinesinde tamamlanma süresi C_{ij} ile ve *j* işinin sistemi terketmesi C_j ile temsil edilirse, C_{max} aşağıdaki gibi gösterilebilir:

$$C_{max} = \max_{1 \leq j \leq n} C_j$$

Amaç fonksiyonu, teslim zamanlarının bir fonksiyonu da olabilir. *j* işinin gecikmesi $L_j = C_j - d_j$ ile tanımlanır. Bu değer, iş geç tamamlandıysa pozitiftir ve iş erken tamamlandıysa negatiftir. Maksimum gecikme L_{max} ile gösterilir ve şu şekilde tanımlanır:

$$L_{max} = \max_{1 \leq j \leq n} L_j$$

Diğer taraftan, *j* işinin pozitif gecikmesi şu şekilde gösterilir:

$$T_j = \max(C_j - d_j, 0) = \max(L_j, 0)$$

L_j ve T_j arasındaki fark T_j'nin negatif değer alamamasıdır. Pozitif gecikmeyi değerlendiren en çok kullanılan ölçüt, ortalama pozitif gecikmedir ($\bar{T} = \sum_j T_j/n$). Maksimum pozitif gecikme, bir çizelgeleme ortamındaki en kötü durumu temsil eder ve şu şekilde tanımlanır:

$$T_{max} = \max_{1 \leq j \leq n} T_j$$

Geç kalan işlerin sayısı ise aşağıdaki gibi tanımlanır. Gecikme, pozitif gecikme ve geç

$$U_j = \begin{cases} 1, \text{Eğer } L_j > 0, \\ 0, \text{Aksi halde.} \end{cases}$$

kalan işlerin sayısı, teslim zamanını dikkate alan üç temel amaç fonksiyonudur. Amaç fonksiyonları iki büyük sınıfta toplanabilir:

1. ***Minimax*:** Minimize edilecek fonksiyon kümesinin maksimum değeridir.
2. ***Minisum*:** Minimize edilecek fonksiyon kümesinin toplam değeridir.

Minimize edilecek olası amaç fonksiyonları aşağıdaki gibidir. Bu amaç fonksiyonları için γ alanında parantez içindeki ifadeler kullanılır. Literatürde en sık kullanılan "*Minimax*" amaç fonksiyonları aşağıdaki gibidir:

- **Maksimum tamamlanma zamanı (C_{max})**: $\max(C_1, ..., C_n)$ ile tanımlanan maksimum tamamlanma zamanı (makespan), sistemi en son terk eden işin tamamlanma zamanıdır.

- **Maksimum gecikme (L_{max})**: $\max(L_1, ..., L_n)$ ile tanımlanan maksimum gecikme, teslim zamanını en çok aşan işi ölçer.
- **Maksimum pozitif gecikme (T_{max})**: $T_{max} = \max_{1 \leq j \leq n} T_j$ ile tanımlanan maksimum pozitif gecikme, teslim zamanını aşan gecikmelerin en büyüğünü temsil eder.
- **Maksimum akış zamanı (F_{max})**: Literatürde tamamlanma zamanlarının toplamı *akış süresi* ile de tanımlanabilir. *j* işinin serbest kalma zamanı r_j ile gösterildiği durumda $F_j = C_j - r_j$ ile hesaplanır ve $F_{max} = \max_{1 \leq j \leq n} F_j$ ile tanımlanır. Sistemde en uzun kalan işi temsil eder ve amaç bu sürenin minimize edilmesidir.
- **Maksimum boş makine zamanı (I_{max})**: *i* makinesinin boş zamanlarının toplamı I_i ile temsil edildiği durumda $I_{max} = \max_{1 \leq i \leq m} I_i$ ile tanımlanır ve en çok boş kalan makinenin boş kalma süresi hesaplanır. Amaç, bu boş kalma sürelerinin minimize edilerek işlerin dengeli dağılımının sağlanmasıdır.
- **Maksimum erken bitirme (E_{max})**: *j* işinin erken tamamlanması $E_j = \max(0, d_j - C_j)$ şeklinde hesaplanır ve maksimum erken bitirme, $E_{max} = \max_{1 \leq j \leq n} E_j$ ile tanımlanır.

Çizelgeleme problemlerinde kullanılan "*Minisum*" amaç fonksiyonlarından bazıları; toplam akış zamanının ($\sum F$) minimizasyonu, toplam gecikmenin ($\sum L$) minimizasyonu, toplam makinelerin boş bekleme zamanlarının ($\sum I$) minimizasyonu ve toplam geciken iş sayısının ($\sum U$) minimizasyonu şeklinde olabilir. Bunlarla beraber, ele alınan fonksiyonlar ağırlıklandırılabilir ve bunların toplamları da dikkate alınabilir. Bu amaç fonksiyonlarına örnekler aşağıdaki gibidir:

- **Toplam ağırlıklı tamamlanma zamanı ($\sum w_j C_{max}$)**: *n* işin ağırlıklandırılmış tamamlanma zamanlarının toplamı, envanter veya elde tutma maliyetleri ile ilgili bir göstergedir. Literatürde tamamlanma zamanlarının toplamı *akış süresi* ve ağırlıklı tamamlanma zamanları toplamı da *ağırlıklı akış süresi* olarak kullanılabilir.
- **Azaltılmış toplam ağırlıklı tamamlanma zamanı ($\sum w_j(1 - e^{-rC_j})$)**: Bu maliyet fonksiyonu önceki amaç fonksiyonunun daha genel halidir ve maliyetler belli bir *r* oranında $0 < r < 1$ birim zaman başına azaltılmıştır. Eğer *j* işi *t* zamanına kadar tamamlanmamışsa [t, t+dt] periyodunda $w_j r e^{-rt}$ ilave maliyeti oluşur. Eğer *j* işi *t* zamanında tamamlanmışsa [0, t] periyodunda $w_j(1 - e^{-rt})$ maliyeti meydana gelir.

Genelde *r*, %10 (0,1) gibi sıfıra yakın bir değerdir.

- **Toplam ağırlıklı pozitif gecikme ($\sum w_j T_j$)**: Toplam ağırlıklı tamamlanma zamanının daha genel bir maliyet fonksiyonudur.
- **Ağırlıklı geç kalmış işlerin sayısı ($\sum w_j U_j$)**: Ağırlıklı geç kalmış işlerin sayısını hesaplamak çok kolaydır ve akademide de kullanılmasıyla birlikte pratikte de tercih edilen bir amaç fonksiyonudur.

Şimdiye kadar detaylandırılan çizelgeleme problemi notasyonu için aşağıdaki örnekler verilebilir.

- **Tek makine ortamı** ($1|r_j, prmp|\sum w_j C_j$): Ortamda tek makine vardır, *j* işi r_j zamanında serbest bırakılır ve işlerin bölünmesine izin verilir. Amaç fonksiyonu toplam ağırlıklı tamamlanma zamanının minimizasyonudur.
- **Paralel makine ortamı** ($Pm|r_j, M_j|\sum w_j T_j$)**:** Sistemde *m* adet paralel makine vardır. *j* işi r_j zamanında serbest bırakılır ve d_j teslim zamanına kadar işlenmelidir. *j* işi makineler alt kümesi olan M_j'deki makineler tarafından işlenebilir. Eğer *j* işi zamanında tamamlanmazsa $\sum w_j T_j$ cezası ortaya çıkar.

Çizelgeleme problemleri çok farklı yapılarda olabilir ve bu farklı yapılardaki problemlerin zorluk dereceleri de farklılık gösterir. Problemlerin zorluk derecesi *karmaşıklık sınıflandırması* (complexity classification) ile belirlenebilir. Çeşitli çizelgeleme problemlerinin karmaşıklık sınıflandırmaları sonraki bölümde detaylandırılmıştır.

2.2. Çizelgeleme Problemlerinin Karmaşıklık Sınıflandırması

Karmaşıklık teorisi, mantık-bilimciler ve bilgisayar mühendisleri tarafından geliştirilen matematiksel bir çerçeveyi temel almıştır. Bu teori, problemlerin zorluk derecelerini göstermek için geliştirilmiştir (Pinedo 2012).

Bir problemin zorluk derecesi, problemi en iyi çözen algoritmanın çözüm için gerektirdiği zamana bağlıdır (Chen ve ark. 1999). *Algoritma*; bir problemin çözümü için adım adım yapılması gerekeni barındıran bir prosedürdür. Verilen bir *x* girdisi için sonlu sayıda adımdan sonra doğru *f(x)* çıktısını üretir. Bir algoritmanın zaman karmaşıklığı,

en kötü durumda algoritmanın çözüm için ihtiyaç duyacağı süredir; örnek olarak toplama, çarpma ve karşılaştırma gibi temel operasyonların toplamı için gereken süredir. Eğer bir algoritmanın karmaşıklığı polinomiyal girdi ile sınırlı ise bu algoritmaya *polinomiyal* veya *polinomiyal süreli* denir (Chen ve ark. 1999).

Karmaşıklık teorisinde eğer bir algoritma polinomiyal ise iyidir, değilse kötü olarak değerlendirilir (Chen ve ark. 1999). Bu ifade çoğunlukla doğru olsa da bazı algoritmalarda geçerli değildir (Ör. Simpleks Algoritması) (Bazaraa ve ark. 2011). Benzer şekilde eğer bir problem polinomiyal bir algoritmaya sahipse kolay olarak değerlendirilir, bu durumda polinomiyal sürede çözülebilir demektir. *P* sınıfı problemleri polinomiyal çözülebilir problemlerin kümesidir.

Bir optimizasyon problemi karar problemi olarak formüle edilirse, herhangi bir x girdisi için *f(x)* çıktısının uygun çözüm olup olmadığı ("*Evet-Hayır*" kararı) onaylama adımlarını barındıran kısa bir *sertifika* ile belirlenebilir. Örnek olarak "bütün işlerin teslim zamanlarından önce tamamlandığı bir çizelge var mı" sorusunun herhangi bir *Evet* kararı, bütün işlerin zamanında tamamlandığı kısa bir sertifikaya sahiptir. Bu sertifika ile *Evet*-cevabı polinomiyal bir sürede doğrulanabilir. *NP* karmaşıklık sınıfı aşağıdaki koşulları sağlayan problemleri barındırır:

- Her *x Evet*-örneği için, $|y|$'nin $|x|$'de polinomiyal olarak sınırlandığı bir *y* sertifikası vardır.
- Verilen *y* sertifikasının *x* örneği için geçerli olup olmadığını doğrulayan bir polinomiyal algoritma vardır.

NP sınıfındaki en zor problemler *NP-tam* (*NP-complete*) problemlerdir. *NP-tam* problemler zor problemlerdir ve polinomiyal sürede çözülemezler. Örnek olarak $J||C_{max}$ problemi *NP-tam* sınıfındandır ve çözümü zordur.

Polinomiyal çözülebilirlik ve *NP-tamlık* aslında algoritmanın kodlama şemasında bağlıdır. Eğer kodlama şeması ikili sistemden birli sisteme çevrilirse problem daha kolay hale gelebilir ve algoritma hızlanabilir. *NP-tam* olan bir problem birli sistemdeyse

bu problem *güçlü NP-tam*'dır, çünkü basitleştirilmesi mümkün değildir. Eğer bir optimizasyon probleminin karar verme versiyonu *güçlü NP-tam* ise bu problem *güçlü NP-zor* olarak sınıflandırılır. Bir problem *NP-zor* olarak sınıflandırılırsa bunun güçlü olup olmadığı bilinmiyor demektir (Chen ve ark. 1999).

Çizelgeleme problemlerinde polinomiyal problemlerle *NP-zor* problemler arasındaki sınırı belirlemek araştırmacılar tarafından ilgi çekici olmuştur. Hatta bu sınırları belirlemek için önemli sayıda çalışma yapılmıştır. Ancak henüz birçok çizelgeleme probleminin karmaşıklığı belirlenememiş, dolayısıyla sınırlar bulanık kalmıştır (Pinedo 2012). Çizelgeleme problemlerini sınıflandırabilmek için yapılan çalışmalardan biri Lageweg ve ark. (1981) aittir. Araştırmacılar çizelgeleme problemlerini otomatik olarak sınıflandıran MSPCLASS adında bir bilgisayar programı geliştirmişlerdir (Lageweg ve ark. 1982). Bu program Graham ve ark. (1979) tarafından geliştirilen 3-alanlı $\alpha|\beta|\gamma$ sınıflandırma şemasını temel almış ve problemleri karmaşıklık sınıflarına ayırmıştır.

- Maksimal polinomiyal çözülebilir: Polinomiyal çözülebilir en zor problemlerdir. Bir diğer ifade ile polinomiyal çözülebilir olduğu bilinen problemlerin daha zor durumlarda polinomiyal olup olmadıkları bilinmemektedir.
- Maksimal sözde polinomiyal çözülebilir: Sözde polinomiyal (polinomiyal değil) çözülebilir en zor problemlerdir.
- Minimal *NP-zor*: *NP-zor* olan en basit problemlerdir. Bir diğer ifade ile, *NP-zor* olduğu bilinen, fakat daha basit durumlarda *NP-zor* olduğu bilinmeyen problemlerdir.
- Minimal açık: Karmaşıklığı henüz bilinmeyen, ancak bütün kolay durumlarda polinomiyal çözülebilir problemlerdir.
- Maksimal açık: Karmaşıklığı henüz bilinmeyen, ancak bütün zor durumlarda *NP-zor* olduğu bilinen problemlerdir.

Brucker ve Knust (2006a) çizelgeleme problemlerini sınıflandırabilmek için yeni bir bilgisayar programı (CLASS) geliştirmişlerdir. Bu çalışmalarında bir çok çizelgeleme problemini sınıflandırmışlardır (http://www.informatik.uni-osnabrueck.de/knust/class/, 2014). Bu sınıflandırmalara örnek olarak Çizelge 2.1 ve Çizelge 2.2 verilebilir. Bu

çizelgelerde işler *i* ile temsil edilmiştir. Ayrıca yazarlar araştırma yapılabilir bir bibliyografya oluşturmuşlardır (Brucker 2006b, http://www-desir.lip6.fr/∼durrc/query/, 2014). Bu sayede diğer araştırmacılar, bibliyografyadaki seçeneklerle problem özelliklerini belirleyerek problemin karmaşıklık sınıfını belirleyebilirler.

Çizelge 2.1. Maksimal polinomiyal çözülebilir problemler (Brucker 2006b)

Tek Makine	Paralel makine	Akış tipi atölye	Atölye tipi
$1\|prec; r_i\|C_{max}$	$P\|p_i = p; outtree; r_i\|C_{max}$		
$1\|prec; p_i = p; r_i\|L_{max}$	$P\|p_i = p; tree\|C_{max}$		
$1\|prec; r_i; pmtn\|L_{max}$	$Q\|p_i = p; r_i\|C_{max}$	$F\|p_{ij} = 1; outtree; r_i\|C_{max}$	
$1\|prec; p_i = p; r_i\|\sum C_i$	$Q2\|p_i = p; chains\|C_{max}$	$F\|p_{ij} = 1; tree\|C_{max}$	
$1\|prec; pmtn; p_i = p; r_i\|\sum C_i$	$P\|p_i = 1; chains; r_i\|L_{max}$	$F2\|\|C_{max}$	
$1\|r_i; pmtn\|\sum C_i$	$P\|p_i = p; intree\|L_{max}$	$F\|p_{ij} = 1; intree\|L_{max}$	
$1\|p_i = p; r_i\|\sum w_i C_i$	$P2\|p_i = p; prec\|L_{max}$	$F2\|p_{ij} = 1; prec; r_i\|L_{max}$	
$1\|sp-graph\|\sum w_i C_i$	$P2\|p_i = 1; prec; r_i\|L_{max}$	$F\|p_{ij} = 1; outtree; r_i\|\sum C_i$	
$1\|r_i; pmtn\|\sum U_i$	$P\|p_i = 1; outtree; r_i\|\sum C_i$	$F2\|p_{ij} = 1; prec; r_i\|\sum C_i$	
$1\|p_i = p; r_i\|\sum w_i U_i$	$P\|p_i = p; outtree\|\sum C_i$	$Fm\|p_{ij} = 1; intree\|\sum C_i$	
$1\|pmtn; p_i = p; r_i\|\sum w_i U_i$	$P2\|p_i = 1; prec; r_i\|\sum C_i$	$F\|p_{ij} = 1; r_i\|\sum w_i U_i$	
$1\|p_i = p; r_i\|\sum T_i$	$P2\|p_i = p; prec\|\sum C_i$	$F\|p_{ij} = 1; r_i\|\sum w_i T_i$	
$1\|pmtn; p_i = p; r_i\|\sum T_i$	$Pm\|p_i = p; tree\|\sum C_i$		
$1\|p_i = 1; r_i\|\sum w_i T_i$	$Qm\|p_i = p; r_i\|\sum C_i$		

Çizelge 2.2. Minimal NP-zor problemler (Brucker 2006b)

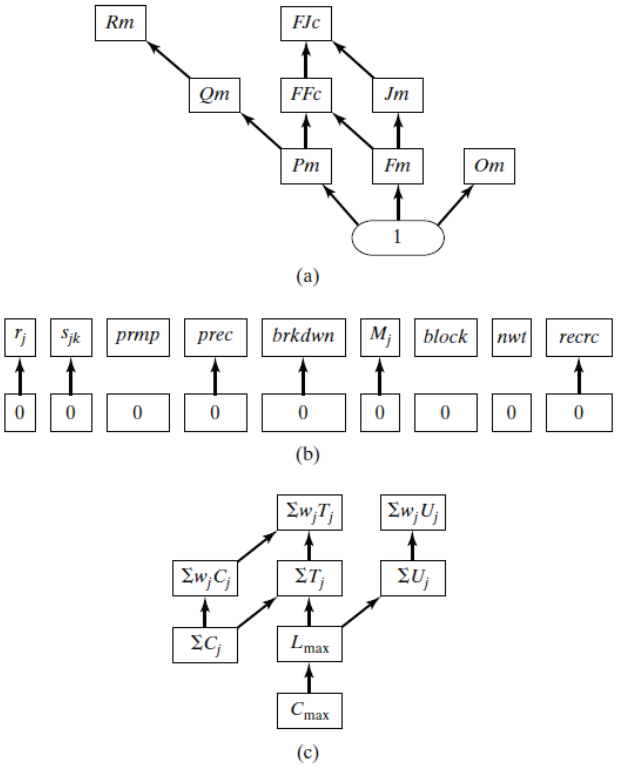
Tek Makine	Paralel makine	Akış tipi atölye	Atölye tipi
$1\|r_i\|L_{max}$	$P2\|\|C_{max}$	$F\|p_{ij} = 1; intree; r_i\|C_{max}$	
$1\|chains; r_i; pmtn\|\sum C_i$	$P\|\|C_{max}$	$F\|p_{ij} = 1; prec\|C_{max}$	
$1\|prec\|\sum C_i$	$P\|p_i = 1; intree; r_i\|C_{max}$	$F2\|chains\|C_{max}$	
$1\|r_i\|\sum C_i$	$P\|p_i = 1; prec\|C_{max}$	$F2\|r_i\|C_{max}$	
$1\|chains; p_i = 1; r_i\|\sum w_i C_i$	$P2\|chains\|C_{max}$	$F3\|\|C_{max}$	
$1\|prec; p_i = 1\|\sum w_i C_i$	$Q\|p_i = p; chains\|C_{max}$	$F\|p_{ij} = 1; outtree\|L_{max}$	
$1\|r_i; pmtn\|\sum w_i C_i$	$P\|p_i = 1; outtree\|L_{max}$	$F2\|\|L_{max}$	
$1\|chains; p_i = 1\|\sum U_i$	$P\|p_i = 1; intree; r_i\|\sum C_i$	$F2\|\|\sum C_i$	
$1\|\|\sum w_i U_i$	$P\|p_i = 1; prec\|\sum C_i$	$F2\|p_{ij} = 1; chains\|\sum w_i C_i$	
$1\|\|\sum T_i$	$P2\|chains\|\sum C_i$	$F3\|p_{ij} = 1; chains\|\sum w_i C_i$	
$1\|chains; p_i = 1\|\sum T_i$	$P2\|r_i\|\sum C_i$	$F2\|p_{ij} = 1; chains\|\sum U_i$	
$1\|\|\sum w_i T_i$	$P2\|\|\sum w_i C_i$	$F3\|p_{ij} = 1; chains\|\sum U_i$	
	$P\|\|\sum w_i C_i$	$F2\|p_{ij} = 1; chains\|\sum T_i$	
	$P2\|p_i = 1; chains\|\sum w_i C_i$	$F3\|p_{ij} = 1; chains\|\sum T_i$	
	$P2\|p_i = 1; chains\|\sum U_i$		
	$P2\|p_i = 1; chains\|\sum T_i$		

Atölye tipi
$J2\|p_{ij} = 1; r_i\|C_{max}$
$J2\|p_{ij} = 1\|\sum C_i$
$J2\|p_{ij} = 1\|\sum U_i$
$J\|prec; p_{ij} = 1; r_i; n = k\|\sum w_i U_i$
$J\|prec; r_i; n = 2\|\sum w_i U_i$
$J2\|n = k\|\sum w_i U_i$
$J\|prec; p_{ij} = 1; r_i; n = k\|\sum w_i T_i$
$J\|prec; r_i; n = 2\|\sum w_i T_i$
$J2\|n = k\|\sum w_i T_i$
$J3\|n = 3\|C_{max}$
$J2\|\|C_{max}$
$J2\|chains; p_{ij} = 1\|C_{max}$
$J3\|p_{ij} = 1\|C_{max}$
$J2\|p_{ij} = 1; r_i\|\sum C_i$
$J3\|n = 3\|\sum C_i$
$J2\|\|\sum C_i$
$J2\|chains; p_{ij} = 1\|\sum C_i$
$J3\|p_{ij} = 1\|\sum C_i$
$J2\|p_{ij} = 1\|\sum w_i C_i$
$J2\|p_{ij} = 1; r_i\|\sum w_i C_i$
$J2\|p_{ij} = 1; r_i\|\sum U_i$
$J2\|p_{ij} = 1\|\sum w_i U_i$
$J2\|p_{ij} = 1\|\sum w_i T_i$

Pinedo (2012) çalışmasında deterministik çizelgeleme problemlerinin karmaşıklık hiyerarşisini Şekil 2.1'deki gibi belirlemiştir. Bu çalışmalarla da ortaya koyulduğu gibi çizelgeleme problemleri çok çeşitlidir ve farklı karmaşıklık derecelerine sahiptir. Tez çalışmasında ele alınan problem tanımı bir sonraki bölümde anlatılmıştır.

2.3. Ele Alınan Çizelgeleme Problemi

Ele alınan çizelgeleme probleminde *n* adet iş vardır. Her *i* işi, bir veya birden fazla operasyondan oluşur. Bir işin tamamlanması için işin *j* adet operasyonu *j* adet ardışık makinede işlenmelidir. Makine ortamında *m* adet ($m > j$) makine vardır. Bu makineler birbirlerinden farklıdır ve her iş her makinede işlenemez. *Makine elverişlilik* kısıtları



Şekil 2.1. Karmaşıklık hiyerarşisi. (a) makine ortamı, (b) işlem sınırlamaları ve kısıtları, (c) amaç fonksiyonları (Pinedo 2012)

altında her işin işlenebileceği makine kümesi belirlenir ve işler bu küme elemanlarına göre atanır. Her işin her operasyonu atandığı makinede işlenebilmesi için bir kaynağa (kalıp) ihtiyaç duymaktadır (Slowinski 1980, Blazewicz ve ark. 1983, Ventura ve Kim 2000) ve bu kaynak her operasyon için ayrıdır ve bir adet bulunur. Makinelere işlerin ataması yapıldıktan sonra her operasyona ait olan ekipman da makineye bağlanır. Dolayısıyla, bir iş aynı anda birden fazla makinede işlenemez. Bütün bu kısıtlar dikkate alındığında, ele alınan problemde makine elverişlilik ve kaynaklar kısıtları vardır ve *m* adet makine sayısı aşılmayacak şekilde birden fazla iş eş zamanlı olarak makine ortamında işlenebilir, dolayısıyla bağımsız paralel makine ortamı oluşmaktadır. Ayrıca işlerin birbirlerinden farklı operasyonları ardışık makinelerde işlenmelidir ve bu ardışıklık/zincir özelliğinden dolayı problem, işlerin başlangıç makinelere atanmasından sonra akış tipi çizelgeleme problemine dönüşmektedir. Dolayısıyla, 3-alanlı $\alpha|\beta|\gamma$ sınıflandırması kullanılarak problem $R_m|M_i; res; chains|C_{max}$ şeklinde tanımlanabilir.

Bu problem tanımı için bir sınıflandırma bulunamamıştır, ancak Garey ve ark. (1978) $R_m|M_i|C_{max}$ problemini *güçlü NP-zor* olarak sınıflandırmıştır. Bu probleme birden fazla operasyon için ardışıklık kısıtı ve kaynak kısıtı eklendiğinde elde edilecek problem de *NP-zor* olarak sınıflandırılabilir. Literatürdeki $R_m|M_i; res; chains|C_{max}$ problemine benzer bağımsız paralel makine çizelgeleme çalışmaları sonraki bölümde ele alınmış ve problemlere geliştirilen yöntemler ayrıntılandırılmıştır.

3. LİTERATÜR TARAMASI

Çizelgeleme problemleri Henry Gannt (1919) tarafından 1900'lerin başında gerçekleştirilen çalışmalarla ilk kez ciddi bir şekilde ele alınmış, ancak ilk yayınların literatürde yer alması uzun sürmüştür. 1950'lerde Naval araştırma merkezindeki çalışmalarla ortaya çıkan ilk yayınları (Jackson 1956, Wagner 1959), 1960'larda çizelgeleme problemlerinin dinamik programlama ve tamsayılı programlama ile modellenmesi çalışmaları izlemiştir (Held ve Karp 1962). 1970'lerdeki çalışmaların odak noktası, Richard Karp (1972) ile başlayan karmaşıklık teorisi olmuştur. 1980'lerde bir çok alanda literatürde ve endüstride uygulamalı çalışmalar gerçekleştirilmiştir. 1990'larda literatür taramalarıyla zenginleştirilen çalışmalar, sezgisel yöntemlerin çizelgeleme uygulamaları ile arttırılmıştır. 2000'li yıllardan itibaren çizelgeleme çalışmaları büyük bir hız kazanmıştır. Bu yıllarda kişisel bilgisayarların kullanımının yaygınlaşması, problemlerin modellenmesi aşamasında çizelgeleme çalışmalarına önemli katkıda bulunmuştur (Pinedo 2012). Sonuç olarak, yüzyılı aşkın süredir çizelgeleme problemleri araştırmacılar tarafından çalışılmakta ve bu çalışmalar yeni algoritmalarla desteklenerek devam etmektedir.

Çizelgeleme problemleri, tanımında kullanılan üç farklı alana; makine ortamına, iş karakteristiğine ve optimallik kriterine göre çeşitlendirilmektedir. Temelde problemin türünü belirleyen makine ortamında tek makine, paralel makine, akış tipi atölye, atölye tipi ve açık tip olarak genellenebilen farklı sınıflar mevcuttur. Üç alanda yer alan dokuzdan fazla farklı sınıfın kombinasyonu ile çeşitlendirilebilen çizelgeleme problemleri ile hala çalışılmamış ve literatürde açık problem olarak nitelendirilen problemler de dikkate alındığında, çizelgelemenin ne kadar geniş bir alana yayıldığı görülebilir. Ayrıca, aynı çizelgeleme problemi için kesin yöntemler, sezgisel yöntemler gibi farklı çözüm yöntemlerinin geliştirildiği de göz önüne alınırsa yapılan çalışmaların katlanarak arttığı anlaşılabilir. Bu çalışmada, muazzam genişlikteki çizelgeleme çalışmalarından bağımsız paralel makine çizelgeleme çalışmaları ele alınmıştır.

Literatürdeki birçok paralel makine çizelgeleme alanındaki çalışmalar, çeşitli tarama çalışmaları ile bir araya getirilmiştir (Cheng ve Sin 1990, Mokotoff 2001, Pfund ve ark. 2004, Li ve Yang 2009, Kravchenko ve Werner 2011, Edis ve ark. 2013). Chen ve Sin (1990) çalışmalarında literatürdeki paralel makine çizelgeleme problemlerini geniş bir şekilde incelemiştir. Pfund ve ark. (2004) bağımsız paralel makine çizelgeleme çalışmalarını ele almış ve tek ve çok amaçlı çizelgeleme problemlerini gruplandırmışlardır. Hazırlık süresinin olmadığı ve işlerin bölünmesine izin verilmeyen deterministik çalışmaları incelemişlerdir. Bu çalışmalar için geliştirilen modelleri özetlemişler ve problemlerin karmaşıklık sınıflarını tanımlamışlardır. Diğer deterministik çizelgeleme alanlarına göre bağımsız paralel makine çizelgeleme problemlerinin daha az çalışıldığını ortaya koymuşlardır. Ayrıca yazarlar bu tarama sonrasında hangi alanlarda boşluklar olduğunu detaylandırmışlardır.

Framinan ve Ruiz çalışmalarında (2010), literatürde yer alan üretim çizelgeleme sistemlerinin yapısını incelemiş ve ileriki çalışmalar için önerilerde bulunmuşlardır. Literatürde üretim çizelgeleme ile ilgili pek çok yayın olmasına rağmen ortaya koyulan ve çalışılan modellerin, prosedürlerin pratikte uygulamalarının azlığından yakınmışlar ve bu durumun teori ile pratik çizelgeleme arasında bir boşluk oluşturduğuna değinmişlerdir. Reisman ve ark. (1997) tarafından gerçekleştirilen bir araştırmada incelenen yüz seksen dört çizelgeleme yayınından sadece beş tanesi (%3'den az) gerçek üretim sistemlerini ele almıştır. Bu farkın kapatılması için teorik çizelgeleme problemlerinin işletmelerdeki çizelgeleme sistemlerine dönüştürülmesi gerekmektedir. Bu dönüşüm, üretim çizelgeleme sistemlerinin mimarisini barındıran yazılımlarla gerçekleştirilebilir. Bu yazılımlar işletmelere göre farklı özellik gösterseler de hepsinde bulunan ortak noktalar da vardır. Yazarlar, üretim çizelgeleme sistemlerinin mimarisini ele alan çalışmaları incelemişler ve çeşitli yazarlar tarafından ortaya koyulan katkıları gruplandırmışlardır. Daha sonra bu çalışmaları problemin modellenmesi, çözümü, çözümün değerlendirilmesi, kapasite analizi ve kullanıcı arayüzü alt bölümlerine göre sınıflandırmışlardır. Bu sınıflandırmalardan yararlanarak çizelgeleme sistemleri için modüler bir mimari geliştirmişlerdir. Bu mimaride ortaya koyulan dört ana modülü çalışmalarında ayrıntısıyla okuyuculara aktarmışlardır. Son olarak yazarlar,

teorideki çizelgeleme algoritmalarının gerçek sistemlerde başarıyla uygulanması için işletmelerin üretim çizelgeleme süreçlerinin detaylarının incelenmesi gerektiğini ve bu detayların modelin senaryosuna, kısıtlarına ve amaç fonksiyonuna yansıtılması gerektiğini vurgulamışlardır.

Üretim sistemlerinin pratik uygulamalarında bağımsız paralel makine çizelgeleme problemleri için geliştirilen çözüm yöntemleri, üretimi en etkin şekilde gerçekleştirmek amacıyla kullanılmaktadır (Edis ve ark. 2013). Bu amaçla, gerçek hayat problemlerini konu alan sezgisel ve kesin yöntemler olarak gruplandırılabilen birçok çalışma gerçekleştirilmiştir.

Gerçek hayat problemlerine sezgisel bir yaklaşımla başarılı çözümler üretebilen çalışmalardan biri Barlatt ve ark. (2012) tarafından gerçekleştirilmiştir. Barlatt ve ark. (2012) Ford firmasındaki pres operasyonlarını incelemişlerdir. Araştırmacılar, özdeş preslerde tek operasyonlu işlerin çizelgelenmesini ele almışlar, çizelgeleme problemini sıralama problemine çevirmişler ve Test-Kes (Test-Prune) algoritmasını geliştirmişlerdir. İş sıralama problemi ile vardiya seçimini birlikte incelemişler ve bir karar destek sistemi (JEDI) geliştirmişlerdir. Araştırmacılar JEDI sistemi ile tedarik zinciri ve işgücü ataması da dahil olmak üzere bütün üretim ortamını gözlemleyebilmişlerdir. Ayrıca üretim planlama, çizelgeleme ve işgücü atama için bileşik karar değişkenleri kullanarak pres çizelgeleme programı (SSO) geliştirmişlerdir. JEDI sistemi ile başlangıç çizelgeyi üretmişler ve karışık tamsayılı programlama yerine SSO programı ile süreci devam ettirmişlerdir. Bu sayede etkin çizelgeler üretebilmişlerdir.

Liao ve ark. (2014) rıhtıma gelen kamyonların sıralanması problemini bağımsız paralel makine problemine dönüştürmüşlerdir. Gelen kamyonları iş olarak ve rıhtım bölümlerini de makine olarak tanımlamışlardır. Dolayısıyla iş-makine atamasının en iyi şekilde gerçekleştirilmesi, problemin amacı olmuştur. Yazarlar, literatürde geliştirilen beş farklı meta-sezgiseli ele almışlar ve kendi problem tanımlarına göre yeniden düzenlemişlerdir. Bu sezgisellerden üçü karınca kolonisi optimizasyon değişkenlerini temel almış ve diğer ikisi yazarların kendi deneyimlerine dayanarak oluşturdukları hibrit tavlama benzetimi

algoritmasını temel almıştır. Değerlendirme kriteri olarak çözüm süresi ve toplam tamamlanma zamanının minimizasyonu olan amaç fonksiyonu dikkate alınmıştır. İki farklı uygulama problemi kullanılarak gerçekleştirilen karşılaştırma sonuçlarına göre tavlama benzetimi&tabu araması kullanan hibrit yöntem ile 2-aşamalı karınca kolonisi sezgiseli en başarılı algoritmalar olmuşlardır.

Bağımsız paralel makine çizelgeleme problemleri için kullanılan sezgisel yöntemlerden biri de genetik algoritmadır (Tavakkoli-Moghaddam ve ark. 2009, Lin ve ark. 2011, Liu 2013). Lin ve ark. (2011) bağımsız paralel makine çizelgeleme problemi ve farklı amaç fonksiyonları için çeşitli sezgisel algoritmalar geliştirmişlerdir. Geliştirilen sezgisel yöntemlerden ilki tamsayılı modelin gevşetilmesi ile başlangıç atamaların elde edilmesi ve sonrasında bir sezgiselle devam edilmesi şeklinde iki aşamalı bir yöntemdir (IP/Roundup). Bu yöntem haricinde HEU-I, HEU-II ve ATC-I olarak adlandırılan üç farklı sezgisel algoritma geliştirmişlerdir. Ayrıca bir genetik algoritma yaklaşımı geliştirerek, diğer sezgisellerle karşılaştırma yapmışlardır. Sezgiseller içinden HEU-I en başarılı sonuçları üretmiştir. Diğer taraftan, IP/Roundup yöntemi sezgisel algoritmalardan daha başarılı olmuştur. Ancak geliştirilen genetik algoritma yöntemi diğer bütün yöntemlere üstün gelmiştir.

Liu (2013) işler arası öncelik kısıtları olan bir bağımsız paralel makine çizelgeleme problemini ele almış ve çözüm için bir hibrit genetik algoritma yaklaşımı geliştirmiştir. Amaç fonksiyonu olarak toplam gecikmenin minimizasyonu dikkate alınmıştır. Hibrit genetik algoritmanın başlangıç çözümü olarak bir sezgisel algoritma geliştirilmiştir. Bu algoritma ile öncelik kurallarına göre işler makinelere atanmakta ve ilk uygun atama belirlenerek, genetik algoritma bu atama ile başlatılmaktadır. Küçük ve büyük boyutlu problemlerle gerçekleştirilen deneylerde hibrit genetik algoritma ile başarılı sonuçlar elde edilmiştir.

Bağımsız paralel makine çizelgeleme problemleri için geliştirilen sezgisel çözüm yöntemleri gün geçtikçe artmakta ve karmaşıklaşmaktadır. Bununla birlikte sezgisel yöntemlere alternatif olan kesin yöntemler de çizelgeleme problemlerinde uygulama

alanı bulabilmektedir. Genelde, çizelgeleme problemlerinin kabul edilebilir sürelerde optimal çözümlerinin kesin yöntemlerle bulunması, problemlerin NP-zor yapısından dolayı oldukça zordur (Tran ve Beck 2012). Ancak, son yıllarda kısıt programlama ve uygulamaları çizelgeleme problemlerinin kesin çözümünü üretebildiği için bu alanda popüler hale gelmiştir (Van Hentenryck 1999). Malapert ve ark. (2012) yığın işleme problemini de barındıran çizelgeleme çalışması için bir kısıt programlama yaklaşımı geliştirmişlerdir. Kısıt programlama ile karşılaştırmak için ayrıca problemin matematiksel modelini de oluşturmuşlardır. Karşılaştırma sonuçlarına göre kısıt programlama modeli, çözüm süresi ve çözüm kalitesi alanlarında matematiksel programlama modeline üstün gelmiştir. Yazarlar, bu problem tipinde kısıt programlamanın matematiksel programlamaya alternatif olabileceğini göstermişlerdir.

Tamsayılı ve karışık tamsayılı programlamaya alternatif olarak kullanılan kısıt programlama yöntemi, sıralama ve çizelgeleme problemlerinde hızlı şekilde uygun çözümleri elde edebilmek gibi kendine özgü avantajlara sahiptir (Smith ve ark. 1996, Jain ve Grossmann 2001, Lustig ve Puget 2001). Ancak kısıt programlama, doğrusal gevşetmelerden elde edilen alt sınıra sahip olmadığı için problem ile ilgili genel bir bakış açısı veya bir diğer deyişle mevcut çözümün hassasiyeti bilgisini verememektedir. Tamsayılı programlama ve kısıt programlama karmaşık çizelgeleme problemlerinde farklı avantajlara sahip oldukları için bu iki farklı yöntemi bir araya getiren çalışmalar son yıllarda artış göstermiştir (Jain ve Grossmann 2001, Chu ve Xia 2005, Hooker 2005, Hooker 2007, Edis ve Ozkarahan 2011, Edis ve Ozkarahan 2012), Tran ve Beck 2012.

Jain ve Grossman (2001) çalışmalarında karışık tamsayılı programlama (KTP) ve kısıt programlama (KP) ile çözülemeyen problemlere alternatif çözüm yöntemleri geliştirebilmeyi amaçlamışlardır. Yazarlar, matematiksel modellerinde sadece ikili değişkenlere ve sıfır olmayan amaç fonksiyonu katsayılarına sahip problem sınıflarını dikkate almışlardır. Bu problemler bir hibrit KTP/KP modeli ile tanımlanmışlardır. Bu modelde KTP kısıtları, KP kısıtları ve KTP ile KP değişkenleri arasındaki eşitlik ilişkisini gösteren kısıtlar yer almaktadır. Hibrit modelleri çözebilmek için KTP/KP tabanlı bir ayrıştırma yöntemi ve doğrusal programlama ile KP temelli bir dal-sınır

algoritması geliştirilmiştir. Örnek problem olarak serbest kalma ve teslim zamanlarına sahip işlerin paralel makinelerde işlenmesi ele alınmıştır. Problem için geliştirilen KTP, KP, birleştirilmiş KTP-KP (Van Hentenryck 1999) ve hibrit KTP/KP modelleri karşılaştırmalarda kullanılmıştır. Birçok farklı veri kümesinde yapılan deneyler, hibrit KTP/KP modelinin diğer yöntemlere göre iki-üç kat çözüm süresini azalttığı görülmüştür.

Edis ve ark. (2012) bir enjeksiyon kalıp fabrikasındaki kaynak kısıtlı paralel makine çizelgeleme problemini ele almışlardır. Toplam tamamlanma zamanının minimizasyonunu (makespan) hedef alan bir tamsayılı matematiksel model (IP) geliştirmişlerdir. Ancak, yüksek sayıda değişken ve kısıt barındıran problemin optimal çözümüne ulaşılamamıştır. Bu sebeple araştırmacılar kısıt programlamayı (CP) dikkate almışlar ve bütün problemi iki alt probleme ayrıştırarak (atama ve sıralama problemi) iki farklı çözüm yaklaşımı (IP/IP ve IP/CP) geliştirmişlerdir. Karşılaştırmalar sonucunda IP/CP modelinin hızlı ve pratik çözümleri sağladığını ve bir dakikadan daha az sürede etkin çözümlere ulaştığını gözlemlemişlerdir. Ancak, IP/IP yöntemi test problemlerinde daha iyi bir performans göstermiştir.

Son yıllarda araştırmacılar farklı metodolojilerin güçlü yönlerini birleştirmek amacıyla bazı ayrıştırma algoritmalarını kullanarak hibrit metotlar geliştirme eğilimi göstermektedirler (Harjunkoski ve Grossmann 2002, Cambazard ve ark. 2004, Chu ve Xia 2005, Hooker 2005, Hooker 2007, Canto 2008, Fazel-Zarandi ve Beck 2009, Coban ve Hooker 2010, Edis ve Ozkarahan 2011, Edis ve Ozkarahan 2012, Heinz ve Beck 2012, Tran ve Beck 2012, Xiaolu ve ark. 2014). Benders ayrıştırma yöntemi en popüler ayrıştırma yöntemlerinden olsa da alt-problem için doğrusal model gerektirmekte, ancak çizelgeleme problemlerinin kombinatoryal yapısı tamsayılı programlamaya ihtiyaç duymaktadır. Hooker (1995) kombinatoryal problemlerde ayrıştırma tekniklerinin düzgün uygulanabilmesi için mantık-tabanlı Benders ayrıştırma (MTBA) yöntemini geliştirmiştir (Hooker ve Yan 1995, Hooker ve Ottosson 2003, Hooker 2011).

Hooker (2005) çalışmasında üretim ve tedarik zinciri ortamlarında sıklıkla rastlanan bir planlama ve çizelgeleme problemini ele almıştır. İşler makinelere atanmakta ve

serbest kalma zamanı ve teslim zamanına göre çizelgelenmektedir. Geç kalan işlerin minimizasyonu ve toplam gecikmenin minimizasyonu olmak üzere iki farklı amaç fonksiyonu dikkate alınmıştır. Problem sadece kısıt programlama (KP) veya karışık tamsayılı programlama (KTP) ile modellenebilse de bu modellerin çözümü oldukça zordur. Bu amaçla yazar, KP ve KTP modellerini mantık-tabanlı Benders ayrıştırma (MTBA) yöntemi ile birleştirmiştir. Ana problem olarak atama problemi ele alınmış ve KTP ile çözülmüş, çizelgeleme problemi ise alt-problem KP ile çözülmüştür. Üretilen örneklerle gerçekleştirilen karşılaştırmalar sonucunda MTBA ile çözüm süresi oldukça iyileştirilmiş ve optimali bulunamayan problemler için kısa sürelerde başarılı çizelgeler elde edilebilmiştir.

Çoban ve Hooker (2010) zaman pencereli, uzun zaman periyotlu bölünemeyen işlerin tek makinede çizelgelenmesi problemini ele almışlardır. Problem için bir MTBA algoritması geliştiren araştırmacılar ana problemi, işlerin önceden tanımlanan zaman periyotlarına atanması olarak ve alt-problemi, işlerin çizelgelenmesini olarak tanımlamışlardır. Toplam tamamlanma zamanının minimizasyonu ve toplam gecikmenin minimizasyonu olarak iki farklı amaç fonksiyonu kullanılmıştır. Karşılaştırmalar sonucunda tek makine çizelgeleme problemlerinde MTBA algoritmasının KP ve KTP yöntemlerine üstün geldiğini ortaya koymuşlardır.

Tran ve Beck (2012) çalışmalarında makine ve sıra bağımlı hazırlık sürelerini barındıran bağımsız paralel makine çizelgeleme problemi ele almışlardır. Toplam üretim zamanının minimizasyonunu hedefleyen problem için bir MTBA algoritması geliştirmişlerdir. Ana problem olarak sıralama problemi tanımlanmış ve karışık tamsayılı programlama kullanılmıştır. Alt-problemler için gezgin satıcı problemi için özelleştirilmiş bir çözücü kullanılmıştır. Ana problem iş-makine atamasını gerçekleştirmiş ve alt-problemler her makinede optimal çözümleri elde etmeyi amaçlamışlardır. Karşılaştırma sonuçları incelendiğinde MTBA ile literatürdeki problemlerin altı katı büyüklüğündeki problemlerin karışık tamsayılı modelden altı kat hızlı şekilde çözüldüğü görülmüştür. Ayrıca MTBA algoritmasının benzer problemler için geliştirilen dal-sınır algoritmasından iki kat daha başarılı olduğu gösterilmiştir.

Ele alınan çalışmalar incelendiğinde paralel makine çizelgeleme problemlerinde KP ve KTP yöntemlerinin başarılı bir şekilde mantık-tabanlı Benders ayrıştırma yöntemi ile bir arada kullanılabileceği görülmüştür. Ayrıca, tez çalışması kapsamında ele alınan çizelgeleme problemi, makine elverişlilik ve kaynak kısıtları altında ardışık makinelerde işlenmesi gereken farklı sayıda operasyona sahip işlerin bağımsız paralel makine ortamında eş zamanlı olarak çizelgelenmesi ($R_m|M_i; res; chains|C_{max}$) problemi şeklinde sınıflandırılabilir. Paralel makine problemleri literatürde çokça çalışılmasına rağmen, bu tanımda bir probleme ve dolayısıyla çözüm yöntemine literatürde rastlanmamıştır. Bu sebeple, ilk önce problemin karışık tamsayılı modeli oluşturulmuştur. Sonrasında kısıt programlama yöntemi ile problem modellenmiş ve çeşitli deneyler gerçekleştirilmiştir. Son olarak, literatürde de başarılı uygulamaları özetlenen mantık-tabanlı Benders ayrıştırma yöntemi kullanılmıştır. Teorik olarak Bölüm 4'de detaylandırılan yöntemlerin bir uygulama probleminde kullanılması, Bölüm 5'de açıklanmıştır.

4. ÇALIŞMADA KULLANILAN YÖNTEMLER

Bu bölümde, çalışmada kullanılan çözüm yöntemleri açıklanmıştır. Ele alınan $R_m|M_i; res; chains|C_{max}$ problemi literatür taramasında da detaylandırıldığı gibi henüz çalışılmamış bir problemdir. Dolayısıyla bu problemi temel hatlarıyla ortaya koyan bir matematiksel model de mevcut değildir. Bu sebeple problem çözümünde ilk kullanılan yöntem, Bölüm 4.1'de detaylandırılan matematiksel programlamadır. Son yıllarda matematiksel programlamaya alternatif olarak kısıt programlama teknikleri kullanılmaya başlanmıştır. Literatürde çizelgeleme problemlerine başarıyla uygulanan kısıt programlama, Bölüm 4.2'de açıklanmış ve bu probleme uyarlanmıştır. Son olarak, *NP-zor* olan problem ayrıştırma teknikleri ile basitleştirilmeye çalışılmış ve Bölüm 4.3'de anlatılan mantık-tabanlı Benders ayrıştırma metodu kullanılmıştır.

4.1. Matematiksel Programlama

Matematiksel programlama, yöneylem araştırmasının araçlarından biridir. Yöneylem araştırması, ilk olarak II. Dünya Savaşı sıralarında askeri operasyonlarda kullanılmıştır. Savaş sırasında yetersiz kaynakların farklı askeri operasyonlara dağıtılması acil olarak gündeme gelmiştir. Bu sebeple, İngiliz ve Amerikan askeri yönetimi, çok sayıda bilim adamı ile irtibata geçerek bu tarz ve diğer stratejik ve taktiksel problemler için bilimsel bir metodun geliştirilmesini istemişlerdir. Aslında bir anlamda askeri operasyonların araştırılması (operations research) istenmiştir. Radarların kullanılması için geliştirilen etkin metotlar ile İngiltere Hava Savaşı kazanılmıştır. Sonrasında, konvoyların ve denizaltıların yönetimini en iyileyen yöntemlerle Kuzey Atlantik Savaşı kazanılmış, Pasifik'te de benzer başarılar elde edilmiştir. Savaş sona erdiğinde yöneylem araştırmasının başarıları, yaklaşımın askeri problemler dışındaki problemlere de uygulanmasını hızlandırmıştır. 1950'lerde yöneylem araştırması teknikleri iş dünyasında, endüstri ve devlet alanlarında kullanılmaya başlanmıştır. Hızlı yayılım, beraberinde yeni çözüm tekniklerinin gelişimini de getirmiştir. 1950'lerin sonunda matematiksel programlamanın alt kümeleri olan doğrusal programlama, tamsayılı programlama, dinamik programlama, kuyruk teorisi ve envanter teorisi ile ilgili çözüm

yöntemleri geliştirilmiştir. Örnek olarak, 1947'de George Dantzig tarafından geliştirilen simpleks algoritması günümüzde de doğrusal programlama problemleri çözümünde kullanılmaktadır (Hillier ve Lieberman 2001).

Matematiksel modeller gerçek hayat problemlerini, karar değişkenleri ve parametreleri kullanarak temsil eder ve geliştirilen modeller matematiksel programlama teknikleri ile çözülür. Bütün matematiksel modeller, kısıtlar ve amaç fonksiyonu olmak üzere iki kısımdan oluşur. Kısıtlar, problemin özelliklerini temsil edip sistemi tanımlarken, amaç fonksiyonu karar vericinin isteklerini gösterir. Matematiksel modeller karar değişkenlerinin tanım kümesine bağlı olarak çeşitlilik gösterir. Örnek olarak, doğrusal programlama modellerinde karar değişkenleri doğrusal olarak tanımlanır ve modeldeki matematiksel fonksiyonlar, doğrusal fonksiyonlardır. Doğrusal programlama modellerinin çözümü için ilk olarak 1947'de George Dantzig tarafından simpleks algoritması geliştirilmiştir. Günümüze kadar bir çok çalışma ile bu algoritmanın en etkin çözüm metodu olduğu gösterilmiştir (Brearley ve ark. 1975, Spielman ve Teng 2004). Doğrusal programlama çözümünde kullanılan diğer yöntemler dual simpleks yöntemi, parametrik doğrusal programlama ve üst sınır tekniği gibi yöntemlerdir (Hillier ve Lieberman 2001). Doğrusal programlamada çözümü kolaylaştıran değişkenlerin doğrusal oluşu varsayımı, pratikte pek uygulanabilir değildir. Örnek olarak işlerin, insanların ve araçların aktivitelere atanması tamsayı değerlerinde olmalıdır. Eğer bir modelde bütün değişkenler tamsayı olarak tanımlanmışsa bu modele tamsayılı programlama modeli denir. Eğer değişkenlerden bazıları tamsayı bazıları doğrusal olarak tanımlanmış ise karışık tamsayılı programlama modeli ortaya çıkar. Tamsayılı değişken barındıran modellerin çözümünde dal-sınır algoritması ve dal-kes algoritması gibi yöntemler kullanılır (Hillier ve Lieberman 2001). Bu çalışmada ele alınan $R_m|M_i; res; chains|C_{max}$ problemi karışık tamsayılı ve tamsayılı programlama ile ilerleyen bölümlerde modellenmiştir. Problem çözümünde matematiksel programlamaya alternatif olarak kullanılan kısıt programlama bir sonraki bölümde anlatılmıştır.

4.2. Kısıt Programlama

Son yıllarda, kısıt programlamanın da tamsayılı programlama gibi kombinatoryal problemlerin temsilinde ve çözümünde başarılı olduğu görülmüştür (Hentenryck 2002). Kısıt programlamanın doğuşu tarihsel olarak incelendiğinde köklerinin 1960'lara *mantık programlamaya* (logic programming) dayandığı görülür. Kısıt kavramı, 1963'te Sketchpad interaktif çizim sisteminde kullanılmıştır. 1970'lerde kısıt çözümü için birçok programlama dili önerilmiştir. Yine bu tarihlerde yapay zeka alanındaki araştırmacılar tarafından *kısıt sağlama problemi* (constraint satisfaction problem) konsepti oluşturulmuştur. Ayrıca, *yerel tutarlılığı* (local consistency) sağlayan algoritmalar ve bir çok *arama metodu* (search method) geliştirilmiştir. Bu arama algoritmalarından bazıları (*geri izleme-backtracking*) 19.yüzyıla dayanırken, dal&sınır gibi diğerleri kombinatoryal optimizasyon kaynaklıdır. Kısıt programlamanın asıl katkısı, bilinen teknikleri çeşitli *kısıt yayılımı* (constraint propagation) algoritmaları ile birleştirerek yeni arama yöntemlerinin geliştirilmesine olanak sağlamasıdır (Apt 2003).

1980'lerde ilk kısıt programlama dilleri tanıtılmış ve uygulamaları ortaya koyulmuştur (Cohen 1988). Bunlardan en önemlileri mantık programlama paradigması ile ilgili olanlardır. Bu diller, mantık programlamaya kısıtların dahil edilmesiyle ortaya çıkan *kısıt mantık programlamanın* (constraint logic programming) gelişimine ön ayak olmuşlardır (Cohen 1990). Kısıt yayılımı ve çeşitli arama algoritmaları bu dillere dahil edilmiştir (Apt 2003).

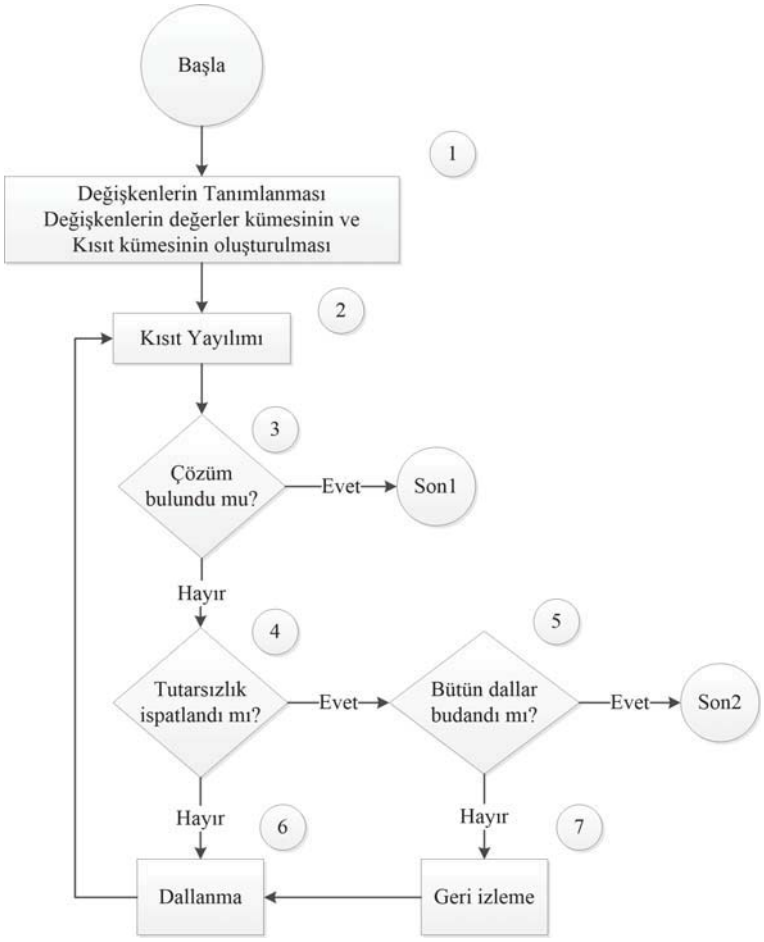
1990'larda araştırmacılar kısıt programlamayı yöneylem araştırması alanında uygulamaya başlamışlar ve günümüze kadar bu uygulamalar ilerleyerek devam etmiştir. İlerleme genelde yeni kısıtların tanımlanması ve yeni arama algoritmalarının geliştirilmesi ile gerçekleşmiştir. Ayrıca ilerleyen çalışmalarda yapay zeka teknikleri, yöneylem araştırması ve matematiksel mantık teknikleri birleştirilerek kısıt programlama hibrid bir alana dönüştürülmüştür (Apt 2003). Son yirmi yılda kısıt programlama tekniklerinin matematiksel programlamaya alternatif olarak kombinatoryal problemlere uygulaması hızla artmış ve başarılı çalışmalar gerçekleştirilmiştir (Van Hentenryck 1999, Khayat ve ark. 2006).

Kısıt programlama (KP), kombinatoryal optimizasyon problemlerinin çözümü için geliştirilmiştir. Bu problemler bir veya birden fazla *kısıt sağlama problemi (KSP)* (constraint satisfaction problem) şeklinde tanımlanarak çözülür. Bir KSP, *değişkenler* kümesi, bu değişkenlerin alabileceği olası değerleri barındıran *değerler* (domains) kümesi ve *kısıtlar* kümesinden oluşur (Baptiste ve ark. 2001). Kısıtlar farklı operatörlerden oluşabilir: $=, <, >, \leq, \geq, \neq$, altküme, üstküme, birleşme, üyelik, $\vee$ (VEYA), $\bullet$ (VE), $\Rightarrow$ (gerektirmek), $\Leftrightarrow$ (IFF). Bu kısıtlara ek olarak, araştırmacılar özel amaçlı kısıtlar geliştirmişlerdir (*alldifferent, span, noOverlap...vb.*) (Kanet ve ark. 2004). KSP'nin amacı, değişkenlerin ilgili değerlere atanmasıyla bütün kısıtların sağlandığı bir çözüme ulaşmaktır. KP'nin KSP'den farkı, matematiksel programlamada olduğu gibi bir amaç fonksiyonuna sahip olmasıdır. Bu sebeple, kısıtları sağlayan herhangi bir çözüm yerine, kısıtları sağlayan ve amaç fonksiyonunu en iyileyen çözüm aranmaktadır.

KP'de kısıtlar aktif olarak hesaplama süresini azaltmak için kullanılırlar. Kısıtlar, çözümün geçerliliğini test etmekle birlikte, değerler kümesinden uygun olmayan değerlerin çıkarılmasında, yeni kısıtların çıkarımının yapılmasında ve tutarsızlıkların test edilmesinde kullanılırlar. Kısıtların aktif şekilde kesin çıkarımlarda kullanılmasına *kısıt yayılımı* (constraint propagation) denir. Ayrıca, bu özel çıkarımlarla değerler kümesinin azaltılmasında *değer azaltma* (domain reduction) denir (Baptiste ve ark. 2001).

Bir KSP çözümü için gereken adımlar Şekil 4.1'de görülmektedir (Kanet ve ark. 2004). KSP algoritması değişkenlerin, değer kümelerinin ve kısıtların tanımlanması ile başlar (Blok 1). Kısıtlar, kısıtlar kümesinde depolanır. Blok 2'de değişkenlerin değerleri, her kısıt için sistematik olarak değişkenlerin değerlerini azaltan mantık-tabanlı *filtreleme algoritmaları* kullanılarak azaltılır. Değişkenin değer kümesi azaltıldığında bu değişkeni barındıran kısıt, ilgili filtreleme algoritmasının uygulanması için aktif hale gelir. Bu sistematik adımlar, kısıt yayılımı ve değer azaltma süreçleridir. Bu süreçlerden sonra Blok 3'teki gibi iki durum ortaya çıkar; bir çözümün bulunması veya bulunamaması. Eğer bir çözüm bulunursa algoritma sonlandırılır (Son1). Eğer bütün sonuçlara ihtiyaç varsa temel süreç tekrarlanır. Eğer hiçbir çözüm bulunamazsa, tutarsızlık (en az bir değişkenin değerler kümesinin tamamen boşalması) meydana gelmiştir (Blok 4). Eğer tutarsızlık

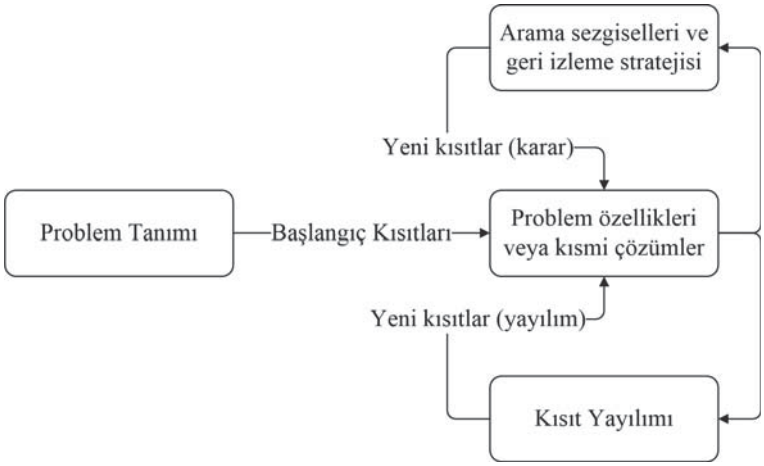
ispat edilemezse, dallanma için bazı arama tekniklerinin kullanılmasıyla aramaya devam edilir (Blok 6). Dallanma ile bir dal seçilir ve filtreleme algoritmaları kullanılarak bütün kısıtların yayılımı gerçekleştirilir (Blok 2). Eğer Blok 4'te tutarsızlık ispatlanırsa, Blok 5'teki arama ağacı bütün alt problemlerin budanıp (fathoming) budanmadığını kontrol eder. Eğer bütün dallar budandıysa, tutarsızlık ispat edilmiştir. Değilse, algoritma geri izleme ile (Blok 7) bir önceki aşamaya gider ve farklı bir alt problemden dallanır (Blok 6) (Kanet ve ark. 2004).



Şekil 4.1. KSP genel algoritması (Kanet ve ark. 2004)

Genel KSP problemi NP-tam olmasına rağmen (Garey ve Johnson 1979), kısıt yayılımı için aynı durum geçerli değildir. Pratikte kısıt yayılımı bütün tutarsızlıkları tespit edemez ve kullanıcı bir arama yöntemi kullanarak mevcut KSP atamalarının bir çözümü olup olmadığını araştırır. Genelde *ağaç arama algoritması* (search tree algorithm) ile arama gerçekleştirilir. Ağaç arama algoritmasının iki bileşeni ileri ve geri yönlü

yapılan ilerlemedir. İleri yönlü ilerlemede aramada alınan kararlar uygulanır, geri yönlü ilerlemede *geri izleme* gibi bir tutarsızlık tespit edildiğinde algoritmanın nasıl davranacağı belirlenir. Aramada hangi aşamada hangi kararların alınabileceği tanımına *arama sezgiseli* (search heuristic) denir. Arama sırasında tutarsızlık meydana gelmesi, mevcut değerlerin değişkenlere atanması ile kısıtlar değerlendirildiğinde bir çözüm elde edilememesidir. En sık kullanılan geri izleme stratejisi *derinlik öncelikli* (depth-first) geri izlemedir, yani son alınan karardan vazgeçilir ve başka bir atama gerçekleştirilir. Literatürde birçok karmaşık geri izleme algoritması mevcuttur (Prosser 1993, Gomes 2004).



Şekil 4.2. Kısıt programlama sistemi davranışı (Baptiste ve ark. 2001)

Kısıt programlama sisteminin davranışı Şekil 4.2'deki gibidir. Şekilde de görüldüğü gibi problem tanımı, kısıt yayılımı, arama sezgiseli ve geri izleme stratejisi açık bir şekilde birbirlerinden ayrılmışlardır. Bu ayrım, kısıt programlamanın çözüm aşamalarının matematiksel programlamadan oldukça farklı olduğunu da göstermektedir. Kısıt programlama, çoğu kez matematiksel programlamanın bir uzantısı gibi kabul edilse de bu doğru değildir. Aslında iki yöntem de kombinatoryal problemlerin çözümüne farklı açıdan yaklaşırlar (Apt 2003). Araştırmacılar, bu farklı açıların güçlü yönlerini bir arada kullanmak için matematiksel programlamanın bir alt kümesi tamsayılı programlamayı ve kısıt programlamayı birleştiren çalışmalar gerçekleştirmişlerdir (Darby-Dowman ve ark. 1997, Jain ve Grossmann 2001, Hooker 2002, Edis ve Ozkarahan 2011). Bu çalışmaların amacı, başarılı iki yöntemin güçlü yönlerini birleştirmektir. Kısıt programlamanın gücü, modelleme esnekliği ve değer azaltma tekniklerinden kaynaklanmaktadır. Tamsayılı

programlamada ise gevşetme, kesme düzlemi ve dual yapılarla ilgili özel arama teknikleri mevcuttur. Dolayısıyla, birbirinin eksiklerini tamamlayan iki yöntemin bir arada kullanılmasıyla başarılı sonuçlar elde edilmiştir (Edis ve Ozkarahan 2011).

4.2.1. KP ile çizelgeleme problemlerinin modellenmesi

KP, kombinatoryal problemlerin modellenmesini kolaylaştıran operatörlere ve farklı değişken tiplerine sahiptir. Bu yapılar çizelgeleme problemlerinin modellenmesini kolaylaştırdığı gibi çözümünü de kolaylaştırır (Kanet ve ark. 2004).

4.2.2. Değişken indeksleme

KP, bir değişkenin bir diğer değişkenin indisi olarak kullanılmasına izin verir. Bu özellik sayesinde çizelgeleme problemi modellemesinde gerekli değişken sayısı azaltılmış olur. Bu duruma örnek olarak sıra bağımlı hazırlık süreleri olan tek makine yığın sıralama problemi verilebilir (Jordan ve Drexl 1995). Bu problemin tamsayılı programlama (TP) modelinde *i* işinden sonra *j* işinin gelmesi *y[i,j]* değişkeni ile temsil edilmiştir. Bu durumda eğer problemde *n* adet iş olursa olası iş sıralarını temsil eden *n(n-1)* adet 0-1 değeri alan değişken tanımlanır. Eğer *i* işinden hemen sonra *j* işinin gelmesiyle oluşan maliyet *maliyet[i,j]* ile temsil edilirse, toplam maliyet hesabı için bütün *i*, *j* ve $i \neq j$ için *maliyet[i,j]*y[i,j]* toplamı hesaplanır. Diğer taraftan, problemin KP ile modellenmesinde işlerin sırasında *k*. pozisyona atanan işi gösteren *iş[k]* değişkeni tanımlanabilir. Bu değişken ile toplam maliyet hesabı yapılırsa bütün $k > 1$ değerleri için *maliyet[iş[k-1],iş[k]]* değerlerinin toplamı hesaplanır. Bu sayede problemdeki değişken sayısı *n(n-1)*'den *n* adete azaltılır. Bir diğer örnek olarak, *m* makine ve *n* iş için *m*n* adet değişken tanımlamak yerine *n* adet iş için değişken tanımlanarak bu değişkenlerin atandıkları makinenin değerini temsil etmeleri sağlanabilir. Bu şekilde değişken sayısı azaltılmış ve problem sadeleştirilmiş olur (Darby-Dowman ve ark. 1997). Sonuç olarak, değişkenlerin alabileceği değerler üzerinde daha özellikli tanımlamalar yapılarak TP'deki 0-1 değişkenlerden daha etkin değişkenler KP'de oluşturulabilir.

4.2.3. Kısıtlar

KP, operatörlerin ve mantıksal ifadelerin kısıtlarda kullanılmasına izin vererek birçok zorlu kısıtın kolaylıkta temsil edilebilmesini sağlar (Williams ve Wilson 1998).

İkili değişkenlerle eşitsizlikler: Bir çizelgeleme probleminde *MakineA* ve *MakineB* değişkenleri 0-1 tanımlı olsun. Eğer makineye bir iş atanmışsa 1, değilse 0 değerini alsın. Makine ortamında yalnızca bir makineye iş atanabilir olsun ve her iki makineye de işin atanması engellensin. Bu durum KP'de direk olarak;

$$MakineA \neq MakineB$$

ile gösterilirken TP'de;

$$MakineA + MakineB = 1$$

şeklinde gösterilir. Her iki ifade de doğrudur, ancak KP'deki kısıt daha doğrudan bir ifadedir.

Tamsayı değişkenlerle eşitsizlikler: Bir çizelgeleme probleminde *MakineA* ve *MakineB*, her makineye atanan işi gösteren tamsayı değişkenler olsun ve bir iş birden fazla makineye atanamasın. Bu durum KP'de basitçe;

$$MakineA \neq MakineB$$

şeklinde gösterilir. Ancak TP'de $\neq$ ifadesi $(<, >)$ ile ve XOR (exclusive or) ile gösterilmelidir.

$$(MakineA > MakineB)XOR(MakineB > MakineA)$$

Hatta TP $\leq$ ve $\geq$ kısıtları ile sınırlıdır. Bu sebeple, eşitsizliği sağlayabilmek için küçük bir ε değeri eklenir ve çıkartılır.

$$(MakineA \geq MakineB + \varepsilon)XOR(MakineB \geq MakineA + \varepsilon)$$

Son olarak, TP'de XOR ifadesi yukarıdaki eşitsizlik gibi gösterilemez, bunu temsilen her kısıt için bir δ gösterge değişkeni tanımlanmalıdır. Ayrıca, sadece bir kısıtın tutmasını sağlamak için büyük bir *M* çarpanı kullanılmalıdır. Sonuç olarak, TP'deki kısıtlar şu şekilde olacaktır:

$$MakineA - MakineB - \varepsilon + \delta * M \geq 0$$
$$MakineB - MakineA - \varepsilon + (1 - \delta) * M \geq 0$$

Bu örnekte $\delta = 1$; $MakineB > MakineA$'yı temsil ederken $\delta = 0$; $MakineA > MakineB$ demektir. İkili değişkenler, *M* çarpanı ve gösterge değişkenleri TP'de kısıtların temsilinde oldukça sık kullanılan ifadelerdir.

Mantıksal Kısıtlar: Çizelgeleme problemlerinde mantıksal ifadelere sıkça rastlanmaktadır; sıralama problemlerindeki öncelik kısıtları bu ifadelere örnek olarak verilebilir. Diyelim ki *A* işi başlamadan *B* işi tamamlanmalıdır; veya *B* işi başlamadan *A* işi tamamlanmalıdır, ancak iki durum aynı anda gerçekleşemez. Bu durum yine XOR durumuna bir örnektir ve KP'de gösterimi;

$$(A.start > B.end)XOR(B.start > A.end)$$

şeklindedir. TP'de ise bu kısıt;

$$A.start - B.end - \varepsilon + \delta * M \geq 0$$
$$B.start - A.end - \varepsilon + (1 - \delta) * M \geq 0$$

şeklindedir. Gösterge değişkeni $\delta = 1$; *A* işinin önce olduğunu gösterirken $\delta = 0$; *B* işinin önce olduğunu gösterir. KP ile XOR ifadesi doğrudan ve doğal bir şekilde gösterilebilmektedir (Kanet ve ark. 2004).

Evrensel Kısıtlar: Evrensel kısıtlar hemen hemen bütün değişkenlere uygulanabilir. Örnek olarak, işlerin *m* adet makineye atandığı bir problem ele alınsın. Bu problemde her işin farklı makineye atanması istensin. KP'de *alldifferent* kısıtı ile bu şart sağlanır:

alldifferent(Makine)

Bu kısıt ile aynı işin iki farklı makineye atanması engellenir. TP'de ise *alldifferent* kısıtı yerine her makine çifti için eşitsizlik yazılması gerekir. Bu durumda *m* makine için *m*(m-1)* kısıt ve *m*(m-1)/2* adet gösterge değişkeni tanımlanmalıdır. KP'de bunun gibi birçok özel kısıt tanımları mevcuttur. Örnek olarak, yine çizelgeleme probleminde bir makinenin birden fazla iş işlemesi engellenmek istenirse *distribute(Makine)* kısıtı kullanılır. Bu kısıt ile her makinedeki işin sayısı bir ile sınırlanır. TP'de ise aynı durum için, gösterge değişkenlerinin tanımlanması, her makine için makineye atanan işlerinin olup olmadığını gösteren kısıtların ve gösterge değişkenlerin toplamını barındıran bir kısıtın oluşturulması gerekir.

Benzer şekilde birçok farklı özel amaçlı kısıt, KP'de çizelgeleme problemleri için oluşturulmuştur, ve bu kısıtlar farklı programlama dillerinde kullanılabilir hale getirilmiştir. Tez çalışması kapsamında KP'nin bu avantajlarından yararlanmak için ele alınan çizelgeleme problemi Bölüm 5'de OPL dili (Van Hentenryck 1999) kullanılarak modellenmiştir. Devamında kullanılan mantık-tabanlı Benders ayrıştırma yöntemi sonraki bölümde detaylandırılmıştır.

4.3. Mantık-Tabanlı Benders Ayrıştırma Yöntemi

Mantık-tabanlı Benders ayrıştırma (MTBA) metodu, genelleştirilmiş Benders ayrıştırma yönteminin tamsayılı modellere uygulanmasıyla geliştirilmiştir (Hooker ve Ottosson 2003). Geliştirildiği zamandan günümüze kadar pek çok araştırmacı tarafından çizelgeleme problemlerine ve atama problemlerine uygulanmıştır (Harjunkoski ve Grossmann 2002, Hooker ve Ottosson 2003, Hooker 2005, Hooker 2007, Coban ve Hooker 2010, Heinz ve Beck 2012). Yöntemin anlatılmasında Hooker'ın (2011) çalışmasından yararlanılmıştır. Klasik Benders ayrıştırma, doğrusal programlamanın zayıf dualite (weak duality) özelliğine dayanır. Verilen bir doğrusal modelin,

$$\begin{aligned} &Min\, cx \\ &st.\, Ax \geq a \end{aligned} \tag{4.1}$$

dual problemi;

$$\begin{aligned} &Max\, ua \\ &st.\, uA = c \\ &u \geq 0 \end{aligned} \tag{4.2}$$

şeklinde olsun. *u* satır vektörü dual değişkenleri temsil etsin. Burada zayıf dualiteyi göstermek kolaydır. $\bar{x}$ ve $\bar{u}$ yukarıdaki sistemlerin uygun çözümleri olursa $c\bar{x} \geq \bar{u}a$ olacaktır. Bunun sebebi, $c = \bar{u}A$ eşitliği ile $\bar{u} \geq 0$ ve $A\bar{x} \geq a$ olduğundan $c\bar{x} = \bar{u}A\bar{x} \geq \bar{u}a$ olmasıdır. Benders ayrıştırması aşağıdaki formdaki problemlere uygulanabilir:

$$\begin{aligned} &Min\, cx + f(y) \\ &st.\, Ax + g(y) \geq a \\ &x \geq 0,\, y \in Y \subseteq R^n \end{aligned} \tag{4.3}$$

Bu sistemde $g(y)$, $g_i(y)$ fonksiyonlarının vektörüdür. Ayrıştırmada temel strateji, her farklı $y \in Y$ değeri için optimal *x* değerini bulmaktır. Metot, *y* değerleri $\bar{y}$'ye sabitlenerek elde edilen aşağıdaki doğrusal alt-problem (4.4)'ü çözer.

$$\begin{aligned} &Min\, cx \\ &st.\, Ax \geq a - g(\bar{y}) \end{aligned} \tag{4.4}$$

Diyelim ki (4.4) sistemi uygun bir model olsun. eğer $z(\bar{y})$ değeri (4.4) sisteminin optimal değeri ise $z(\bar{y}) + f(\bar{y})$ değeri orijinal (4.3) sisteminin optimal değeri olacaktır. Ayrıca (4.4) sisteminin optimal çözümü, orijinal sistemin optimal çözümde sağlaması gereken ve *Benders kesimi* olarak adlandırılan kısıtları sağlamaktadır. Benders kesimi, (4.4) sisteminin dualinin optimal $\bar{u}$ çözümünden elde edilir. (4.4) sisteminin duali aşağıdaki gibidir:

$$
\begin{aligned}
&Max\, u(a - g(\bar{y})) \\
&st.\, uA = c \\
&u \geq 0
\end{aligned} \tag{4.5}
$$

Benders ayrıştırmanın temeli şu şekildedir: Herhangi bir $\bar{y}$ değeri için $\bar{u}$ çözümü (4.5) dual problemin uygun çözümü olacaktır. Zayıf dualiteden herhangi bir *y* için $z \geq \bar{u}(a - g(y))$ olacaktır. Bu da demek olur ki, zayıf dualite ile, herhangi bir *y* için $\bar{u}(a - g(y)) + f(y)$ değeri orijinal (4.3) probleminin optimal z değeri için bir alt sınır olacaktır. Aşağıdaki eşitsizlik (4.6) $\bar{y}$ değerleri için üretilen Benders kesimidir.

$$
z \geq \bar{u}(a - g(y)) + f(y) \tag{4.6}
$$

Algoritmanın her *p* iterasyonunda $\bar{y^p}$ değerleri için $\bar{u^p}(a - g(y)) + f(y)$ Benders kesimi üretilir ve bu kesim ana-probleme eklenir.

$$
\begin{aligned}
&Min\, z \\
&st.\, z \geq \bar{u^p}(a - g(y)) + f(y), q \in Q\,(a) \\
&\bar{u^p}(a - g(y)) + f(y) \leq 0, q \in \bar{Q}\,(b) \\
&y \in Y
\end{aligned} \tag{4.7}
$$

(4.7) sistemi iki farklı kesim barındırır. Bunlardan ilki (a) kesimleri, alt-problem uygun olduğunda üretilen kesimlerdir ve bu durum $q \in Q \subset 1, ..., p$ iterasyonlarında gerçekleşir. İkinci (b) kesimleri, alt-problem uygun olmadığında üretilir ve bu durum $q \in \bar{Q}$ iterasyonlarında görülür. Uygun çözüm olmadığı durumda dual problem sınırsızdır ve $\bar{u^q}$ çözümü ışın olur, bu durumda $\alpha\bar{u^q}$ değeri dual problem (4.8)'in herhangi bir $\alpha \geq 0$ için uygun çözümü olur.

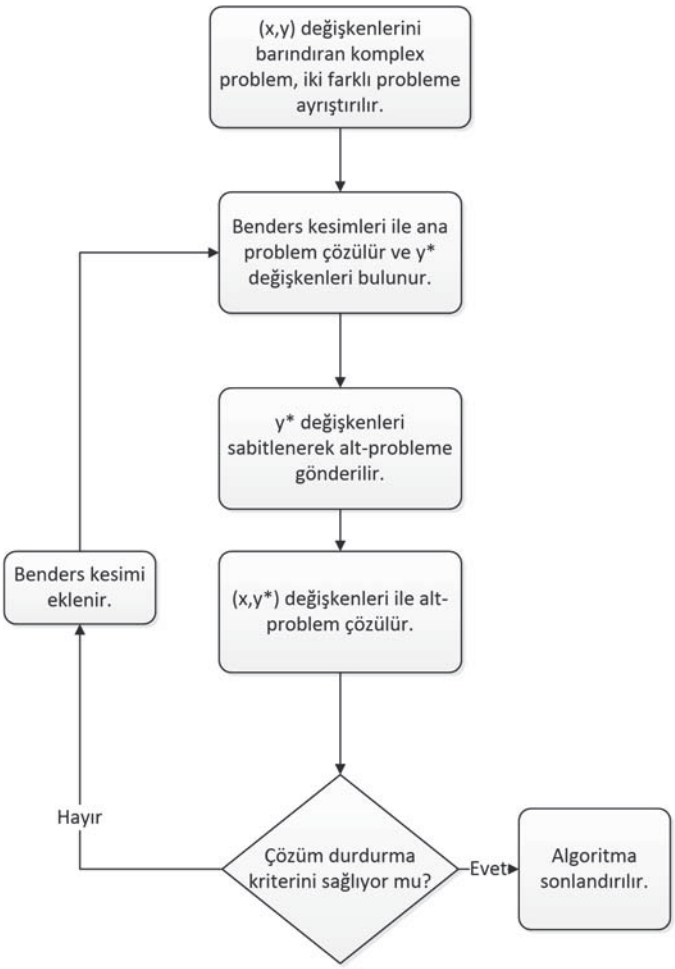
$$
\begin{aligned}
&Max\ \bar{u^q}(a - g(\bar{y})) \\
&st.\ \bar{u^q}A \leq 0 \\
&\bar{u^q} \geq 0
\end{aligned} \tag{4.8}
$$

MTBA yöntemi şu şekilde özetlenebilir: *(x,y)* değişkenlerinden oluşan bir karmaşık problem iki farklı probleme ayrıştırılır; *ana-problem (master problem)* sadece zor y değişkenlerini barındırır, ve *alt-problem (sub-problem) x* ve *y* değişkenlerini barındırır. Ana-problem zor değişkenlerin değerlerini bulur ve bu bilgiyi y^* değerlerini sabitleyerek uygun bir çözüm bulması için alt-probleme iletir. Alt-problem sabitlenmiş y^* değerleri için uygun x değerleri bulmaya çalışır. Alt-problem çözümü uygun değilse Benders kesimleri ana probleme eklenir ve mevcut y^* çözümü çözüm uzayından çıkarılır. İteratif olarak devam eden bilgi alışverişi ve kesimlerin eklenmesiyle, optimal sonuç bulunana kadar veya ana-problemin çözüm uzayının tamamı taranana kadar MTBA algoritması devam eder. Algoritma durdurma kriterleri problemlerin özelliklerine göre belirlenir. MTBA algoritması genel olarak Şekil 4.3'deki gibidir.

Benders kesimlerinin üretilmesi Benders ayrıştırma yönteminin temelidir (Chu ve Xia 2004). Klasik Benders ayrıştırmada, kesimler doğrusal programlamadaki zayıf dualite özelliği temel alınarak oluşturulur. Ancak tamsayılı programlamanın kesikli yapısından dolayı bu özellik kullanılamaz ve geçerli bir kesim üretmek güçleşir. Chu ve Xia (2004) çalışmalarında geçerli bir Benders kesiminin sahip olması gereken iki özelliği aşağıdaki gibi tanımlamışlardır:

- *Özellik 1:* Eğer mevcut ana-problem çözümü global uygun değilse, kesim bu çözümü, çözüm uzayından çıkarmalıdır.
- *Özellik 2:* Kesim başka herhangi bir global uygun çözümü dışarıda bırakmamalıdır.

Yazarlar *Özellik 1* ile eğer değişkenlerin sonlu bir tanım uzayı varsa sonlu bir yakınsamayı ve *Özellik 2* ile diğer çözümler dışarıda bırakılmadığından optimal çözümü garantilediğini göstermişlerdir. MTBA algoritmasında üretilen sonuçların optimalliğinden bahsedebilmek için bu özellikler gösterilmelidir.



Şekil 4.3. MTBA yöntemi genel akışı

Çalışmada ele alınan çizelgeleme problemi için geliştirilen MTBA algoritmaları sonraki bölümde diğer yöntemlerle karşılaştırmalı olarak anlatılmıştır.

5. GELİŞTİRİLEN ÇÖZÜM YÖNTEMLERİ

5.1. Uygulamada Kullanılanılacak Problemin Tanımı

Bu çalışmada ana otomotiv sanayi için çeşitli otomobil parçaları üreten Beyçelik Gestamp kalıp fabrikası incelenmiştir. Pres çizelgeleme problemi, makine elverişlilik ve kaynak kısıtları altında ardışık makinelerde işlenmesi gereken farklı sayıda operasyona sahip işlerin bağımsız paralel makine ortamında eş zamanlı olarak çizelgelenmesi ($R_m|M_i; res; chains|C_{max}$) problemi şeklinde sınıflandırılabilir. Paralel makine problemleri literatürde çokça çalışılmasına rağmen, bu tanımda bir probleme ve dolayısıyla çözüm yöntemine literatürde rastlanmamıştır.

Ele alınan pres hattı 13 adet ardışık presten oluşmaktadır. Her parça son ürün haline gelmek için farklı sayıda operasyona ihtiyaç duyar ve her operasyonun gerçekleşmesi için üst ve alt olmak üzere özel bir kalıp seti gereklidir. Üretim süreci kalıpların preslere yüklenmesi ile başlar. Hammadde olan şekillenmemiş sac, pres hattında ilerleyerek kalıplar tarafından şekillendirilir. Bir parçanın bütün operasyonları tamamlandığında son ürün elde edilir. Toplam operasyon sayısı toplam pres sayısını aşmayacak şekilde, birden fazla farklı ürün aynı anda pres hattına yüklenebilir. Örnek olarak, 5, 5, ve 3 operasyondan oluşan 3 farklı ürün aşağıdaki şekildeki gibi eş zamanlı olarak pres hattında üretilebilir.

Presler	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13
Operasyonlar	A.1	A.2	A.3	A.4	A.5	B.1	B.2	B.3	C.1	C.2	C.3	C.4	C.5
Bitmiş ürünler	Ürün A					Ürün B			Ürün C				

Şekil 5.1. Pres hattında 3 farklı ürün ataması

Pres hattı çizelgelemede birçok zorluk vardır:

- Planlama dönemi süresinde müşteri taleplerinin değişmesi.
- Preslerin arızalanması.
- Kalıp değişim sürelerinin uzunluğu.

Pres hattı çizelgeleme pek çok farklı zorluğu da beraberinde getirir. Bunların en önemlisi müşteri taleplerindeki dalgalanmadır. Firma haftalık taleplerle çalışsa da müşteriler

taleplerini hafta içinde acil isteklerle değiştirme eğilimindedirler. Bu acil talepler bütün çizelgeyi değiştirmesine rağmen, firma bu dalgalanmalara elinden geldiğince cevap vermeye çalışmaktadır.

İkinci zorluk, pres arızalarıdır. Bir pres arızalandığında buradaki operasyon arıza giderilene kadar veya buradaki kalıplar uygun bir prese aktarılana kadar ertelenir. Arızanın gerçekleştiği presin pozisyonu, ilgili işin diğer operasyonlarını da etkiler. Eğer arıza son operasyon hariç diğer operasyonların olduğu preslerde gerçekleşmiş ise, işin diğer operasyonları da arıza giderilene kadar ertelenir. Eğer arıza son operasyonun bulunduğu preste gerçekleşmiş ise, ilgili kalıplar eğer varsa uygun bir başka prese aktarılır, son prese kadar operasyonlar yine devam eder ve yarı-ürün yeni prese taşınarak, son ürün burada elde edilir. Örnek olarak, Şekil 5.1'de eğer arıza P3'de gerçekleşirse A işinin tüm operasyonları P3 tamir edilene kadar ertelenir, bu da dört presin boş beklemesine sebep olur. Eğer arıza P5'te gerçekleşirse, buradaki kalıplar eğer varsa uygun bir prese yüklenir, ilk dört operasyonu tamamlanan yarı-ürün yeni prese taşınarak burada son ürün elde edilir. Ancak presler ağırlık, boyut ve tonaj bakımından birbirlerinden farklıdır, dolayısıyla uygun bir pres bulmak kolay değildir. Örneğin, eğer bir operasyon yüksek tonaj gerektiriyorsa ilgili kalıplar düşük tonajlı bir prese yüklenemez. Diğer taraftan, eğer bir operasyon düşük tonaj gerektiriyorsa bu kalıplar yüksek tonajlı preselere yüklenebilir. Makine özellikli kısıtlamalardan doğan değişkenlikten dolayı her iş için alternatif makine kümeleri tanımlanmıştır. İşler yalnızca bu makine kümelerindeki makinelerde çizelgelenebilir ve bu kısıtlama etkin çizelgelerin bulunmasını zorlaştırmaktadır.

Üçüncü zorluk ise etkin kalıp değiştirmelerin belirlenmesidir. Bir kalıbın değişimi ortalama olarak 30 dakika sürmektedir ve bu durum üretim zamanını azaltmaktadır. Kalıp değiştirmeler sırasında ilgili preslerde üretim durur ve bu durum firmaya kayıplar yaşatır. Firma kalıp değişimlerinin yemek molasında ve vardiya değişimlerinde yapılmasına izin vermektedir. Acil durumlar haricinde diğer zamanlarda kalıp değişimi yapılamaz. Bu varsayım, dört saatlik üretim periyotlarının tanımlanmasına sebep olur. Bu durumda, eğer bir üretim başlarsa bu üretim en az dört saat sürer. Eğer bir iş dört saatte tamamlanamazsa

sonraki dört saat boyunca da üretilmeye devam eder. Bu durumda ürün talebi dört saatlik üretimden az bile olsa dört saat boyunca üretilir ve fazla üretim stok alanlarında depolanır.

Bu problem için başlangıç olarak karışık tamsayılı programlama ile bir matematiksel model (KTP1) geliştirilmiş ve sonraki bölümde detaylandırılmıştır.

5.2. KTP1 Modeli

Beyçelik firması, işlerin bölünmediğini varsaymakta ve ilgili kalıpların bir defa yüklenmesiyle işlerin talepleri kadar veya daha fazlası üretilmektedir. Gerçek sistemi tam anlamıyla temsil edebilmek ve modelleyebilmek için KTP1 modelinde işlerin bölünemediği kabul edilmiş ve aşağıdaki kısıtlarla model geliştirilmiştir:

- Her işin farklı sayıda operasyonu vardır ve bu operasyonlar ardışık makinelere atanmalıdır.
- Her iş bir periyotta sadece bir makinede başlayabilir.
- Her makine bir periyotta sadece bir tip iş işleyebilir.
- Toplam üretim miktarı talep kadar veya talepten büyük olmalıdır.
- Kalıp değişimleri işlerin başlangıcında ve bitişinde gerçekleşir.
- Periyot uzunluğu dört saattir, dolayısıyla eğer *i* işi *m* makinesinde başladıysa en az dört saat boyunca işlenir.

Problemler için gerekli olan periyot adedi parametresi K_{max} problemin zaman sınırını temsil eder ve bir periyot adedi dört saatlik periyot uzunluğunu temsil eder. Bu parametrenin değeri büyük bir sayı olarak belirlenebilir. Ancak çözüm uzayını küçültmek ve modelin çözüm hızını arttırmak amacıyla deneylerde herhangi bir problem için ilk uygun çözüm noktasının elde edildiği K_{max} parametresi dikkate alınmıştır. Örnek olarak, eğer bir problem, K_{max} değeri 20 olduğunda çözümsüz ise ve bu değer 21 yapıldığında uygun çözüm noktaları bulunabiliyorsa K_{max} değeri 21 olarak belirlenmiştir (bu problem için K_{max} değeri 21'den büyük herhangi bir sayı alınabilir).

KTP1 için indisler, parametreler, kümeler ve değişkenler aşağıdaki gibi tanımlanmıştır:

İndisler(kümeler aşağıda tanımlanmıştır):

i: Çizelgelenecek işler, $i \in I$.

j_i: İşlerin operasyon adetleri, $j \in O_i$.

k: Periyotlar, $k \in K$.

m: Makineler, $m \in M$.

Parametreler:

A_{im} : Eğer *i* işi *m* makinesine atanabilirse A_{im}= 1 ve aksi halde sıfır.

J_i : *i* işinin toplam süresi: $\sum_{j \in O_i} P_{ij} + (a_{ik} - 1) * P_{ij_{max}}$

U: Periot uzunluğu.

R : Kalıp değiştirme süresi.

D_i : *i* işinin talebi.

P_{ij}: *i* işinin *j* operasyonu için işlem süresi.

L_1: Büyük sayı. Model boyutunu azaltmak için *maks(Talep)*+1 olarak belirlenmiştir.

L_2: Büyük sayı. Model boyutunu azaltmak için *maks(Makine sayısı)*+1 alınmıştır.

L_3: İşlerin bölünmesine izin veren veya bölünmesini engelleyen pozitif bir sayı.

Kümeler:

I: İşler kümesi. *I*=$\{1,2,...,I_{max}\}$.

K: Periyotlar kümesi. *K*=$\{1,2,...,K_{max}\}$.

M: Makineler kümesi. *M*=$\{1,2,...,M_{max}\}$.

O_i: *i* işinin operasyonları kümesi. O_i=$\{1,2,...,O_{i_{max}}\}$.

Set_1: $\{ (i,k,m) \mid i \in I, k \in K, m \in M, A_{im} = 1\}$.

Karar Değişkenleri:

$$x_{ikm} = \begin{cases} 1, \text{Eğer } i\text{.iş } k\text{.periyotta } m\text{.makinede işlenirse;} \\ 0, \text{Aksi halde.} \end{cases}$$

a_{ik}: *k.* periyotta üretilen *i* ürünü miktarı. $i \in I, k \in K$.

c_{ikm}: *i* işinin *k.* periyotta *m.* makinedeki kalıp değişim süresi. $i \in I, k \in K, m \in M$.

C_{max}: Toplam çizelge uzunluğu.

Bu varsayımlara ve kısıtlara göre KTP1 modeli:

$$Min\, z = C_{max} + L_3 * \sum_{(i,k,m) \in Set_1} c_{ikm} \tag{5.1}$$

$$(k-1) * U * x_{ikm} + \frac{J_i}{3600} + \sum_{\substack{(t,l,n) \in Set_1 | 0 \leq l \leq k+1, \\ n \leq m, n + O_{t_{max}} - 1 \geq m}} c_{tln} \leq C_{max}, \forall (i,k,m) \in Set_1 \tag{5.2}$$

$$\sum_{j \in O_i} P_{ij} + (a_{ik} - 1) * P_{ij_{max}} \leq 3600 * U, \quad \forall i \in I, k \in K \tag{5.3}$$

$$\sum_{i \in I | (i,k,m) \in Set_1} x_{ikm} \leq 1, \forall k \in K, m \in M \tag{5.4}$$

$$\sum_{m \in M | (i,k,m) \in Set_1} x_{ikm} \leq 1, \forall i \in I, k \in K \tag{5.5}$$

$$a_{ik} \leq L_1 * \sum_{m \in M | (i,k,m) \in Set_1} x_{ikm}, \forall i \in I, k \in K \tag{5.6}$$

$$\sum_{k \in K} a_{ik} \geq D_i, \forall i \in I \tag{5.7}$$

$$x_{tkn} \leq 1 - x_{ikm}, \forall (i,k,m) \in Set_1, \quad (t,k,n) \in Set_1 \mid t \neq i, m \leq n \leq m + O_{i_{max}} - 1 \tag{5.8}$$

$$x_{ikm} * m + O_{i_{max}} - 1 - L_2 * (1 - x_{ikm}) \leq M_{max}, \quad \forall (i,k,m) \in Set_1 \tag{5.9}$$

$$R * (x_{ikm} - x_{ik+1m}) \leq c_{ikm}, \quad \forall (i,k,m) \in Set_1 \mid k+1 \leq K \tag{5.10}$$

$$R * (-x_{ikm} + x_{ik+1m}) \leq c_{ikm},$$
$$\forall (i,k,m) \in Set_1 \mid k+1 \leq K \tag{5.11}$$

$$a_{ik}, c_{ikm}, C_{max} \geq 0, \forall i \in I, k \in K, m \in M \tag{5.12}$$

$$x_{ikm} \in \{0,1\}, \forall (i,k,m) \in Set_1 \tag{5.13}$$

KTP1 amaç fonksiyonu maksimum tamamlanma zamanını (C_{max}) ve kalıp değişim sürelerini barındırmaktadır. İkinci kısımda toplam kalıp değişimi için harcanan süre hesaplanmakta ve L_3 değeri işlerin bölünmesi kontrolünde kullanılmaktadır. L_3 değeri sıfır yapılarak işlerin bölünmesine izin verilmekte; büyük bir sayı olarak belirlenerek işlerin bölünmesi engellenmektedir. Tamamlanma zamanı içinde kısıt (5.2)'de görüldüğü gibi üretim süreleri ve kalıp değişim süreleri vardır. Saniye olarak hesaplanan üretim süresi bölüm işlemiyle saate çevrilir. Her işin maksimum tamamlanma zamanı, önceki periyot adedinin periyot limitleri ile çarpılması, son periyottaki üretim miktarının bu değere eklenmesi ve hazırlık sürelerinin üretim süresine dahil edilmesiyle bulunur. Son periyottaki üretim süresi belirlenirken, J_i parametresinin hesaplanmasında olduğu gibi, operasyonların ardışık preslerde olduklarına dikkat edilmelidir. İlk parçanın tamamlanma süresi operasyon sürelerinin toplamı olurken geri kalan parçalar için farklı bir durum söz konusudur. İlk parça için işlenmemiş sac ilk operasyon için ilk prese yüklenir ve buradan ikinci operasyon için ikincisine ilerler, bu ilerlemeden sonra ikinci parça için işlenmemiş sac ilk operasyon için ilk prese yüklenir. Dolayısıyla, ilk ürün hariç diğer ürünlerin tamamlanma süreleri aslında en uzun operasyonun süresi kadardır. Ele alınan problemde işlerin operasyon süreleri birbirlerine eşittir ve bu durum $P_{i1} = P_{i2} = ... = P_{ij}$ şeklinde gösterilebilir. Ancak geliştirilen modeldeki P_{ij} parametresi, operasyon sürelerinin farklı olduğu durumlarda modele esneklik sağlamaktadır. Kısıt (5.3)'de $U = 4$ olarak alınmış ve periyotlar dört saat ile sınırlanmıştır. Kısıt (5.4) işlerin aynı periyotta farklı makinelerde üretilmesini sınırlar. Kısıt (5.5) bir periyotta bir makinenin sadece bir iş işlemesini sağlar. Kısıt (5.6) eğer *i* işi *k*. periyotta *m*. makineye atanmamışsa buradaki üretim adedinin sıfır olmasını sağlar. Kısıt (5.7) bir işin toplam üretim adedinin talebine eşit veya büyük

olmasını sağlar. Kısıt (5.8) eğer bir iş *m*. makineye *k*. periyotta atanmışsa bu işin operasyon sayısı kadar makineye o periyotta başka bir işin atanmasını engeller. Kısıt (5.9) preslere atanan işlerin operasyon adetlerini pres hattındaki toplam makine sayısı ile sınırlandırır. Kısıt (5.10) ve kısıt (5.11) işin pozisyonuna göre kalıp değişim sürelerini hesaplar. Kısıt (5.12) ve kısıt (5.13) sürekli ve tamsayılı değişkenleri tanımlar. Modeli basitleştirmek için üretim adetlerini temsil eden değişken a_{ik} sürekli olarak alınmıştır.

5.2.1. KTP1 amaç fonksiyonu değerlendirmesi

KTP1 modeli amaç fonksiyonunda C_{max} ve kalıp değişim sürelerini barındırmaktadır. Bu hesaplamaların model performansına etkisini inceleyebilmek için kalıp değişim süresi hesabında çarpan olarak kullanılan L_3 değeri ele alınmıştır. İlk olarak L_3 değeri sıfır olarak alınmış ve KTP1 amaç fonksiyonu ifade (5.14) ile gösterilen maksimum tamamlanma zamanının minimizasyonu haline dönüştürülmüştür. Bu durumda işlerin bölünmesine izin verilmektedir.

$$Min\, z = C_{max} \tag{5.14}$$

KTP1 performansını değerlendirmek için gerçek veriler dikkate alınmıştır. Firmaya gelen haftalık talepler genellikle 50 ve daha fazlası farklı işi barındırmaktadır. Bu deneyde kullanılan verilerde bir haftada 52 farklı iş, farklı talep miktarlarında sipariş verilmiştir. KTP1 performansını değerlendirmek için bu verilerden işlerin sayısı arttırılarak 10 farklı problem oluşturulmuştur. KTP1, ILOG CPLEX 12.4 ile modellenmiş ve 1800 saniye ile sınırlanmıştır. Deneyler 2.90 GHz i7 işlemcili 6 GB RAM bilgisayarda çalıştırılmıştır.

Bu amaç fonksiyonu ile gerçek veriler kullanılarak elde edilen sonuçlar Çizelge 5.1'deki gibidir. İlk üç sütunda problemlerin içerdiği iş sayısı ve çözümü için gerekli periyot adedi verilmiştir. Dördüncü kısımda süre limiti 1800 saniye kullanılarak elde edilen çizelge uzunlukları gösterilmiştir. Bulunan amaç fonksiyonu değeri, çözüm süresi ve %fark sütunları detaylandırılmıştır. Fark sütunu; tamsayılı programlama çözümünde kullanılan dal-kes (branch&cut) algoritmasında bulunan en iyi tamsayılı amaç fonksiyonu z* ve

henüz tamsayılı olarak dallanması yapılmamış en iyi çözüm z' değerleri kullanılarak ifade (5.15)'deki gibi hesaplanmaktadır. Optimal çözüme erişildiğinde %fark değeri sıfır olmaktadır. Bu sebeple %fark değeri optimal çözümle mevcut arasındaki hassasiyeti göstermektedir.

$$\%fark = \frac{|z' - z^*|}{e^{-10} + |z^*|} \tag{5.15}$$

Bu kısım incelendiğinde 20 iş ve üstü problemlerde optimal sonucun bulunamadığı ve %fark sütunu incelendiğinde Problem 7'den itibaren farkın %93,6'lardan büyük olduğu görülmektedir. Bu durum problem çözümünün %0-%93,6 arasında olduğunu ve elde edilen çözümün çok da hassas olmadığını gösterir. Çözüm uzayı yeni taranmaya başlamışken süre limitinden dolayı model sonlandırılmaktadır. Bu sebeple, KTP1 modeli sınırlarını test edebilmek için modeller Neos Sunucusunda (2014) 10 saat (36 000sn) limit ile çözülmüştür (http://www.neos-server.org/neos/solvers/go:scip/CPLEX.html, 2014). Bulunan sonuçlar Çizelge 5.1'nin son kısmında gösterilmektedir. Bu kısımda 10 saat sonunda bulunan en iyi C_{max} değeri ve son sütunda da bu değere ne zaman ulaşıldığı gösterilmektedir. Neos sunucusundan gelen verilerde ilk dört problem için belirtilen sürelerde optimal ve diğer problemler için daha başarılı sonuçlar bulunmuştur. Örnek olarak, Problem 7'de 1800 saniyede C_{max} değeri 376 saat olarak bulunmuşken, bu problem sunucuda 10 saat limit ile çözüldüğünde 6,7 (24 300sn) saat sonrasında çizelge uzunluğu 136 saat olarak bulunmuştur.

Çizelge 5.1. Gerçek verilerle KTP1 performansı (z=C_{max})

			Süre Limiti: 1800sn			Süre Limiti: 36000sn	
Problem	İşler	Periyotlar	C_{max}(s)	Çözüm Süresi(sn)	%Fark	C_{max}(s)	Çözüm Süresi(sn)
1	5	5	15,5	1,7	-	15,5	0,3
2	10	8	30,8	7,3	-	30,8	28,3
3	15	13	44,9	176	-	44,9	1381
4	20	22	91	1800	4,5	91	14255
5	25	25	100,5	1800	75	100,5	4098
6	30	27	109	1800	55,5	109	33180
7	35	32	376	1800	93,6	136	24300
8	40	40	472	1800	95	176	17820
9	45	49	628	1800	97,3	233	27120
10	52	54	719,5	1800	97,8	252	5100

Sonuç olarak, KTP1 problemleri sunucuda bile çözülse makul çizelgelere saatler sonra ulaşılmaktadır. Bu durum modeli hızlandırma arayışlarını da beraberinde getirmiştir. Alternatif olarak, L_3 değeri bir olarak alınmış ve amaç fonksiyonu C_{max} ve kalıp değişim sürelerinin minimizasyonu olacak şekilde güncellenmiştir. Bu güncellemenin ilk versiyon ile aynı olabilmesi için C_{max} değişkeni büyük bir *L* sayısı ile çarpılmıştır. Yenilenen amaç fonksiyonu ifade (5.16)'daki gibidir.

$$Min\, z = L * C_{max} + \sum_{(i,k,m)\in Set_1} c_{ikm} \tag{5.16}$$

Güncellenen amaç fonksiyonu (5.16) ile testler tekrarlanmıştır. Elde edilen sonuçlar Çizelge 5.2'de özetlenmiştir. Çözüm süresi 1800 saniye ile sınırlanmıştır. Çizelgede optimal sonuçta elde edilen C_{max} değerleri verilmiştir. Elde edilen C_{max} değerleri Çizelge 5.1 ile karşılaştırıldığında modelin verilen limitler içerisinde çözüm uzayını daha hızlı taradığı görülmüştür. İki amaç fonksiyonunun iterasyon sayıları ve çözüm süreleri oranlandığında, ilk amaç fonksiyonlu modelde bir iterasyon için harcanan sürenin dört kat daha fazla olduğu görülmüştür. Güncellenmiş amaç fonksiyonu kullanılarak aynı limitlerde daha iyi çizelgeler elde edilmiş, hatta 10 saatlik limitlerde elde edilen çözümlerden de daha iyi sonuçlar bulunmuştur. Bu sebeple bundan sonraki karşılaştırmalarda güncellenmiş amaç fonksiyonu ve deney sonuçları kullanılmıştır.

Çizelge 5.2. Gerçek verilerle KTP1 performansı($z = L * C_{max} + \sum_{(i,k,m)\in Set_1} c_{ikm}$)

Problem	İşler	Periyotlar	C_{max} (s)	Çözüm Süresi(sn)	%Fark
1	5	5	15,5	2,2	-
2	10	8	30,8	9,9	-
3	15	13	44,9	389,5	-
4	20	22	91	1800	5,7
5	25	25	100,5	1800	58,3
6	30	27	104,6	1800	56,5
7	35	32	127,6	1800	67,6
8	40	40	178	1800	85,6
9	45	49	238	1800	90,7
10	52	54	265,5	1800	93,3

5.2.2. Firma varsayımlarının test edilmesi

Firmanın dört saatlik aralıklarla kalıp değişimine izin vermesi, üretim periyotlarının dört saat olmasını gerektirmiştir. Bu periyot uzunluğunun uygunluğunu test etmek amacıyla KTP1 modeli farklı problem büyüklüklerinde ve farklı periyot uzunluklarında çalıştırılmıştır. Bu deneylerde firma varsayımlarının aksine işlerin bölünebilir olduğu varsayılmıştır (L_3 değeri sıfır kabul edilmiştir). Gerçek veriler incelenerek Çizelge 5.3'deki düzgün dağılımlar oluşturulmuş ve bu dağılımlarla beş adet örnek problem üretilmiştir. KTP1 kısa sürede en fazla 10 iş barındıran problemleri çözebildiğinden iş sayısı 10 olarak alınmıştır. Deney sonuçları Çizelge 5.4'de özetlenmiştir. Çizelge

Çizelge 5.3. Model parametreleri için düzgün dağılım değerleri

Parametreler	Düzgün dağılım: U(a,b)
Operasyon sayıları	U(1,6)
Operasyonların işlem süreleri (saniye)	U(5,20)
Alternatif makine sayısı	U(1,3)
Alternatif makine kümesi	U(1,13)

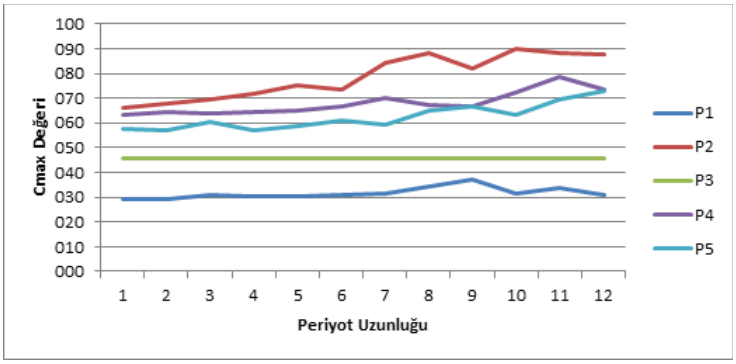
5.4'de ilk kısım 1-12 saat arasında değişen periyot uzunluklarını gösterir. 30 dakika olarak kabul edilen kalıp değişimlerinin periyot uzunları bitiminde yapıldığı varsayılır. İkinci kısım, 1800 saniye ile sınırlanan KTP1'in optimali bulmak için harcadığı süredir. Üçüncü kısım, çizelgenin maksimum tamamlanma zamanıdır (C_{max}). Son sütunda ise gerekli olan periyot adetleri görülmektedir. Çizelge 5.4'de görüldüğü gibi düşük periyot

Çizelge 5.4. Rassal örneklerle periyot uzunluğunun test edilmesi

Periyot	Çözüm süresi(sn)					C_{max}(saat)					Periyot adedi				
uzunluğu(s)	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
1	1800	1800	1800	1800	1800	29,4	66,1	45,7	63,4	57,4	34	67	46	64	58
2	1800	1800	1800	1549,3	1141,4	29	68	45,6	64,6	57	15	38	24	37	30
3	1800	1800	1800	71,9	402,2	31	69,6	45,6	63,6	60,5	11	25	16	23	22
4	99,5	1800	465	15,4	21,6	30,3	72	45,6	64,6	56,8	8	20	12	17	16
5	17,34	932,6	134,1	10,8	7,1	30,3	75	45,6	65,1	58,5	7	16	10	14	12
6	29,7	93,2	80	5,6	10,5	31	73,4	45,6	66,6	60,8	6	13	8	12	11
7	5	242,9	23,3	11,5	6,2	31,3	84,4	45,6	70,1	59,5	5	13	7	11	10
8	23,5	92	6,5	2,6	6	34,3	88,3	45,6	67,1	64,7	5	12	6	9	9
9	38,9	35,7	12	2,3	4,9	37	82	45,6	66,5	66,5	5	10	6	8	8
10	2,5	37,1	2,5	2,7	2,6	31,7	90	45,6	72,5	63,5	4	10	5	8	7
11	5,5	11,3	2,7	2,3	2,8	33,7	88,3	45,6	78,5	69,5	4	9	5	8	7
12	1,8	6,5	1,7	1,9	2,9	31	87,6	45,6	73,5	72,7	3	8	4	7	7

uzunluklarında optimal sonuçlar bulunamamış ve problem çözümü için gerekli olan periyot adedi de artmıştır. Örneğin, periyot uzunluğu bir saat alındığında, P1 problemini çözmek için 34 periyoda ihtiyaç duyulur ve optimal çizelgenin maksimum tamamlanma zamanı 29,4 saat olarak bulunur. Eğer periyot uzunluğu iki saat olarak değiştirilirse gerekli periyot adedi 15'e düşer ve karar değişkeni sayısı da azalır. Bununla birlikte C_{max} değeri 29 saate düşer ve sonrasında düzensiz olarak artışlar ve azalmalar görülür. Burada dikkat edilmesi gereken bu değerin aslında optimal çözüm olmadığıdır. Bir diğer deyişle, düşük periyot uzunluklarında C_{max} değerleri daha düşük bulunsa da gerekli periyot sayısı arttığından aslında problem de zorlaşmakta ve optimal çözüm bulunamamaktadır.

Periyot uzunluğuna göre C_{max} değişimi Şekil 5.2'de verilmiştir. Örnek problemlerin C_{max} değerleri üç saat boyunca dalgalanma gösterir ve dört saat sonrasında büyük artışlar görülür. Bu periyot uzunluklarında kalıp değişimi için harcanan süreler hemen hemen aynıdır. Çizelgelerin daha esnek hale gelmesi ve gereksiz üretimlerin azaltılması için kısa periyot uzunluklarına ihtiyaç duyulmaktadır. Bununla birlikte problemlerin optimal sonuçlarını elde edebilecek büyüklükte periyot uzunluğu belirlenmelidir. Bu sebeple, firmanın kabulünde olduğu gibi periyot uzunlukları dört saat olarak alınmıştır.



Şekil 5.2. Farklı periyot uzunluklarında C_{max} değişimi

Periyot uzunlukları matematiksel modeli hızlandıracak şekilde seçilse de NP-zor yapısından dolayı KTP1 modeli orta ve büyük boyutlu problemleri süre limitleri içinde optimal olarak çözememektedir. Haftalık çizelgelerde ortalama 50 işin olduğu göz önünde bulundurulursa bu modelin gerçek problem çözümünde kullanılamayacağı aşikârdır.

Diğer taraftan kısıt programlama son yıllarda çizelgeleme problemlerinde kullanılır olmuş ve birçok araştırmacı başarılı çizelgeleme uygulamaları geliştirmişlerdir (Nuijten ve Aarts 1996, Van Hentenryck 1999, Khayat ve ark. 2006). Kısıt programlamanın kendine özgü değişken tanımları ve kısıt ifadeleri, çizelgeleme problemlerinin çözümünü hızlandırmaktadır. Bu sebeple pres hattı çizelgeleme problemi için, kısıt programlamanın çizelgeleme problemlerine özgü yapısı kullanılarak bir kısıt programlama modeli (KP1) geliştirilmiş ve sonraki bölümde detaylandırılmıştır.

5.3. KP1 Modeli

Kısıt programlama (KP), kısıt sağlama problemlerinin (KSP) çözümü için geliştirilmiştir (Baptiste ve ark. 2001). KSP tanımında *değişkenler*, her değişkenin alabileceği *değerler kümesi* (domains) ve *kısıtlar* vardır. KSP'nin amacı, değerlerin değişkenlere atanarak bütün kısıtların sağlandığı bir atama bulmaktır. KSP, kısıtları baz alarak değişkenlerin değer kümelerinde elemeler yapar, gereksiz değerleri çıkarır ve çözüm uzayını küçültür. Bu küçültme *kısıt yayılımı* (constraint propagation) olarak adlandırılır ve kısıt programlamaya özgü bir özelliktir. Diğer taraftan KP, KSP'nin amaç fonksiyonuna sahip halidir. KP de kısıtları sağlayan çözümlerle ilgilenir, ancak KSP gibi ilk uygun çözümde durmaz; en iyi amaç fonksiyonu değeri için uygun çözümler arasında arama yapmaya devam eder. Kısıt yayılımı sayesinde hızlı bir şekilde uygun çözümlere ulaşır ve amaç fonksiyonlarını değerlendirir. Bu özelliği ile KP, NP-zor yapıda ve matematiksel programlama ile çözülemeyen problemlerde hızlı ve başarılı çözümler üretebilen bir yöntem olarak son yıllarda kullanımı artan bir yaklaşım olmuştur. Doktora tez çalışmasında ele alınan problem çözümü için bir kısıt programlama modeli (KP1) geliştirilmiştir. Ayrıca, pres çizelgeleme probleminin modellenmesi için ILOG CPLEX 12.4 programında OPL dili ve kısıt programlamanın özel değişken ve kısıt tanımları kullanılmıştır. Geliştirilen KP1 modelinde kısıtlar aşağıdaki gibi özetlenebilir:

- Her işin ardışık makinelere atanması gereken farklı sayıda operasyonu vardır.
- Bir iş bir üretim periyodunda bir yerde işlenebilir.
- Bir makine aynı anda yalnızca bir iş işleyebilir.
- Toplam üretim adedi talep kadar olmalıdır.

- İşler bölünemez.
- Bir iş bir makineye atandığında, bu işin operasyon sayısı kadar ardışık makineye başka iş atanamaz.

Matematiksel programlamadan farklı olarak kısıt programlama, çözüm uzayını kısıtları ve değişkenlerin değerlerini kullanarak azaltır ve bütün olası denemelerden sonra optimal sonuç kararını verir. Ayrıca, çizelgeleme problemleri için özel değişken tanımları vardır. Örneğin "*interval*" değişkeni ile bir işin boyutu, başlangıç ve bitiş süreleri tek değişkende depolanabilir. Ayrıca kısıt programlamada kullanılan "*alternative*", "*span*" ve "*noOverlap*" gibi özel kısıt tanımlamaları da modellemeyi kolaylaştıran ifadelerdir. KP1 modelinde kullanılan indisler, parametreler, kümeler ve değişkenler aşağıdaki gibidir:

İndisler(kümeler aşağıda tanımlanmıştır):

i: Çizelgelenecek işler, $i \in I$.

j_i: İşlerin operasyon adetleri, $j \in O_i$.

k: Periyotlar, $k \in K$.

m: Makineler, $m \in M$.

Parametreler:

A_{im}: Eğer *i* işi *m* makinesine atanabilirse A_{im}= 1 ve aksi halde sıfır.

U: Periot uzunluğu.

J_i: *i* işinin toplam süresi.

H_i: *i* işinin toplam periyot adedi: $H_i = \lceil J_i/3600 * U \rceil$.

D_i: *i* işinin talebi.

P_{ij}: *i* işinin *j* operasyonu için işlem süresi.

Kümeler:

I: İşler kümesi. *I*=$\{1,2,...,I_{max}\}$.

K: Periyotlar kümesi. *K*=$\{1,2,...,K_{max}\}$.

M: Makineler kümesi. *M*=$\{1,2,...,M_{max}\}$.

O_i: *i* işinin operasyonları kümesi. O_i=$\{1,2,...,O_{i_{max}}\}$.

Alter: $\{ (i,m) \mid i \in I, m \in M: A_{im} = 1\}$.

Curr: $\{ (i,j) \mid i \in I, j \in O_i: P_{ij} > 0\}$.

Karar Değişkenleri:

y_i: *i* işinin interval değişkeni; *size* J_i.

e_{ij}: *i* işinin *j* operasyonunun interval değişkeni; *in* $0..3600 * K_{max} * U$.

x_{im}:*i* işinin *m* makinesinin interval değişkeni $(i, m) \in$*Alter*;*optional in* $0..3600 * K_{max} * U$.

g_m: $\sum_{(i,m)\in Curr} pulse(e_{ij}, 1), \forall m \in M$. (Pulse aşağıda tanımlanmıştır.)

Operasyonlar ve işleri birbirleri ile ilişkilendirebilmek için bunlar "*interval*" olarak tanımlanır. İşlerin boyutu "*size*" ile tanımlanır ve J_i parametresine eşittir. Ayrıca her operasyonun tanım uzayı *"in"* ile tanımlanır; aslında bu tanım çizelgeleme dönemini temsil eder ve işlerin başlangıç ve bitişleri 0 ile $3600 * K_{max} * U$ zaman aralığında olmalıdır. Daha önce de bahsedildiği gibi her işin alternatif makine kümesi vardır ve işler alternatif makine kümelerindeki makinelerde başlayabilir. *i* işinin *m* makinesine atanması halinde başlangıç ve bitiş sürelerini barındıran x_{im} değişkeni "*interval*" olarak tanımlanır ve "*Alter*" ifadesi ile her işin alternatif makinelerinden yalnızca birine atanması sağlanır. Ayrıca değişken maksimum planlama dönemi ile sınırlanmıştır. Bu değişkende sınırlamaya gerek olmadığından *"optional in"* kullanılmıştır. Zaten e_{ij} değişkeninin sınırları x_{im} değişkeninin de sınırlarıdır. Bir operasyonun bir makine üzerinde bir birimlik kullanıma sahip olduğu varsayılır ve kümülatif kullanım fonksiyonu g_m hesaplamasında "*pulse*" ifadesi ile bu kullanımlar toplanır. Bu değişkenler kullanılarak geliştirilen KP1 modeli aşağıdaki gibidir:

$$min\, max_i(endOf(y_i)) \tag{5.17}$$

$$endBeforeStart(e_{ij}, e_{ij+1}), \forall i \in I, j \in O_i \mid j+1 \leq O_{i_{max}} \tag{5.18}$$

$$\begin{aligned} &presenceOf(e_{ij}) \overset{yields}{\rightarrow} presenceOf(e_{ij+1}), \\ &\forall i \in I, j \in O_i \mid j+1 \leq O_{i_{max}} \end{aligned} \tag{5.19}$$

$$span(y_i, all(j \in O_i)e_{ij}), \forall i \in I \tag{5.20}$$

$$alternative(y_i, all(a \in Alter : a.i = i)x_a), \forall i \in I \tag{5.21}$$

$$noOverlap(all(a \in Alter : a.m = m)x_a), \forall m \in M \tag{5.22}$$

$$\begin{pmatrix} presenceOf(x_a) > 0 \\ AND \\ presenceOf(x_b) > 0 \end{pmatrix} \overset{yields}{\rightarrow}$$
$$\begin{pmatrix} endOf(x_a) \leq startOf(x_b) \\ OR \\ endOf(x_b) \leq startOf(x_a) \end{pmatrix}$$
$$\forall a, b \in Alter \mid O_{a.i_{max}} > 1 \, AND \, b.i \neq a.i \, AND$$
$$b.m \geq a.m \, AND \, b.m \leq a.m + O_{a.i_{max}} - 1 \tag{5.23}$$

$$\sum_{m \in M} g_m \leq M \tag{5.24}$$

KP1'in amacı maksimum tamamlanma zamanının minimizasyonudur ve *endOf* ifadesi ile bir işin bitiş zamanına ulaşılır, dolayısıyla ifade (5.17) C_{max} minimizasyonunun kısıt programlama diliyle gösterimidir. Bu problemde işlerin kalıp değişim süreleri, operasyon sürelerine dâhil edilebilir. Kısıt (5.18)'de *endBeforeStart* terimi e_{ij+1} operasyonunun e_{ij} bitmeden başlamasını engeller, dolayısıyla bu kısıt operasyonların başlama zamanlarının ardışık olmasını sağlar. Kısıt (5.19)'daki *presenceOf* terimi mantıksal doğru-yanlış geri bildirimi olan 1-0 değerlerini yansıtır böylece kısıt (5.19), bir operasyonun varlığında diğerinin de olmasını ve dolaylı olarak işlerin operasyonları arasındaki bağlantıyı sağlar. Kısıt (5.20) *span* terimi ile, bir işin başlangıç ve bitiş süreleri arasında bütün operasyonlarını kapsamasını sağlar. Kısıt (5.21) her işin sadece alternatif makine kümesindeki bir makinede başlamasını garanti eder. Kısıt (5.22) *noOverlap* terimiyle bir makinenin aynı anda birden fazla iş işlemesini engeller. Kısıt (5.23) ile eğer *a* işi *m* makinesine atanırsa gerekli operasyon adedi boyunca yanındaki makinelere ve *a* işinin bitiş zamanına kadar bu makineye başka bir *b* işinin atanması engellenir. Kısıt (5.24) ile kümülatif makine kullanımı makine kapasitesiyle sınırlanır.

KP1 performansını test etmek amacıyla gerçek verilerden elde edilen 10 problem kullanılmış ve model KTP1 ile karşılaştırılmıştır. KTP1'de ve KP1'de işlerin bölünemez olduğu kabul edilmiştir. KTP1 modeli 1800 saniye ile ve KP1 modeli (5.3.1) bölümünde açıklandığı üzere 700 000 hata limiti ile sınırlanmıştır. Deneyler 2.90 GHz i7 işlemcili 6 GB RAM bilgisayarda çalıştırılmıştır. Çizelge (5.5)'de ilk üç sütun problemlerin barındırdığı iş adedi ve problem çözümü için gerekli olan minimum periyot adedini göstermektedir. Dördüncü kısım KTP1 ve KP1 modelleri tarafından hesaplanan çizelgelerin uzunluklarını, beşinci kısım da bu çözümlere ulaşabilmek için harcanan süreyi göstermektedir. Fark sütunu önceki bölümdeki (5.15) ifadesi ile hesaplanır.

Çizelge 5.5. Gerçek veri ile KTP1 ve KP1 karşılaştırması

			C_{max} (saat)		Çözüm süresi(sn)		%Fark	%EAOS
Problem	İşler	Periyotlar	KTP1	KP1	KTP1	KP1	KTP1	KTP1-KP1
1	5	5	16,3	16,3	1,1	2,9	-	-
2	10	8	29,3	29,3	6,7	2,7	-	-
3	15	13	44,9	**43,9**	1800	109,7	22,3	2,1
4	20	22	77,6	**73,8**	1800	119,5	63,7	4,9
5	25	25	85,0	**78,8**	1800	136,1	78,7	7,3
6	30	27	99,0	**89,0**	1800	141,8	87,8	10,0
7	35	32	119,2	**98,2**	1800	175,2	91,3	17,6
8	40	40	146,8	**126,1**	1800	211,0	95,9	14,1
9	45	49	-	**156,4**	1800	305,7	-	-
10	52	54	-	**170,0**	1800	371,2	-	-

KTP1 yerine KP1 modeli kullanılarak oluşturulan çizelge ile elde edilen iyileştirmenin hesaplanması için en iyi amacın ortalama yüzde sapması (EAOS) (average percentage deviation of the best objective line) kullanılmıştır (Meyr ve Mann 2013).

$$\%EAOS = \frac{KTP1_{C_{max}} - KP1_{C_{max}}}{KTP1_{C_{max}}} * (100) \tag{5.25}$$

Çizelge (5.5)'de de görüldüğü gibi KTP1 modeli büyük boyutlu problemlerin optimal çözümlerini bulamamaktadır. Diğer taraftan, KP1 modeli KTP1'den daha başarılıdır ve ortalama 157,6 saniyede başarılı çizelgeler üretebilmektedir. Ayrıca KP1 problemlerin uygun çözümlerinin kalitesini %7 oranında iyileştirmektedir ve büyük boyutlu problemlerde başarılı çizelgeler üretebilmektedir. Fakat, modelin durma kriterinin

kullanıcı tarafından belirlenmesi çözümlerde sezgiselliğe sebep olmaktadır. Çözüm kalitesi ve çözüm süresi arasındaki dengeyi sağlayan bir durma kriteri belirlenmelidir. Bu değerlendirme için gerekli deneyler bir sonraki bölümde detaylandırılmıştır.

5.3.1. KP1 modelinde etkin durdurma ölçütü belirlenmesi

KP1'nin çözüm prosedürü KTP1'den farklıdır. KP1 ilk önce bütün uygun olmayan çözüm noktalarını çözüm uzayından çıkarır, sonrasında yayılma teknikleri ile uygun çözüm noktalarından bir çözüm uzayı oluşturur. Bütün alternatif çözüm noktalarını deneyerek en iyi amaç fonksiyonu değerini belirlemeye çalışır. Bu anlamda yeterli süre verildiğinde optimal çözümü garantileyen bir yöntemdir. Ancak, durdurma kriteri kullanıcı tarafından belirlenir ve bu durum çözüm kalitesini de etkiler. Eğer kullanıcı düşük süreli bir limit belirlerse, yöntem uygun çözümü hızlı bir şekilde elde eder, fakat bu çözüm ile optimum arasındaki hassasiyet düşük olur. Diğer taraftan kullanıcı yüksek süreli bir limit belirlerse, çözüm kalitesi iyileşir ancak çözüm süresi de uzar. Durdurma kriteri, KP1 modelinde önemli bir parametre olduğu için, farklı problemler kullanılarak deneyler yapılmış ve uygun bir kriter belirlenmeye çalışılmıştır. Gerçek üretim verilerinden 10 haftalık veri incelenmiş ve her hafta bir problem oluşturacak şekilde deney kümesi oluşturulmuştur.

Kısıt programlama modelleri hata limiti veya süre limiti ile sınırlandırılabilir. Daha önce de bahsedildiği gibi KP1 modeli uygun çözüm noktalarını deneyerek çözüm uzayında arama yapar. Uygun olmayan noktaları "hatalı nokta" olarak kaydeder. Hata limiti, bu denenecek noktaların sayısını sınırlandırmaktadır. Örneğin, hata limiti 300 000 olarak belirlenirse KP1 modeli 300 000 uygun çözüm noktasında amaç fonksiyonu değerlerini hesaplar ve en iyi çözümü kaydeder, sonrasında ise durur. Bu deneylerde KP1 model farklı hata limitleri ve 1800 saniye süre limiti kullanılarak sınırlandırılmıştır. 1800 saniye limitli çözümler en iyi çözüm olarak alınmış ve farklı hata limitlerindeki çözümler için %EAOS değerleri hesaplanmıştır. Böylece farklı hata limitlerindeki çözümlerle 30 dakikalık çözümler arasındaki fark bulunmuştur. Deney sonuçları Çizelge 5.6 ve Çizelge 5.7'de detaylandırılmıştır.

Çizelge 5.6'da ilk iki sütun problemler ile ilgili bilgi vermektedir. Diğer sütunlar ise KP1 modellerinin C_{max} değerleri arasındaki %EAOS farklarını göstermektedir. Örnek olarak, 100 000 hata limiti ile KP1 modeli çalıştırılırsa 1800 saniyede bulunan çözümlerden ortalama %13,2 daha kötü çizelgeler elde edilmektedir. Benzer olarak, eğer 700 000

Çizelge 5.6. Farklı limitlerde KP1 modeli çözümleri

		Farklı Hata Limitlerinde % EAOS Değerleri							
Problem	İşler	KP1_100000	KP1_300000	KP1_500000	KP1_700000	KP1_900000	KP1_1000000	KP1_1500000	KP1_1800sn.
1	63	-12,4	-8,1	-5,8	-3	-2,3	-2,3	-1,7	0
2	64	-14,6	-8,5	-6,6	-6,6	-6,6	-6,6	-4,2	0
3	55	-11,8	-5,7	-5,7	-5,7	-5,7	-5,7	-5,7	0
4	61	-13,3	-7,5	-5,8	-5,1	-5,1	-5,1	-2,7	0
5	59	-6,8	-2,4	-2,3	-1,2	-1,2	-1,2	0	0
6	58	-10,4	-5,9	-5,9	-0,6	0	0	0	0
7	57	-27,7	-20,2	-7,2	-7,2	-4	-2,3	-1,5	0
8	54	-9,8	0	0	0	0	0	0	0
9	54	-15,2	-8,8	-6,6	-2,8	-2,8	-2,8	0	0
10	49	-10	-6,1	-4,6	-1,3	-1,1	0	0	0
	Ort.	-13,2	-7,3	-5	-3,4	-2,9	-2,6	-1,6	0

hata limiti kullanılırsa %3,4 oranında kötü çizelgeler elde edilmekte, ancak bu sonuçlara 1800 saniye yerine ortalama 353 saniyede ulaşılmaktadır. Bunun yerine, hata limiti 900 000 olarak belirlenirse ortalama çözüm süresi 446,4 saniye olmakta, fakat çözüm kalitesi sadece %0,5 oranında artmaktadır. Sonuç olarak, çözüm kalitesi ve çözüm süresi arasındaki denge incelenmiş ve hata limiti 700000 olarak belirlenmiştir. KP1

Çizelge 5.7. Farklı limitlerde KP1 modeli çözüm süreleri

		Farklı Hata Limitlerinde Çözüm Süresi(sn)							
Problem	İşler	KP1_100000	KP1_300000	KP1_500000	KP1_700000	KP1_900000	KP1_1000000	KP1_1500000	KP1_1800sn.
1	63	31,5	113,2	199	320,4	578,4	640,5	981,9	1800
2	64	35,7	131	262,3	522,4	659	702,2	1055,8	1800
3	55	57,8	172,1	258,8	435,6	569	609,7	943,3	1800
4	61	40,3	108,4	164,9	222,4	287	319,9	450,6	1800
5	59	65,6	187,4	306,6	546,6	578,9	621	898,2	1800
6	58	56,5	235,7	379,1	549,6	674,6	756,1	1206,6	1800
7	57	24,9	78,5	108,8	157,8	118,3	247,5	427,8	1800
8	54	54,4	155,6	259,5	393,9	466,4	523,1	817,7	1800
9	54	20,2	47,7	76,8	140,1	227,2	268,9	424,1	1800
10	49	28,4	87,3	195,1	241,5	305,3	393,2	696,1	1800
	Ort.	41,5	131,7	221,1	353	446,4	508,2	790,2	1800

modeli için en uygun hata limiti belirlenmesi deneylerinde modelin 1800 saniye süre limiti ile optimal sonuçlara ulaşamadığı görülmüştür. Diğer taraftan KTP1 modeli de büyük boyutlu problemler için benzer sürelerde optimal çözümleri üretememektedir. Bu sebeple, iki yöntem için çeşitli iyileştirme çalışmaları gerçekleştirilmiş ve sonraki bölümde detaylandırılmıştır.

5.3.2. KP1 ile KTP1 iyileştirmeleri

KTP1 modeli gerçek boyutlu problemlerde yavaş çalışsa da çözümün optimalliği hakkında bilgi sağlayan bir yöntemdir. Bu sebeple iyileştirme çalışmaları KTP1 ile başlamış ve modeli hızlandırabilmek için bir alt sınır arayışına gidilmiştir. KTP1 modeli gevşetilerek bir doğrusal programlama modeli (DP1) oluşturulmuş ve bu modelin amaç fonksiyonu, KTP1 modelinin alt sınırını belirlemiştir. KTP1 modeli bu sınırla çözüme başlamış ve önceki bölümde belirtilen kısıtlarla optimal sonuçları bulmaya çalışmıştır. Çizelge 5.8'de DP1, KTP1 ve KP1 modelleri deney sonuçları gösterilmektedir. Çizelgeden de görüldüğü gibi DP1 modeli ile üretilen ve alt sınır olarak kullanılan değerler, KTP1 modeliyle bulunan amaç fonksiyonu değerlerinden çok uzaktır. Dolayısıyla DP1 çözümleri, KTP1 için yararlı bir alt sınır sağlayamamaktadır. Diğer taraftan KP1, kısa sürelerde başarılı atamalar elde edebilmekte, ancak sonucun

Çizelge 5.8. Gerçek veri ile KTP1, KP1 ve DP1 gevşetmesi karşılaştırılması

			C_{max}(s)			Çözüm süresi(sn)		
Problem	İşler	Periyotlar	DP1	KTP1	KP1	DP1	KTP1	KP1
1	5	5	11,7	16,3	16,3	0,1	1,1	2,9
2	10	8	19,0	29,3	29,3	0,1	6,7	2,7
3	15	13	18,9	44,9	**43,9**	1,0	1800	109,7
4	20	22	18,9	77,6	**73,8**	1,7	1800	119,5
5	25	25	18,9	85,0	**78,8**	2,0	1800	136,1
6	30	27	45,5	99,0	**89,0**	2,5	1800	141,8
7	35	32	45,5	119,2	**98,2**	3,1	1800	175,2
8	40	40	45,5	146,8	**126,1**	8,2	1800	211,0
9	45	49	48,4	-	**156,4**	40,7	1800	305,7
10	52	54	48,3	-	**170,0**	75,8	1800	371,2

optimalliğini ispat edememektedir. KP1'nin bu özelliği kullanılarak uygun çözümler hızlı bir şekilde bulunabilir ve KTP1 bu çözüm noktaları ile çözüme başlarsa çözüm uzayını daha hızlı bir şekilde küçültebilir. Bu düşünce ile ikinci yaklaşım olarak, önce KP1 çalıştırılmış ve belirlenen limitlerle model çözülmüştür. Daha sonra elde edilen uygun çözüm, KTP1'e başlangıç çözüm olarak verilmiştir. KTP1 modeli bu çözümle işe başlamış ve bu çözümden daha iyi çözümlere ilerlemeye çalışmıştır. Bulunan sonuçlar, başlangıç çözümü olmadan çalıştırılan KTP1 ile karşılaştırılmıştır. Deneylerde ILOG CPLEX 12.4 programının model çözüm aşamaları kaydedilmiştir. Deney sonuçları

incelendiğinde, KTP1'in verilen başlangıç çözümle daha iyi bir alt sınırdan başladığı ancak bunun modeli hızlandırmada yeterli olmadığı görülmüştür. ILOG CPLEX 12.4'ün yaptığı kesimler (http://www-eio.upc.es/lceio/manuals/cplex-11/ html/ usrcplex/ solveMIP14.html, 2014) incelendiğinde, başlangıç çözümü olmadan çalıştırılan KTP1'de daha fazla kesim yaptığı görülmüştür. Sonuç olarak, KP1 ile başlangıç çözümlerin KTP1'e verilmesi de modeli hızlandırmada yeterli olmamıştır.

Üçüncü yaklaşım olarak, KP1 amaç fonksiyonunun alt sınır olarak kullanılması düşünülmüştür. KP1 ile elde edilen amaç fonksiyonu değeri alt sınır olarak KTP1'e verilmiş ve KTP1 bu alt sınırla başlamıştır. Deney sonuçları incelendiğinde, KTP1'in KP1'den gelen alt sınır ile çözüme başladığında daha farklı sayıda kesim yaptığı gözlenmiş, ancak bu iyileşme de KTP1'i hızlandırmada yeterli olmamıştır. Aslında optimal olan bir değer alt sınır olarak verildiğinde dahi, KTP1 performansında beklenen iyileşme sağlanamamıştır. Sonuç olarak, KTP1 modelinin iyileştirilmesi için geliştirilen üç farklı çalışma beklenen faydayı sağlayamamıştır. Bu durum, farklı arayışları da beraberinde getirmiş ve bir sonraki bölümde detaylandırılan KP1 modelinin iyileştirilmesi çalışmaları gerçekleştirilmiştir.

5.3.3. KP1 iyileştirme çalışmaları

Daha önce de bahsedildiği gibi, KP1 modeli başarılı çözümlere hızlı bir şekilde ulaşabilmekte ancak çözümün optimalliğini ispat edememektedir. Eğer KP1 performansı iyileştirilebilirse, optimal sonuca ulaşmak kolaylaşabilir. KP, kısıt yayılımı aşamasında değişken ve değer seçimi kurallarını barındıran arama algoritmalarının kullanıcı tarafından tanımlanmasına izin verir. Bu amaçla KP1'de çeşitli arama algoritmaları geliştirilmiş ve bunların performans değerlendirmesi yapılmıştır. Pres çizelgeleme probleminde işlerin alternatif makinelere atanmasını temsil eden değişken ele alınmış ve değişken seçimi gerçekleştirilmiştir. Bu değişken seçimiyle üç farklı değer seçimi metodu geliştirilmiştir; işlerin ilk önce en küçük indisli alternatif makineye atanması (a), işlerin ilk önce en büyük indisli alternatif makineye atanması (b) ve işlerin rassal olarak alternatif makinelerden birine atanması (c). Geliştirilen üç arama algoritması ile KP1

deneyleri tekrarlanmış ve modeller KP1(a), KP1(b) ve KP1(c) olarak temsil edilmiştir. Karşılaştırmalarda modellerin verilen limitler içinde (100 000 hata limiti) buldukları çözüm adetleri, ürettikleri dal sayısı ve bulabildikleri potansiyel çözüm noktaları dikkate alınmıştır. Sadece iki problem için yapılan deney sonuçları Çizelge 5.9 ve Çizelge 5.10'da özetlenmiştir.

Çizelge 5.9. Arama algoritmalarının karşılaştırılması: C_{max} ve çözüm süresi

	C_{max}				Çözüm Süresi			
İş sayısı	KP1	KP1(a)	KP1(b)	KP1(c)	KP1	KP1(a)	KP1(b)	KP1(c)
10	131,3	131,3	131,3	131,3	**2,8**	30,8	32,1	31,4
15	259,9	259,9	259,9	259,9	**36,1**	56,2	59,1	59,6

Çizelge 5.10. Arama algoritmalarının çözüm adetleri, dal sayıları ve seçim noktaları

	Çözüm adedi				Dal sayısı				Seçim noktaları			
İş sayısı	KP1	KP1(a)	KP1(b)	KP1(c)	KP1	KP1(a)	KP1(b)	KP1(c)	KP1	KP1(a)	KP1(b)	KP1(c)
10	1	1	1	1	25259	233081	233081	233081	14703	143865	143865	143865
15	3	2	2	2	238073	237273	237273	237273	141762	141294	141294	141294

Yapılan deneyler sonunda üç senaryoda da dal sayısı ve seçim noktalarının aynı olduğu görülmüştür. KP1(a) modeli diğer modellere göre biraz daha hızlı çalıştığı için bu modelle büyük örnekli karşılaştırmalara devam edilmiştir. KP1 ve KP1(a) karşılaştırma sonuçları Çizelge 5.11'de özetlenmiştir. Farklı büyüklüklerde rassal verilerle örnekler üretilmiştir. Bu örnekler kullanılarak oluşturulan deney sonuçları çözüm süresi, amaç fonksiyonu değeri, bulunan çözüm adetleri, dal sayısı ve potansiyel çözümleri barındıran seçim noktaları olarak detaylandırılmıştır. Dal sayıları, seçim noktaları ve amaç fonksiyonları arasındaki farklar yüzdesel olarak hesaplanmıştır.

Çizelge 5.11. KP1 ve KP1(a) performans karşılaştırması

	KP1					KP1(a)					%EAOS		
İşler	Süre	C_{max}	Çözüm sayısı	Dal sayısı	Seçim noktaları	Süre	C_{max}	Çözüm sayısı	Dal sayısı	Seçim noktaları	Dal sayısı	Seçim noktaları	C_{max}
10	2,8	131,3	1	25259	14703	30,8	131,3	1	233081	143865	-822,8	-878,5	0
15	36,1	259,9	3	238073	141762	56,2	259,9	2	237273	141294	0,3	0,3	0
20	73,5	415,8	1	254126	156923	105,5	415,8	1	230526	133991	9,3	14,6	0
25	99,1	468,8	6	242452	143672	148,5	468,8	3	235992	138156	2,7	3,8	0
30	143,8	**551,0**	9	264249	166401	136,8	551,3	5	227885	130216	13,8	21,7	-0,1
35	191,9	636,9	7	257140	158066	251,0	**632,2**	10	238141	139121	7,4	12,0	0,7
40	267,2	**770,2**	4	285566	187026	366,1	772,0	8	253825	152946	11,1	18,2	-0,2
45	313,6	**909,8**	6	253935	154073	471,4	924,6	4	252853	153480	0,4	0,4	-1,6
50	494,6	**982,5**	7	280986	180449	739,8	988,1	10	259643	159266	7,6	11,7	-0,6
										Ort.	6,6	10,3	-0,2

Çizelge 5.11 genel olarak incelendiğinde ilk dört örnek için aynı sonuçların KP1 tarafından daha kısa sürelerde bulunduğu görülmektedir. Çizelgenin son kısmı incelendiğinde ilk örnek haricinde (KP1 optimal sonucu bulduğu için dallanmayı

25259'da bitirmiştir, bu sebeple ilk örnek %EAOS hesaplamalarına dahil edilmemiştir) KP1(a) ile dal sayısının %6,6 oranında azaltıldığı görülmektedir. Benzer bir şekilde potansiyel çözümleri içeren seçim noktaları da KP1(a) ile %10,3 azalmıştır. Bu iyileştirmeler, eklenen arama algoritmasının çözüm uzayını küçülttüğünü gösterse de KP1(a) ile çözümlerin kalitesi %0,2 kötüleşmektedir. Bunun sebebi, KP1'nin hata limiti olarak belirlenen 100 000 noktayı daha seri şekilde taramasıdır. KP1(a)'da modele eklenen seçim kuralları, modelin seçim noktaları arasındaki ilerleyişini yavaşlatmıştır. Sonuç olarak, KP1 modeli geliştirilen arama metodları olmadan daha hızlı çalışmaktadır.

Bu çalışmalara ek olarak, farklı arama algoritmaları geliştirebilmek için üretim çizelgesi üzerinde büyük etkiye sahip işler dikkate alınmıştır. Bu işler operasyon adedi ve üretim adedi olarak hacimli işler olup çizelgede yer kaplayan işlerdir. Etkisi en büyük olan işlerin tespit edilerek önceden atanmasını gerçekleştirecek bir algoritma geliştirilmiştir. Ancak, bu arama algoritması ile önceki algoritmalar arasında anlamlı bir farklılık gözlenmemiştir.

KP1 performansı farklı arama stratejileri ile arttırılamasa da modele özel kısıtlar eklenerek çözüm uzayı küçültülebilir. *Simetri kırma* (symmetry breaking) olarak adlandırılan kısıtlar, çözüm uzayında yapılan elemelerde simetrik noktaların kısıt yayılımı ile elenmesini sağlar ve modeli hızlandırır. Üretim çizelgeleri zaman/periyot düzleminde düşünüldüğünde toplam çizelge zamanının ilk yarısında yapılan atamalar, çizelge uzunluğunu değiştirmediği sürece ikinci yarısından itibaren yapılabilir; dolayısıyla bu çözüm, uygun çözüm noktası olarak çözüm uzayında yer alır. Bu çözümlerin, ele alınan problemde teslim tarihleri olmadığından amaç fonksiyonunda bir farklılık oluşturmayacağı için elenmesi KP1 modelini hızlandırabilir. Bu düşünceyi test etmek amacıyla makinelerin boş kalma sürelerinin minimizasyonu hedeflenmiş, makine ve periyoda bağımlı yeni bir değişken $occupied_{mk}$ tanımlanmıştır. Bu değişken *m* makinesini *k* periyodunda kontrol eder, eğer makine herhangi bir işi işliyorsa değişkenin değeri bir olur; eğer herhangi bir iş yoksa yani makine boşsa sıfır değerini alır. Bu kontrol her makine ve periyot için yapılır. Tanımlanan değişken kullanılarak simetri kırma kısıtı geliştirilir ve KP1 modeline eklenir. Kısıt (5.26) toplam periyotların ilk yarısındaki atamaların, sonraki yarısına eşit veya daha büyük olması gerektiğini vurgular. Bu kısıt

ile hem simetri kırma yapılmış olur hem de ilk kısımdaki periyotların daha dolu olması sağlanır.

$$\sum_{m=1}^{M}\sum_{k=1}^{K/2} occupied_{mk} \geq \sum_{m=1}^{M}\sum_{k=1+K/2}^{K} occupied_{mk} \tag{5.26}$$

Kısıt (5.26)'nın KP1 performansına etkisini incelemek amacıyla rassal olarak üretilen problemlerle deneyler tekrarlanmış ve sonuçlar Çizelge 5.12'de detaylandırılmıştır. KP1 modeli tek başına ve kısıt (5.26) ile çalıştırılmış ve deney sonuçları KP1(kısıt(5.26)) ile temsil edilmiştir. Deneylerde çözüm süreleri, amaç fonksiyonu değerleri, ulaşılan çözüm adetleri, oluşturulan dal sayıları ve seçim noktaları dikkate alınmıştır. Ayrıca son kısımda iki modelin performanslarını karşılaştırmak için %EAOS değerleri hesaplanmıştır. Yeni kısıtla, beklendiği gibi dal sayısı ve seçim noktalarında artış meydana gelmiştir, ancak bu artış modelin performansını olumsuz etkilemiştir. Kısıt (5.26)'sız ve değişkensiz KP1 modeli aynı limitlerde daha iyi çözümlere ulaşabilmiştir. İki model tarafından üretilen çizelgeler incelendiğinde doluluk oranı açısından çizelgelerde bir fark olmadığı gözlenmiştir. Sonuç olarak, KP1'nin kısıt (5.26)'ya gerek kalmadan, boşlukları azaltmaya yönelik atamalar yaptığı söylenebilir.

Çizelge 5.12. Kısıt 5.26'un KP1 performansına etkisi

	KP1					KP1(kısıt(5.26))					%EAOS		
İşler	Süre	C_{max}	Çözüm sayısı	Dal sayısı	Seçim noktaları	Süre	C_{max}	Çözüm sayısı	Dal sayısı	Seçim noktaları	Dal sayısı	Seçim noktaları	C_{max}
10	2,8	131,3*	1	25259	14703	4,4	131,3*	1	25433	16350	-0,7	-11,2	-
15	36,1	259,9	3	238073	141762	92,1	259,9	4	313010	215426	-31,5	-52,0	-
20	73,5	415,8	1	254126	156923	376,4	415,8	4	394509	297572	-55,2	-89,6	-
25	99,1	468,8	6	242452	143672	456,3	468,8	7	365114	267207	-50,6	-86,0	-
30	143,8	**551,0**	9	264249	166401	366,4	551,3	8	293161	194710	-10,9	-17,0	-0,1
35	191,9	636,9	7	257140	158066	667,6	637,9	8	305326	206781	-18,7	-30,8	-0,1
40	267,2	**770,2**	4	285566	187026	865,6	771,9	3	311744	212921	-9,2	-13,8	-0,2
45	313,6	**909,8**	6	253935	154073	1114,5	**906,5**	11	323773	225204	-27,5	-46,2	0,4
50	494,6	**982,5**	7	280986	180449	1833,8	984,8	20	321911	222315	-14,6	-23,2	-0,2
										Ort.	-24,3	-41,1	-0,1

KP1(kısıt(5.26))'da çözüm uzayı zaman ekseninde ikiye ayrılarak azaltılmaya çalışılmış, ancak bu azaltmanın mevcut KP1 ile yapıldığı tespit edilmiştir. Dolayısıyla "$occupied_{mk}$" değişkeninin ve kısıt (5.26)'un eklenmesi modeli hızlandıramamıştır. Alternatif olarak çözüm uzayı makineler ekseninde ele alınabilir. KP1'nin yaptığı atamalar ve işlerin alternatif makineleri incelendiğinde makinelerin sırasının (pres hattında sıralı ardışık makineler) ilk yarısında yapılan atamaların daha fazla olduğu tespit edilmiştir. Bu sebeple yeni bir simetri kırma denklemi (5.27) geliştirilmiştir.

$$\sum_{a \in Alter | a.m \leq 6} presenceOf(x_a) \geq \sum_{a \in Alter | a.m > 6} presenceOf(x_a) \qquad (5.27)$$

Kısıt (5.27) ile deneyler tekrarlanarak etkileri incelenmiştir. Çizelge 5.13'de özetlenen sonuçlar incelendiğinde, KP1(kısıt(5.27))'nin çözüm uzayını yalnızca ortalama %1,7 azaltmasına rağmen daha fazla uygun çözüme ulaştığı ve hatta büyük örneklerde daha iyi çözümlere ulaşarak daha düşük amaç fonksiyonu değerleri bulduğu gözlenmiştir. Dolayısıyla kısıt (5.27) ile amaç fonksiyonunda %0,05 iyileşme sağlanabilmiştir.

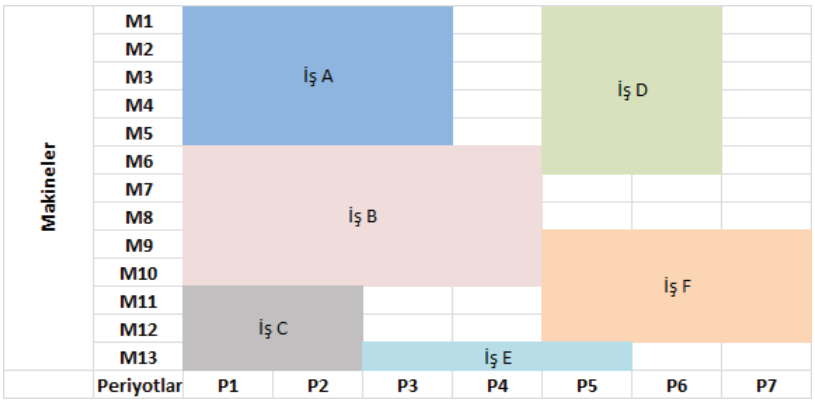
Çizelge 5.13. Kısıt 5.27'in KP1 performansına etkisi

	KP1					KP1(kısıt(5.27))					%EAOS		
İşler	Süre	C_{max}	Çözüm adedi	Dal sayısı	Seçim noktaları	Süre	C_{max}	Çözüm adedi	Dal sayısı	Seçim noktaları	Dal sayısı	Seçim noktaları	C_{max}
10	2,8	131,3	1	25259	14703	3	131,3	1	22231	12942	12	12	0
15	36,1	259,9	3	238073	141762	39,9	259,9	2	249220	154250	-4,7	-8,8	0
20	73,5	415,8	1	254126	156923	74,5	415,8	1	254159	156949	0	0	0
25	99,1	468,8	6	242452	143672	96,8	468,8	6	240544	142679	0,8	0,7	0
30	143,8	551	9	264249	166401	131,9	551	10	257615	160432	2,5	3,6	0
35	191,9	636,9	7	257140	158066	178,4	636,9	7	254668	155192	1	1,8	0
40	267,2	770,2	4	285566	187026	240,4	770,2	4	285798	187381	-0,1	-0,2	0
45	313,6	**909,8**	6	253935	154073	319,2	**906,5**	5	266882	167312	-5,1	-8,6	0,36
50	494,6	**982,5**	7	280986	180449	451,4	**981,8**	15	255443	155303	9,1	13,9	0,07
										Ort.	1,7	1,6	0,05

Bu bölümde KP1 modeli performansını iyileştirebilmek için farklı arama algoritmaları geliştirilmiş ve yeni simetri kırma kısıtları ile modeller test edilmiştir. Kısıt (5.27) ile iyileşme sağlanabilmiş, ancak optimale erişebilmek için yeterli olamamıştır. KP1'de her iş talebi kadar üretilir ve işlem süreleri talep kadar üretim göz önüne alınarak hesaplanır. Bu değer işlerin zaman sınırını temsil etmektedir. Bununla birlikte, işlerin operasyon adetleri de bir başka sınırı oluşturmakta ve KP1 işlerin atamasını bu sınırları dikkate alarak bloklar halinde yapmaktadır. Bloklar halinde atama yapıldığı için üretim adetleri değişken olmaktan çıkar ve parametre olarak kullanılır, dolayısıyla bu değişiklik modelin çözümünü kolaylaştırır. Bu basitleştirmelere rağmen KP1'de dual çözümden gelen bir alt sınır yoktur ve optimalin ispatı bütün olası çözümlerin denenmesiyle yapılır. Dolayısıyla, KP1'nin dezavantajı optimali ispatlayamamasıdır. Bu dezavantajı ortadan kaldırmak için blok atama yapabilen yeni bir tamsayılı matematiksel model geliştirilmiştir (TP1). TP1 modelinde üretim sadece talep kadar gerçekleşir ve işler bölünemezdir. Dolayısıyla, işlerin üretim adetleri değişken olmaktan çıkar ve blok atamanın sınırlarını belirlemede kullanılan bir parametre olarak modele eklenir. Geliştirilen matematiksel model sonraki bölümde detaylandırılmıştır.

5.4. TP1 Modeli

Bu bölümde pres hattı çizelgeleme problemi için yeni bir tamsayılı model geliştirilmiştir (TP1). TP1 modelinde işlerin bölünemez olduğu kabul edilmiştir. Bu kabul, kalıp değişim sürelerini değişken olmaktan kurtarır ve kalıp değişim süreleri sabit olarak üretim sürelerine eklenebilir. Bir diğer kabul, işlerin talepleri kadar üretilmesidir. Bu kabulle bir işin tamamlanması için gereken süre hesaplanmakta ve bu süreden periyot sayısı hesaplanabilmektedir. İşlerin operasyon sayıları da bilindiğinden işlerin bloklar halinde tanımlanması gerçekleştirilmiş olur. Ayrıca KTP1'de bahsedilen şekilde ilk uygun çözümün bulunduğu nokta kullanılarak tüm işlerin atanması için gerekli olan periyot sayısı belirlenir.



Şekil 5.3. Blok atama örneği

Şekil 5.3'de görüldüğü gibi planlama döneminde dikey eksen makineleri ve yatay eksen de zamanı temsil etsin. İşlerin yüksekliği operasyon sayıları ile ve genişliği de işin tamamlanması için gereken periyot ile belirlenerek bloğun sınırları tanımlanır. Bu bloklar dikkate alınarak TP1'de işlerin makinelere ataması gerçekleştirilir.

TP1 modeli notasyonunda problemler için gerekli olan periyot adedi parametresi K_{max} problemin zaman sınırını temsil eder. Bölüm 5.2'de de bahsedildiği gibi, bu parametrenin değeri büyük bir sayı olarak belirlenebilir. Ancak çözüm uzayını küçültmek ve modelin çözüm hızını arttırmak amacıyla deneylerde herhangi bir problem için ilk uygun çözüm noktasının elde edildiği K_{max} parametresi dikkate alınmıştır. Modelde kullanılan indisler, parametreler, kümeler ve değişkenler aşağıdaki gibidir:

İndisler(kümeler aşağıda tanımlanmıştır):

i: Çizelgelenecek işler, $i \in I$.

j: İşlerin operasyon adetleri, $j \in O_i$.

k: Periyotlar, $k \in K$.

m: Makineler, $m \in M$.

Parametreler:

A_{im}: Eğer *i* işi *m* makinesine atanabilirse A_{im}= 1 ve aksi halde sıfır.

J_i: *i* işinin toplam süresi.

U: Periot uzunluğu.

H_i: *i* işinin toplam periyodu; $\lceil (J_i/3600)/U \rceil$.

Kümeler:

I: İşler kümesi. *I*=$\{1,2,...,I_{max}\}$.

K: Periyotlar kümesi. *K*=$\{1,2,...,K_{max}\}$.

M: Makineler kümesi. *M*=$\{1,2,...,M_{max}\}$.

O_i: *i* işinin operasyonları kümesi. O_i=$\{1,2,...,O_{i_{max}}\}$.

Set_1: $\{ (i,k,m) \mid i \in I, k \in K, m \in M, A_{im} = 1\}$.

Karar Değişkenleri:

x_{ikm}= $\begin{cases} 1, \text{Eğer } i\text{.iş } k\text{.periyotta } m\text{.makinede başlarsa;} \\ 0, \text{Aksi halde} \end{cases}$

C_{max}: Toplam çizelge uzunluğu

Modelin formülasyonu aşağıdaki gibidir:

$$min\ C_{max} \tag{5.28}$$

$$x_{ikm} * (H_i + k - 1) * U \leq C_{max}, \forall (i,k,m) \in Set_1 \tag{5.29}$$

$$\sum_{i \in I} x_{ikm} \leq 1, \forall k \in K, m \in M \mid (i,k,m) \in Set_1 \tag{5.30}$$

$$\sum_{k \in K} \sum_{m \in M} x_{ikm} = 1, \forall i \in I \mid (i,k,m) \in Set_1 \tag{5.31}$$

$$x_{ikm} * (H_i + k - 1) \leq K_{max}, \forall (i,k,m) \in Set_1 \tag{5.32}$$

$$\begin{aligned}
&x_{tzn} \leq 1 - x_{ikm}, \forall (i,k,m),(t,z,n) \in Set_1 \mid \\
&F_0 \, AND \, (F_1 \, OR \, F_2 \, OR \, F_3) \\
&F_0 : t \neq i \, AND \, k \leq z \leq k + H_i - 1 \\
&F_1 : m = n, \\
&F_2 : m \leq n + O_{t_{max}} - 1 \leq m + O_{i_{max}} - 1, \\
&F_3 : n \leq m + O_{i_{max}} - 1 < n + O_{t_{max}} - 1.
\end{aligned} \tag{5.33}$$

$$C_{max} \geq 0 \tag{5.34}$$

$$x_{ikm} \in \{0,1\}, \forall (i,k,m) \in Set_1 \tag{5.35}$$

Amaç fonksiyonu toplam çizelge uzunluğunun (C_{max}) minimizasyonudur (5.28). Kısıt (5.29) ile her *m* makinesi için üretim süresi hesaplanır. Bu kısıt, işlerin başladıkları periyot ve işlerin periyot adetleri dikkate alınarak hesaplanır. Kısıt (5.30) ile her *m* makineye ve her *k* periyoda en fazla bir *i* işi atanabilir. Kısıt (5.31), bir *i* işinin yalnızca bir *m* makinesine ve bir *k* periyoduna atanmasını sağlar. Kısıt (5.32), toplam periyot uzunluğunu aşan işlerin son periyotlara atanmasını engeller. Kısıt (5.33) ile bir işin başladığı makine ve periyot dikkate alınarak bu işin operasyon adedi ($O_{i_{max}}$) kadar makineye ve periyot adedi (H_i) kadar periyoda başka bir işin atanması engellenir. Kısıt (5.34-5.35) ile değişkenlerin tanım uzayları belirlenir.

5.4.1. TP1 ve KTP1 karşılaştırması

Bölüm 5.2'de bahsedildiği gibi KTP1 modelinde işlerin bölünmesine izin verilir ve bölünme istenmediği durumlarda amaç fonksiyonunda katsayı değişimi ile modelin bunu engellemesi sağlanır. Diğer taraftan, TP1'de işlerin bölünmesi bir kısıtla engellenmiş ve işlerin tek seferde tamamlanması sağlanmıştır. Aslında, işlerin bölünmesi ile daha düşük

C_{max} değerli çizelgeler elde edilebilirken bu kabul, modeli karmaşık hale getirebilir ve gerçek boyutlu çizelgeleme problemleri için uygun çözümlere bile ulaşılamayabilir.

Bu bölümde, işlerin bölünmesi ve C_{max} arasındaki ilişkiyi incelemek için KTP1 ve TP1 modelleri kullanılmıştır. KTP1 amaç fonksiyonu kalıp değişim sürelerini barındırırken TP1'de kalıp değişim süreleri üretim sürelerine eklenerek hesaplamalara dahil edilmiştir. Modellerin performansları, kalıp değişim süresi (*R*) sıfır ve 0,5 saat kullanılarak karşılaştırılmıştır. Teoride en iyi çizelge KTP1 modelinde kalıp değişim süresinin sıfır olduğu (R=0), yani işlerin bölünmesine izin verildiği ve bunun bir bedeli olmadığı durumda elde edilir. Bu varsayımı test etmek amacıyla gerçek verilerden elde edilen 10 adet örnek problem (önceki bölümlerle aynı) kullanılmıştır. KTP1(R=0), KTP1(R=0,5) ve TP1(R=0,5) durumları için deneyler tekrarlanmış ve sonuçlar Çizelge 5.14'de özetlenmiştir. Çizelgede ilk üç sütun problemlerdeki işlerin sayısını ve çözüm için gereken minimum periyot adedini göstermekte ve üçüncü kısım karşılaştırılan üç farklı durum ile elde edilen C_{max} değerlerini göstermektedir. Dördüncü kısım, deneyler için çözüm sürelerini ve son kısım CPLEX tarafından hesaplanan %fark değerleri ile KTP1(*R*=0,5) ve TP1(*R*=0,5) arasındaki %EAOS değerini göstermektedir.

Çizelge 5.14. KTP1 ve TP1 modellerinin karşılaştırılması.

				C_{max}(s)		Çözüm süresi(sn)			%Fark			%EAOS
Problem	İşler	Periyotlar	KTP1 (R=0)	KTP1 (R=0,5)	TP1 (R=0,5)	KTP1 (R=0)	KTP1 (R=0,5)	TP1 (R=0,5)	KTP1 (R=0)	KTP1 (R=0,5)	TP1 (R=0,5)	KTP1(R=0,5) -TP1(R=0,5)
1	5	5	14,3	15,5	18,5	1,7	2,2	0	-	-	-	-19,4
2	10	8	30,3	30,8	30,3	6,5	9,9	0	-	-	-	1,7
3	15	13	42,3	44	46,8	12,4	389,6	2,2	-	-	-	9,1
4	20	22	84	91	92,1	4,1	1800	1,7	-	5,7	-	-1,1
5	25	25	92	100,5	100,1	4,7	1800	5,2	-	58,3	-	0,5
6	30	27	96	104,6	100,1	1800	1800	11,7	4,2	56,5	-	4,3
7	35	32	108	127,6	112,1	1800	1800	23,1	3,7	67,6	-	12,1
8	40	40	-	178,1	144,1	1800	1800	1800	-	85,6	40,9	19,1
9	45	49	-	238	182	1800	1800	1800	-	90,7	39,9	23,5
10	52	54	-	265,5	194,5	1800	1800	1800	-	93,3	43,9	26,7

KTP1 ile işlerin bölünmesine izin verilmiş ve kalıp değişim süreleri *R*=0 saat ve *R*=0,5 saat olarak alınmıştır. Çizelgede de görüldüğü gibi KTP1(R=0) ile en kısa çizelgeler bulunmuştur. *R*=0 durumunda kalıp değişim süreleri sıfır olarak kabul edilir ve bu kabulle KTP1 hızlanarak 25 işe kadar optimal çizelgeleri bulabilir hale gelir. Ancak, 35 işten sonra model fazla karmaşık hale gelir ve model hiçbir uygun çözüme ulaşamadan sonlanır. KTP1(R=0,5) durumunda işlerin bölünmesine izin verilir ancak kalıp değişim maliyetleri

amaç fonksiyonuna yansıtılmaktadır. TP1(R=0,5) durumunda yine kalıp değişim süreleri dikkate alınır ancak işlerin bölünmesine izin verilmez. Dolayısıyla işlerin bölünmesinin C_{max}'e etkisini incelemek için, KTP1(R=0,5) ve TP1(R=0,5)'in karşılaştırılması daha anlamlı olacaktır. Yapılan deneylerde KTP1(R=0,5) çözümleri incelendiğinde sadece ilk iki problem (10 iş) için hızlı çözümlere ulaşılabildiği, ve sadece ilk üç problem için (15 iş) optimal çözümlere ulaşılabildiği görülmektedir. Model, 20 iş ve fazlasını verilen süre limitlerinde optimal olarak çözememekte ve %fark değerlerine bakıldığında çözüm uzayını taramakta zorlanmaktadır. Dolayısıyla işlerin bölünmesine izin verilmesi problemi kompleks hale getirmektedir. Diğer taraftan, TP1(R=0,5) modeli ilk yedi problem için (35 iş) çok hızlı şekilde optimal çözümlere ulaşabilmiştir. Bu durum, blok atama mantığının kompleks ve çözülemeyen bir problemi çözülebilir hale getirdiğini göstermektedir. KTP1(R=0,5)'deki çizelgelerin daha kısa olmasının sebebi işlerin bölünmesine izin verilmesidir, ancak gerçek hayat problemlerinde 15 işten daha fazla işin olacağı ve bu durumda KTP1 modelinin kullanılamayacağı aşikardır. Bir diğer deyişle, işlerin bölünmesine izin verildiğinde çözümlerin optimalliği kaybedilmekte, işlerin bölünmesi engellendiğinde hızlı bir şekilde orta büyüklükte problemlerin optimal çözümü elde edilebilmekte ve büyük boyutlu problemlerde optimal sonuca yakın %farklarla uygun çözümler bulunabilmektedir. Bu sebeple, bu bölümden sonraki kısımlarda KTP1 modeli karşılaştırmalara dahil edilmeyecektir. TP1 performansı sonraki bölümde KP1 modeli ile karşılaştırılmış, matematiksel programlama ve kısıt programlama arasındaki farklar detaylandırılmıştır.

5.4.2. TP1 ve KP1 karşılaştırması

Bu bölümde TP1 ve KP1 modelleri performansları karşılaştırılmıştır. Anlamlı bir karşılaştırma yapabilmek için, iki modelin C_{max} hesaplarının tutarlı olması gerekir. TP1 ile işler periyot bitiminden önce tamamlansa da yeni iş atamaları periyot başlangıçlarında yapılır. Diğer taraftan, tasarımına bağlı olarak KP1 modeli periyotları dikkate almadan bir iş bitiminde diğer işi başlatabilir. Bu sebeple KP1 modeli TP1 ile aynı çizelgeyi üretse bile daha kısa C_{max} değeri hesaplamaktadır. Anlamlı bir karşılaştırma için aynı atamalara sahip bir çizelge için iki model tarafından hesaplanan C_{max} değerlerinin aynı

olması gerekir. Dolayısıyla, TP1 ve KP1 arasında doğru bir karşılaştırma yapabilmek için TP1 modelindeki kısıt (5.29) hesaplaması (5.29') ile değiştirilmiştir.

$$\sum_{(i,k,n)\in Set_1 | n\leq m, n+O_{i_{max}}-1\geq m} x_{ikn} * H_i * U \leq C_{max}, \forall m \in M \qquad (5.29')$$

TP1 ve KP1 modellerini karşılaştırmak için 10 problemden oluşan örnek kümesi kullanılmıştır. TP1 modeli yeni kısıtıyla kullanılmış ve 1800 saniye ile, KP1 modeli de 700 000 hata limiti ile sınırlandırılmıştır. Karşılaştırma sonuçları Çizelge 5.15'de detaylandırılmıştır. İlk üç sütun problem hakkında bilgi verirken, dördüncü kısım modeller tarafından hesaplanan C_{max} değerlerini göstermektedir. Beşinci kısım çözümler için gereken süreyi ve son kısım TP1 çözümünün optimalden uzaklığını gösteren %fark ile iki model arasındaki farkı gösteren %EAOS sütunlarını barındırmaktadır.

Çizelge 5.15. TP1 ve KP1 karşılaştırması

			C_{max}(s)		Çözüm Süresi(sn)		%Fark	%EAOS
Problem	İşler	Periyotlar	TP1	KP1	TP1	KP1	TP1	KP1-TP1
1	5	5	16,3	16,3	1,7	2,9	-	-
2	10	8	29,3	29,3	3,2	2,7	-	-
3	15	13	43,9	43,9	3,8	109,7	-	-
4	20	22	73,8	73,8	5,3	119,5	-	-
5	25	25	78,8	78,8	7,1	136,1	-	-
6	30	27	89,0	89,0	9,2	141,8	-	-
7	35	32	98,2	98,2	12,5	175,2	-	-
8	40	40	**121,9**	126,1	24,5	211,0	-	3,4
9	45	49	**156,3**	156,4	1080,0	305,7	-	0,1
10	52	54	**164,0**	170,0	325,7	371,2	-	3,5

Çizelge incelendiğinde TP1 modelinin süre limitleri içinde bütün problemlerin optimal çözümlerine ulaştığı görülmektedir. Çizelge 5.15'deki TP1 sonuçlarının Çizelge 5.14'den farklı çıkmasının sebebi, kısıt (5.29) hesaplamasının kısıt (5.29') ile değiştirilmiş olmasıdır. Bu değişiklik, aslında C_{max} hesabının basitleştirilmesinden ibarettir ve TP1'i oldukça hızlandırmıştır. TP1 modeli ile KP1 modeli amaç fonksiyonu değerleri Problem 7'ye kadar aynıdır, bu durum KP1 modelinin de optimal sonuçları yakaladığı ancak optimal ispatını yapamadığını gösterir. Problem 8'den itibaren TP1 modeli ile KP1 modeli arasındaki fark belirginleşmektedir. Çözüm süresi anlamında çoğu problem için TP1 modeli KP1'den daha kısa sürede optimal çözümlere ulaşabilmiştir. TP1 modeli problem

çözümlerini sadece ortalama % 2,3 oranında iyileştirse de sonuçlara daha kısa sürede ulaşmıştır ve bu çözümler optimaldir.

TP1 performansını büyük boyutlu problemlerle analiz etmek için gerçek verilerin dağılımı belirlenmiştir. Çizelge 5.3'deki düzgün dağılımlar kullanılarak iş adetleri 50-70 arasında değişen ve her iş büyüklüğü için üç farklı problem oluşturulmuştur. TP1 ve KP1 modelleri bu problemler ile test edilmiş ve sonuçlar Çizelge 5.16'da gösterilmiştir. İlk üç sütun problem hakkında bilgi vermektedir, sonraki kısımlar sırasıyla modeller tarafından bulunan çizelgelerin tamamlanma zamanlarını, TP1 çözümünün optimale ne kadar yaklaştığını ve TP1'in KP1'i ne kadar iyileştirdiğini göstermektedir. Bu deneylerde TP1 1800 saniye ile ve KP1 700 000 hata limiti ile sınırlanmıştır.

Çizelge 5.16. Düzgün dağılımlı veri ile TP1 ve KP1 karşılaştırması

			C_{max}(s)		Çözüm Süresi(sn)		%Fark	%EAOS
Problem	İşler	Periyotlar	TP1	KP1	TP1	KP1	TP1	KP1-TP1
1		66	226,1	240,1	1800	574,0	1,2	5,8
2	50	69	238,4	266,0	1800	554,9	2,0	10,4
3		66	221,3	240,0	1800	945,1	1,4	7,8
1		81	269,3	291,1	1800	622,3	0,6	7,5
2	60	78	262,1	287,1	1800	831,9	1,9	8,7
3		74	244,2	279,2	1800	351,9	2,1	12,5
1		88	296.0	319,3	1800	1400,5	0,2	7,3
2	70	90	299,6	320,2	1800	2561,6	1,0	6,4
3		90	268,4	298,3	1800	2120,8	0,2	10,0
						Ort.	1,2	8,5

Deney sonuçları incelendiğinde TP1'in çözümleri ortalama %8,5 iyileştirdiği ve bunu optimal çözüme ortalama %1,2 yakın çözümlerle elde ettiği görülmektedir. Ancak model bu çözümlere ulaşabilmek için 1800 saniye çalıştırılmalıdır. Bu süre bazı problemler için KP1'in süresinden daha az olsa da 30 dakika çözümler için fazla bir süredir. TP1'den daha iyi bir yöntem geliştirmek amacıyla kompleks çizelgeleme problemi atama ve sıralama olmak üzere iki probleme ayrıştırılmıştır. Bu ayrıştırma için klasik Benders ayrıştırma yönteminin kombinatoryal problemler için olan versiyonu; mantık-tabanlı Benders ayrıştırma yöntemi kullanılmıştır. Yöntemin detayları bir sonraki bölümde verilmiştir.

5.5. Mantık-Tabanlı Benders Ayrıştırma Modelleri

Pres hattı çizelgeleme problemindeki x_{ikm} karar değişkeni üç indisten oluşmaktadır. Bu indislerden biri (m) o işin hangi makinede işleneceğini temsil ederken, diğeri (k) hangi periyotta başlayacağını göstermektedir. Bu sebeple, aslında çözülmesi gereken iki problem vardır: Bunlardan ilki işlerin makinelere atanması problemidir ve ikincisi de işlerin atandıkları makinelerde sıralanması, bir diğer ifadeyle işlerin başlangıç zamanlarının belirlenmesi problemidir. Bu amaçla, mantık-tabanlı Benders ayrıştırma (MTBA) yöntemi kullanılarak kompleks çizelgeleme problemi iki probleme ayrıştırılabilir. Ana-problem bir atama problemidir ve sadece işleri ve makineleri barındırır. Ana-problem çözülerek en iyi iş-makine atamaları bulunur ve bu atamalar sabitlenerek alt-probleme gönderilir. Alt-problemin çözümü için geliştirilen yöntem, sabitlenen iş-makine atamalarında işlerin sırasını, yani işlerin bu makinelerdeki başlangıç zamanlarını belirler. Elde edilen sonuç değerlendirilir, eğer önceki sonuçlardan daha kötü bir sonuç elde edilmişse veya çözüm uygun değilse mevcut çözüm, ana-probleme eklenen kesim denklemi ile çözüm uzayından çıkarılır. İteratif olarak devam eden bilgi alışverişi ve kesimlerin eklenmesiyle en iyi sonuç bulunana kadar veya ana-problemin çözüm uzayının tamamı taranana kadar MTBA algoritması devam eder.

Literatürde MTBA yöntemini kullanan çalışmalar incelendiğinde yazarların genelde alt-problemi birden fazla parçaya ayırdığı görülmektedir. Örnek olarak, bir çizelgeleme probleminde her makine için bir alt-problem tanımlanabilir (toplamda makine sayısı kadar alt-problem olur) ve bu alt-problemle makineye atanan işlerin en iyi sırası belirlenebilir. Ancak, ele alınan pres hattı çizelgeleme probleminde bir iş birden fazla operasyona sahip olabilir ve bu operasyonlar ardışık makinelerde işlenmelidir; dolayısıyla makinelerdeki işlem süreleri birbirlerinden bağımsız olarak hesaplanamaz. Bu sebeple, pres hattı çizelgeleme probleminde her makine için bir alt-problem tanımlanmamış, bunun yerine tek alt-problemle bütün makineler dikkate alınarak sıralama problemi çözülmüştür.

Pres hattı çizelgeleme probleminde ana-problem atama problemidir ve problem çözümü için tamsayılı matematiksel bir model (TP2) geliştirilmiştir. TP2'de sadece işler

ve makineler dikkate alınmakta, işlerin başlangıç zamanları göz ardı edilmektedir. Dolayısıyla, ana-problem çözümü sonrasında iş-makine atamaları belirlenmiş olacaktır. TP2 modeli notasyonu aşağıdaki gibidir. *BR (Benders Returns)* kümesi, algoritma tarafından yeterince iyi olmadığına karar verilen TP2 atamalarını barındıran kümedir. Bu küme elemanları kullanılarak ana-probleme kesimler eklenir.

İndisler(kümeler aşağıda tanımlanmıştır):

i: Çizelgelenecek işler, $i \in I$.

m: Makineler, $m \in M$.

Parametreler:

A_{im}: Eğer *i* işi *m* makinesine atanabilirse A_{im}= 1 ve aksi halde sıfır.

J_i: *i* işinin toplam süresi.

Q_i: *i* işinin alternatif makine sayısı.

H_i: *i* işinin toplam periyodu; $\lceil (J_i/3600)/U \rceil$.

Kümeler:

I: İşler kümesi. *I*=$\{1,2,...,I_{max}\}$.

M: Makineler kümesi. *M*=$\{1,2,...,M_{max}\}$.

Set_2: $\{ (i,m) \mid i \in I, m \in M$: $A_{im} = 1\}$.

BR: $\{(id, x_{im}) \mid id \in \mathbb{Z}, (i,m) \in Set_2 \}$.

Karar Değişkenleri:

x_{im}= $\begin{cases} 1, \text{Eğer } i\text{.iş } m\text{.makineye atanırsa;} \\ 0, \text{Aksi halde} \end{cases}$

C_{max}: Toplam çizelge uzunluğu.

TP2 modeli kısıtları aşağıdaki gibidir:

$$min\, C_{max} \tag{5.36}$$

$$\sum_{(i,m) \in Set_2} x_{im} * \frac{J_i}{3600} \leq C_{max}, \forall m \in M \tag{5.37}$$

$$\sum_{m \in M} x_{im} = 1, \forall i \in I \mid (i,m) \in Set_2 \tag{5.38}$$

$$\sum_{(i,m)\in Set_2} (1 - x_{im}) \geq 1, \forall x_{im} \in BR \mid x_{im} > 0, Q_i > 1 \tag{5.39}$$

$$C_{max} \geq 0 \tag{5.40}$$

$$x_{im} \in \{0, 1\}, \forall (i, m) \in Set_2 \tag{5.41}$$

Ana-problemin amaç fonksiyonu maksimum tamamlanma zamanının minimizasyonudur (5.36). Kısıt (5.37) ile her makinede başlayan işlerin süreleri toplanarak makinelerin C_{max} değerleri bulunur, böylece en uzun tamamlanma zamanını veren makine ile toplam çizelge uzunluğu belirlenmiş olur. Kısıt (5.38) her işin sadece bir makineye atanmasını sağlar. Kısıt (5.39) Benders ayrıştırma denklemleridir ve bu kısıtlara *zayıf kesimler (weak cuts)* denir. Bu denklemlerle alt-problem tarafından *BR* kümesine aktarılan, uygun olmayan çözümleri barındıran atamalar, çözüm uzayından çıkarılır. Kısıt (5.39)'da var olan ve bir değerini alan x_{im} değişkenlerinden en az bir tanesinin sıfır olarak değişmesi bu denklemin sağlaması demektir ve önceki çözümün kombinasyonu değiştiğinden sadece o çözüm, çözüm uzayından çıkarılmış olur. Kısıt (5.40) ve (5.41) ise değişkenlerin tanım aralıklarını vermektedir.

Pres hattı çizelgeleme problemi için MTBA yöntemi kullanılarak iki farklı algoritma geliştirilmiştir. MTBA1 algoritmasında ana-problem için TP2 modeli kullanılır ve iş-makine atamaları yapılır. Alt-problem olarak kısıt programlama KP1 modeli kullanılarak işlerin makineler üzerindeki sıraları belirlenir. MTBA2 algoritması ise yine ana-problem olarak TP2 modelini kullanırken, alt-problem olarak TP1 modelinden yararlanır. Geliştirilen iki algoritma ilerleyen bölümlerde detaylandırılmıştır.

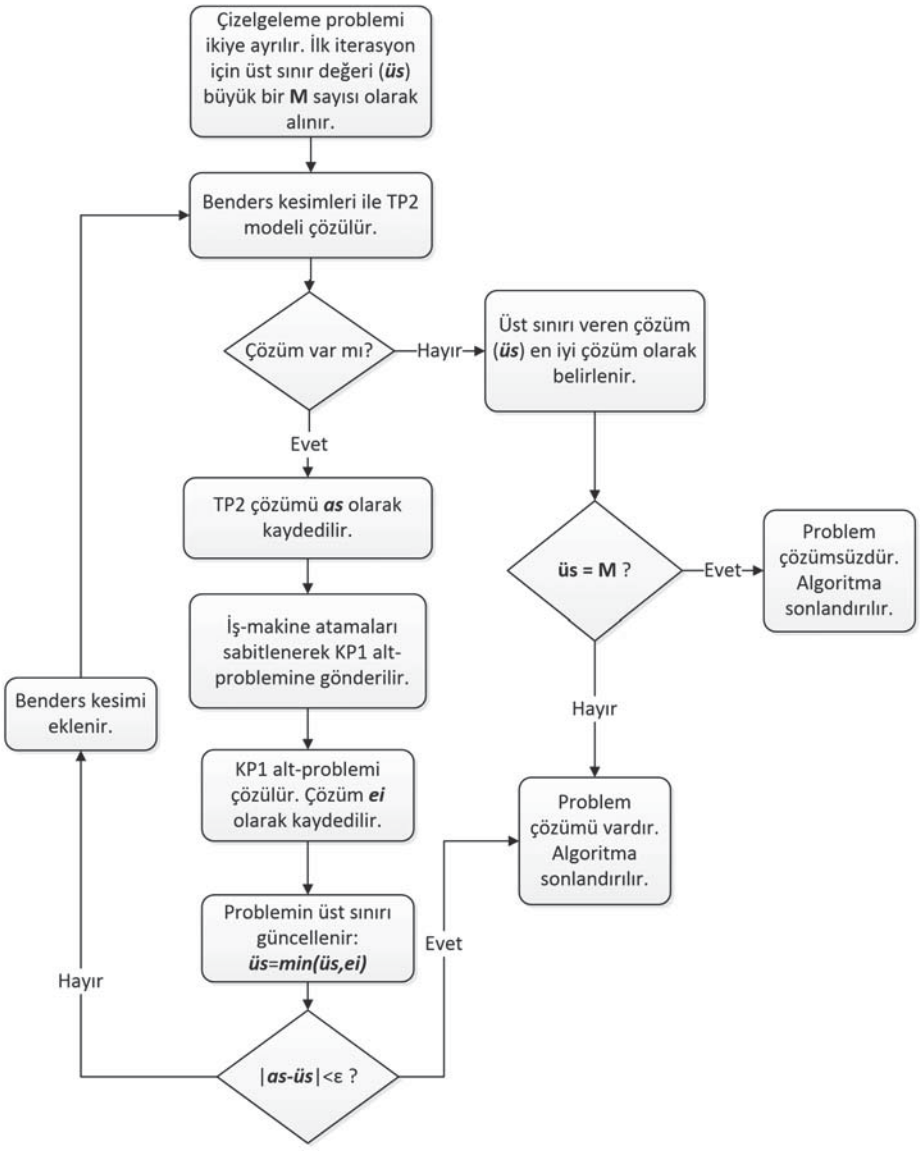
5.5.1. MTBA1 algoritması

MTBA1 algoritmasının amacı, matematiksel programlama ve kısıt programlamanın başarılı yönlerini birleştirmektir. Ana-problem olan atama problemi TP2 ile çözülmüş ve atamalar sabitlenerek KP1 modeline gönderilmiştir. KP1 modeli mevcut atamaların

başlangıç zamanlarını belirlemiştir. TP2 modeli ile saniyeler içinde işlerin makinelere atamaları yapılabilmektedir, çünkü problemdeki karar değişkenin zaman indisi çıkarılmış ve değişken boyutu iş*makine haline gelmiştir.

MTBA1 algoritmasının ilk iterasyonunda problemin *üst sınırı (üs)* büyük bir *M* sayısı olarak belirlenir. TP2 modeli kesim denklemleri olmadan çalıştırılır ve bulunan C_{max} değeri *alt sınır (as)* olarak kaydedilir. İşlerin tamamlanma süreleri dikkate alındığından, işlerin makinelerde sıralanmasına gerek kalmadan elde edilebilecek en iyi çözüm TP2 ile bulunur. Bu durum, 3-boyutlu bir problemin 2-boyutlu uzaya yansıtılması ve kümülatif olarak işlerin tamamlanma zamanlarının toplanarak C_{max} değerinin bulunması olarak düşünülebilir. TP2 çözümünden gelen C_{max} değeri problemin her zaman *alt sınırı (as)* olarak kaydedilir. TP2 tarafından bulunan iş-makine atamaları sabitlenerek KP1 modeline gönderilir, KP1 ile bu atamalara göre işlerin makineler üzerinde başlangıç zamanları belirlenir ve elde edilen çözüm *en iyi çözüm (ei)* olarak kaydedilir. Üst sınır, bulunan çözümle güncellenir, *üs=min(üs,ei)*; böylece en iyi üst sınırı veren çözüm saklanmış olur. Benders ayrıştırmada bu değerler, primal ve dual sistemlerin amaç fonksiyon değerleri olarak düşünülebilir. Teoride iki problemden elde edilen çözümlerin eşit olduğu noktada optimal kararı verilir. Ancak, MTBA1 algoritmasında kesim sayısını azaltmak ve algoritma çözüm süresini hızlandırmak için iki çözüm arasındaki farkın bir ϵ değerinden küçük olması yeterli olarak görülmüştür. Algoritma durdurma kriteri olarak *üs* ve *as* arasındaki farka bakılır. Eğer *üs* ve *as* arasındaki fark önceden belirlenen bir ϵ sayısından küçükse algoritma sonlandırılır, değilse Benders kesimi TP2 modeline eklenerek mevcut çözüm, çözüm uzayından çıkarılır ve TP2 ile yeni atamalar üretilir. MTBA1 algoritmasının akış şeması Şekil 5.4'deki gibi özetlenebilir. MTBA1 algoritmasında kullanılan KP1 modeli Bölüm 5.3'deki gibidir. Bu modele TP2 modelinden gelen sabitlenmiş atamalı temsilen bir parametre ve bir kısıt ilave edilmiştir. İlave olarak $SabitX_{im}$ parametresi tanımlanır ve kısıt (5.42) KP1 modeline eklenir.
$SabitX_{im}$: *i* işinin *m* makinesine atanması ile ilgili ana-problemden gelen ikili değerler.

$$(SabitX_a = 1) \overset{yields}{\rightarrow} (presenceOf(x_a) > 0), \forall a \in Alter \tag{5.42}$$



Şekil 5.4. MTBA1 akış şeması

MTBA1 algoritması farklı büyüklükte problemler kullanılarak TP1 ve KP1 modelleri ile karşılaştırılmıştır. Deneyler ilk önce gerçek verilerden elde edilen ve 10 adet problemden oluşan veri ile yapılmış ve deney sonuçları Çizelge 5.17'de detaylandırılmıştır. Çizelgede ilk üç sütun problemler hakkında bilgi vermektedir, sonraki kısımlar sırasıyla yöntemler tarafından bulunan çizelge uzunluklarını, bu çizelgeler için çözüm süresini ve yöntemlerin birbirinden farkını gösteren %EAOS hesaplamalarını göstermektedir. Son sütunda ise MTBA1 algoritmasının kaç kesimde çözüme ulaştığını gösteren kesim sayısıdır. MTBA1 ve TP1 1800 saniye ile KP1 700 000 hata limiti ile ve MTBA1 algoritmasındaki alt-problem KP1 de 700 000 hata limiti ile sınırlanmıştır. Durdurma kriteri 0,1 kabul edilmiş ve ana-problem ile alt-problem çözümü arasındaki fark 0,1 saatten veya 6 dakikadan az olması yeterli görülmüştür.

Çizelge 5.17. TP1, KP1 ve MTBA1 karşılaştırması

			C_{max}(saat)			Çözüm süresi(sn)			%EAOS		
Problem	İşler	Periyotlar	TP1	KP1	MTBA1	TP1	KP1	MTBA1	TP1-MTBA1	KP1-MTBA1	Kesim
1	5	5	16,3	16,3	16,3	1,7	2,9	4,1	0,0	0,0	4
2	10	8	29,3	29,3	29,3	3,2	2,7	1800	0,0	0,0	171
3	15	13	**43,9**	**43,9**	44,3	3,8	109,7	1800	-0,9	-0,9	16
4	20	22	73,8	73,8	73,8	5,3	119,5	474,7	0,0	0,0	3
5	25	25	78,8	78,8	78,8	7,1	136,1	120,9	0,0	0,0	7
6	30	27	**89,0**	**89,0**	93,5	9,2	141,8	1800	-5,0	0,0	15
7	35	32	**98,2**	**98,2**	104,6	12,5	175,2	1800	-6,5	0,0	13
8	40	40	**121,9**	126,1	135,3	24,5	211,0	1800	-11,0	0,0	10
9	45	49	**156,3**	156,4	177,5	1080,0	305,7	1800	-13,5	0,0	6
10	52	54	**164,0**	170,0	195,5	325,7	371,2	1800	-19,2	0,0	6
								Ort.	-5,6	-0,1	

Deney sonuçları incelendiğinde geliştirilen MTBA1 algoritmasının küçük boyutlu problemlerde TP1 ve KP1 kadar iyi sonuçlar ürettiği, ancak problem boyutu arttıkça çözüm kalitesinin kötüleştiği görülmektedir. Ayrıca, çözümlere ulaşabilmek için süre limitinin tamamı kullanılmaktadır. Bu durum, diğer modellerle arasında ciddi hız farklarının ortaya çıkmasına sebep olmaktadır. Algoritma kesimler üretmekte ve her kesim sonrasında KP1 algoritması aramaya baştan başlamaktadır. MTBA1 algoritmasının yavaşlığı daha büyük boyutlu problemlerde daha net anlaşılabilir. Bu amaçla Çizelge 5.3'deki düzgün dağılımlar kullanılarak üç farklı iş sayısı için üç farklı problem üretilmiş ve toplamda dokuz problem için Çizelge 5.18'deki karşılaştırmalar yapılmıştır.

Çizelge 5.18. Düzgün dağılımlı veri ile TP1, KP1 ve MTBA1 karşılaştırması

			Cmax (s)			Çözüm Süresi(sn)			%EAOS		
Problem	İşler	Periyotlar	TP1	KP1	MTBA1	TP1	KP1	MTBA1	TP1-MTBA1	KP1-MTBA1	Kesim
	1	66	226,1	240,1	226,4	1800	574,0	1800	-0,1	5,7	2
50	2	69	238,4	266,0	239,7	1800	554,9	1800	-0,5	9,9	2
	3	66	221,3	240,0	221,3	1800	945,1	1480,1	0,0	7,8	0
	1	81	269,3	291,1	276,5	1800	622,3	1800	-2,7	5,0	1
60	2	78	262,1	287,1	279,4	1800	831,9	1800	-6,6	2,7	1
	3	74	244,2	279,2	247,3	1800	351,9	1800	-1,3	11,4	3
	1	88	296,0	319,3	316,5	1800	1400,5	1800	-6,9	0,9	2
70	2	90	299,6	320,2	304,5	1800	2561,6	1800	-1,6	4,9	1
	3	90	268,4	298,3	279,4	1800	2120,8	1800	-4,1	6,4	1
								Ort.	2,7	6,1	

Çizelge 5.18'de de görüldüğü gibi MTBA1 algoritması çözümlere ulaşmak için kesimler üretmektedir, ancak süre limiti içinde fazla kesim yapamamaktadır. MTBA1 ile TP1 çözümleri ortalama %2,7 oranında kötüleşmekte, ancak KP1 çözümleri %6,1 oranında iyileşmektedir. Büyük boyutlu problemlerde KP1'den daha iyi sonuçlara ulaşabilse de bu sonuçlar için KP1'den daha uzun çözüm süresi gerekmektedir. Ayrıca kesim denklemlerinin zayıf olması çözüme ulaşmayı güçleştirmektedir. Bu sebeple, sonraki bölümde de detaylandırılan yeni kesim denklemleri araştırılmıştır.

5.5.2. Yeni kesimlerin geliştirilmesi

MTBA1 algoritması test edildikten sonra mevcut kesimlerin çok zayıf kaldığı ve bu durumun başarılı çözümlere ulaşmayı engellediği görülmüştür. Kesim denklemi (5.39), sadece mevcut çözümde bir değerini alan ve alternatif makine sayısı birden büyük olan değişkenleri *BR* kümesine eklemekte ve bu değişkenlerle kısıtı oluşturmaktadır. Bu denklemin geçerli olması için değişkenlerden en az bir tanesinin sıfır yapılması yeterli olmakta ve küçük bir değişiklikle kısıt sağlanmaktadır. Sonuç olarak, bu kesimler çok zayıftır ve başarılı çözümlere ulaşmayı engellemektedir. MTBA1 algoritmasını iyileştirebilmek amacıyla daha güçlü ve etkin kesimler geliştirilmiş ve performansları karşılaştırılmıştır.

$$\sum_{(i,m)\in Set_2} (1 - x_{im}) \geq 1, \forall x_{im} \in BR \mid x_{im} > 0, Q_i > 1 \tag{5.39}$$

KP1 atamaların yapısı incelendiğinde süre ve operasyon sayısı bakımından büyük hacimli işlerin atamayı zorlaştırdığı gözlenmiştir. Bu sebeple, bir eşik değeri *(ϵ)* ve işlerin boyutları *(E_i)* kullanılarak büyük boyutlu işlerin tespit edilmesi ve bu işlerin bir önceki uygun çizelgeden çıkarılarak kesim denkleminin oluşturulması düşünülmüştür. Öncelikle alt-problem KP1, ana-problem TP2'den gelen atamalarla çözüme başlar ve bir çizelge belirler. Eğer üretilen çizelgenin amaç fonksiyonu değeri istenen şartları sağlamıyorsa kesim denklemi üretilir. Bunun için önceden belirlenen bir *(ϵ)* sayısı, her iş için hesaplanan *(E_i)* ile karşılaştırılır. Alt-problem, $E_i > \epsilon$ olan işleri ***BR*** kümesine atar, TP2 sonraki iterasyonda ***BR*** kümesindeki işlerden birini sıfır yapar, bu şekilde hem önceki uygun çözüm uzaydan çıkarılmış olur, hem de büyük hacimli bir işin sıfır yapılması daha çok x_{im} değişkeninin değişmesini sağlayacağından bu kesim, öncekilere göre daha etkin olur. İşlerin boyutlarını gösteren *(E_i)* şu şekilde hesaplanır:

$$E_i = O_{i_{max}} * H_i, \forall i \in I \tag{5.43}$$

İşlerin hacimleri *(E_i)* hesaplandıktan sonra KP1, $E_i > \epsilon$ eşitsizliğine göre ***BR*** kümesini oluşturur. ***BR*** kümesindeki işler için kısıt (5.39') yazılır.

$$\sum_{(i,m)\in Set_2} (1 - x_{im}) \geq 1, \forall x_{im} \in BR \mid x_{im} \geq 0, Q_i \geq 1, E_i \geq \epsilon \tag{5.39'}$$

Kısıttaki ϵ değeri kullanıcı tarafından sezgisel olarak belirlenir. Mevcut problem verileri incelenir ve çoğunlukla hangi büyüklükte işlerin olduğu gözlenir. Örneğin, 4 operasyona sahip 3 periyot süren bir işin E_i değeri 12 olacaktır. Bu iş, planlama döneminde önemli bir yere sahiptir. Bu işin yerinin değişmesi, 12'den küçük boyutlu işleri de etkileyeceğinden ana-problemin amaç fonksiyonunun belli bir değere takılıp kalması önlenebilir.

Burada ϵ değerinin çözüm kalitesini etkileyip etkilemediği araştırılmalıdır. Bunun için deneyler $\epsilon = 8$ ve $\epsilon = 12$ değerleri için tekrarlanmıştır. Deney sonuçları Çizelge 5.19'da ve Çizelge 5.20'de özetlenmiştir. Karşılaştırılan yöntemlerin hepsi 1800 saniye süre limiti ile sınırlanmıştır. Çizelge 5.19'da TP1, KP1, MTBA1 zayıf kesimler, MTBA1_$\epsilon = 8$

ve MTBA1_ϵ $=$ 12 modelleri ile elde edilen C_{max} değerleri karşılaştırılmıştır. Deney sonuçları incelendiğinde orta ve büyük boyutlu problemlerde TP1 ve KP1 yöntemlerinin daha başarılı olduğu görülmektedir. Diğer taraftan MTBA1 algoritmaları kendi aralarında değerlendirildiğinde genelde MTBA1 zayıf kesimler, MTBA1_$\epsilon = 8$ ve MTBA1_$\epsilon = 12$ yöntemlerinin aynı sonucu buldukları, ancak MTBA1_$\epsilon = 8$'in son problemde daha iyi bir çözüme ulaştığı görülmektedir.

Çizelge 5.19. Güçlü kesimlerin geliştirilmesi: Hacimli işler ve C_{max} değerleri

			C_{max}(saat)				
Problem	İşler	Periyotlar	TP1	KP1	MTBA1	MTBA1_$\epsilon = 8$	MTBA1_$\epsilon = 12$
1	5	5	16,3	16,3	16,3	16,3	16,3
2	10	8	**29,3**	**29,3**	**29,3**	32,0	32,0
3	15	13	**43,9**	**43,9**	44,3	44,3	44,3
4	20	22	73,8	73,8	73,8	73,8	73,8
5	25	25	78,8	78,8	78,8	78,8	78,8
6	30	27	**89,0**	**89,0**	93,5	89,3	89,3
7	35	32	**98,2**	**98,2**	104,6	104,0	104,6
8	40	40	**121,9**	**121,9**	135,3	135,3	135,3
9	45	49	**156,3**	**156,3**	168,2	168,2	168,2
10	52	54	**164,0**	170,0	184,9	178,9	184,0

Çizelge 5.20'de detaylandırılan verilerden MTBA1 yöntemlerinin benzer sürelerde ve benzer kesim adetlerinde çözümlere ulaştığı görülmektedir. En büyük farklılık ikinci problemde MTBA1'in 171 kesim yapması ve MTBA1_$\epsilon = 8$ ve MTBA1_$\epsilon = 12$'nin 44 kesim yapmasıdır. Bu farklılığa rağmen bütün yöntemler 1800 saniye çalışmıştır.

Çizelge 5.20. Güçlü kesimlerin geliştirilmesi: Hacimli işler, çözüm süreleri ve kesimler

			Çözüm Süresi(sn)			Kesimler		
Problem	İşler	Periyotlar	MTBA1	MTBA1_$\epsilon = 8$	MTBA1_$\epsilon = 12$	MTBA1	MTBA1_$\epsilon = 8$	MTBA1_$\epsilon = 12$
1	5	5	4,1	9,5	12	4	3	3
2	10	8	1800	1800	1800	171	44	44
3	15	13	1800	1800	1800	16	15	14
4	20	22	474,7	387,0	383,1	3	3	3
5	25	25	120,9	897,6	893,4	7	7	7
6	30	27	1800	1800	1800	15	14	14
7	35	32	1800	1800	1800	13	14	14
8	40	40	1800	1800	1800	10	13	14
9	45	49	1800	1800	1800	6	14	13
10	52	54	1800	1800	1800	6	13	13

Bu deneyler sonucunda büyük hacimli işlerin çözüm kalitesini arttıramadığı söylenebilir, bu sebeple yeni güçlü kesim arayışına gidilmiştir. Amaç fonksiyonu değeri maksimum tamamlanma zamanıdır. Maksimum tamamlanma zamanını veren makineler ve buradaki

değiştirilebilecek işler belirlenebilirse, bu işlerin daha düzgün bir şekilde makinelere dağıtılmasıyla amaç fonksiyonu iyileştirilebilir. Bunun için alt-problemde maksimum tamamlanma zamanı C_{max} değeri maksimum kapasite *maxkap* olarak tanımlanmış ve alt-problem bitiminde ilave algoritmayla her makinenin kapasitesi belirlenmiştir. Bu durumda, ana-problemin sabitlediği atamalar alt-probleme verilir ve alt-problem yeterince iyi bir çizelge üretemezse kesim denklemi oluşturmak üzere *maxkap* tespit edilir. Daha sonra kapasitesi *maxkap*'a eşit olan makineler bulunur. Bu makinelerde, alternatif makine sayısı birden büyük olan değişkenler *BR* kümesine eklenir. Burada çok güçlü kesimlerden kaçınmak için *BR* kümesine eklenen değişkenlerin birden fazla olması istenmiştir. Eğer sadece bir değişken eklenebiliyorsa bu durumda kapasitesi *maxkap/1,25* olan makinelere bakılır ve buradaki işlerden alternatif makine sayısı birden fazla olanlar *BR* kümesine eklenir. Bu şekilde geliştirilen MTBA1_maxkap algoritması sonuçları önceki kesimlerle birlikte Çizelge 5.21'de ve Çizelge 5.22'de özetlenmiştir. Çizelge 5.21 incelendiğinde MTBA1_maxkap ile elde edilen C_{max} değerleri diğer MTBA1 yöntemlerine eşit veya daha iyidir. Ayrıca, elde edilen çizelge uzunlukları TP1 ve KP1 ile elde edilenlere daha yakındır. MTBA1 algoritmaları arasındaki fark özellikle yedinci problemden sonra belirginleşmekte ve MTBA1_maxkap ile daha başarılı sonuçlar elde edilmektedir.

Çizelge 5.21. MTBA1_maxkap yöntemi C_{max} değerlerinin diğer yöntemlerle karşılaştırması

			C_{max}(saat)					
Problem	İşler	Periyotlar	TP1	KP1_1800	MTBA1	MTBA1_$\epsilon = 8$	MTBA1_$\epsilon = 12$	MTBA1_maxkap
1	5	5	16,3	16,3	16,3	16,3	16,3	16,3
2	10	8	**29,3**	**29,3**	**29,3**	32,0	32,0	**29,3**
3	15	13	**43,9**	**43,9**	44,3	44,3	44,3	**43,9**
4	20	22	73,8	73,8	73,8	73,8	73,8	73,8
5	25	25	78,8	78,8	78,8	78,8	78,8	78,8
6	30	27	**89,0**	**89,0**	93,5	89,3	89,3	**89,0**
7	35	32	**98,2**	**98,2**	104,6	104,0	104,6	100,4
8	40	40	**121,9**	**121,9**	135,3	135,3	135,3	127,6
9	45	49	**156,3**	**156,3**	168,2	168,2	168,2	164,0
10	52	54	**164,0**	170,0	184,9	178,9	184,0	178,8

Çizelge 5.22'de de görüldüğü gibi MTBA1_maxkap bazı problemlerde diğer MTBA1 yöntemlerinden hızlı olsa da orta ve büyük boyutlu problemlerde süre limitine takılmıştır. Küçük boyutlu problemlerde daha az kesimle aynı sonuçları elde edebilmesine rağmen, orta ve büyük boyutlu problemlerde ise benzer sayıda kesim üretmiştir.

Çizelge 5.22. MTBA1_maxkap yöntemi çözüm süreleri ve kesimlerin diğer yöntemlerle karşılaştırması

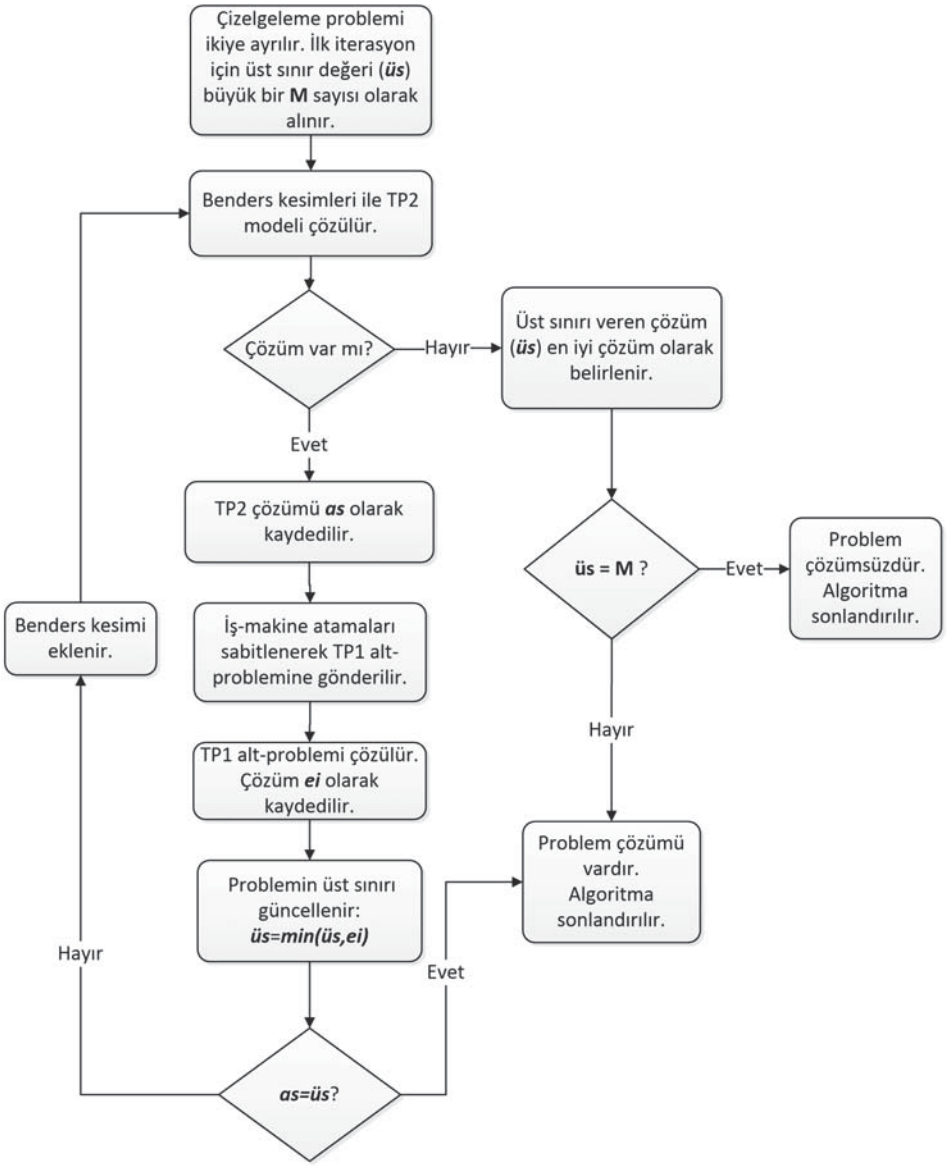
Problem	İşler	Periyotlar	Çözüm Süresi(sn)				Kesimler			
			MTBA1	MTBA1_ $\epsilon = 8$	MTBA1_ $\epsilon = 12$	MTBA1_ maxkap	MTBA1	MTBA1_ $\epsilon = 8$	MTBA1_ $\epsilon = 12$	MTBA1_ maxkap
1	5	5	4,1	9,5	12	8	4	3	3	2
2	10	8	1800	1800	1800	126	171	44	44	3
3	15	13	1800	1800	1800	255,4	16	15	14	2
4	20	22	474,7	387,0	383,1	256,5	3	3	3	2
5	25	25	120,9	897,6	893,4	259,3	7	7	7	2
6	30	27	1800	1800	1800	1611,2	15	14	14	13
7	35	32	1800	1800	1800	1800	13	14	14	14
8	40	40	1800	1800	1800	1800	10	13	14	13
9	45	49	1800	1800	1800	1800	6	14	13	14
10	52	54	1800	1800	1800	1800	6	13	13	13

Çizelgeler incelendiğinde MTBA1_maxkap ile diğer MTBA1 yöntemlerinden daha başarılı sonuçlar, daha hızlı şekilde elde edilebilmiştir. Ancak, elde edilen sonuçlar TP1 ve KP1 ile elde edilenler kadar iyi değildir. Bunun sebebi, MTBA1 algoritmasının performansının sezgisel olarak belirlenen bir hata limitine bağlı olmasıdır. Alt-problem KP1'in durdurma kriteri olarak kullanılan limit ne kadar yüksek belirlenirse elde edilen çözüm kalitesi artmakta, ancak çözüm süresi de uzamaktadır. Geliştirilen farklı kesimler MTBA1 yöntemini hızlandırsa da yeterli olmamıştır. Bu sebeple alt-problem olarak kullanılacak alternatif yöntemler araştırılmıştır. Ana-problemin tamsayılı matematiksel model ile çok hızlı şekilde çözülmesi, alt-problem olarak da tamsayılı matematiksel modelin kullanılabileceği fikrini doğurmuştur. Bu sebeple ikinci bir MTBA algoritması, ana-problem ve alt-problemlerde tamsayılı matematiksel modellerin kullanılmasıyla geliştirilmiştir. Bu şekilde geliştirilen MTBA2 algoritması ve performans değerlendirmeleri sonraki bölümde detaylandırılmıştır.

5.5.3. MTBA2 algoritması

MTBA2 algoritması ana-problem ve alt-problem için matematiksel programlama modelleri kullanılarak geliştirilmiştir. Bu sayede alt-problem sezgisel olmaktan kurtulmakta ve her iki problem için optimal bilgisi elde edilebilmektedir. MTBA2 yönteminin ilk iterasyonunda problemin *üst sınırı (üs)* olarak büyük bir *M* sayısı belirlenir. TP2 modeli Benders kesimleri olmadan çalıştırılır ve ilk çözüm C_{max} değerinin *alt sınırı (as)* olarak kaydedilir. Bulunan iş-makine atamaları sabitlenerek TP1 modeline

gönderilir, TP1 bu atamalara göre işlerin makineler üzerinde başlangıç periyotlarını belirleyerek çözümünü üretir ve *en iyi çözüm (ei)* kaydedilir. Üst sınır bulunan çözümle güncellenir, *üs=min(üs,ei)*, böylece en iyi üst sınırı veren çözüm saklanmış olur. Eğer *üs* and *as* arasındaki fark sıfır ise algoritma sonlandırılır, değilse Benders kesimi TP2 modeline eklenerek mevcut çözüm, çözüm uzayından çıkarılır ve TP2 ile yeni atamalar üretilir. Algoritma akış şeması Şekil 5.5'deki gibidir. MTBA2 algoritması şimdiye



Şekil 5.5. MTBA2 akış şeması

kadar en başarılı sonuçları üreten TP1 modeli ile en başarılı MTBA1 algoritması olan MTBA1_maxkap ile karşılaştırılmıştır. Deneyler için gerçek verilerden elde edilen problemler kullanılmış ve sonuçlar Çizelge 5.23'de gösterilmiştir.

Çizelge 5.23. MTBA2 algoritmasının diğer yöntemlerle karşılaştırması

			C_{max}(saat)			Çözüm süresi			Kesimler	
Problem	İşler	Periyotlar	TP1	MTBA1_maxkap	MTBA2	TP1	MTBA1_maxkap	MTBA2	MTBA1_maxkap	MTBA2
1	5	5	16,3	16,3	16,3	1,7	8	4,5	2	3
2	10	8	29,3	29,3	29,3	3,2	126	500,5	3	216
3	15	13	43,9	43,9	43,9	3,8	255,4	323,3	2	144
4	20	22	73,8	73,8	73,8	5,3	256,5	13,3	2	3
5	25	25	78,8	78,8	78,8	7,1	259,3	32,0	2	7
6	30	27	89,0	89,0	89,3	9,2	1611,2	1800	13	299
7	35	32	98,2	100,4	100,4	12,5	1800	1800	14	98
8	40	40	121,9	127,6	135,3	24,5	1800	1800	13	8
9	45	49	156,3	164,0	168,2	1080,0	1800	1800	14	124
10	52	54	164,0	178,8	216	325,7	1800	1800	13	21

Deney sonuçları incelendiğinde MTBA2 yönteminin çözümlere ulaşmak için MTBA1_maxkap yönteminden oldukça fazla kesimlere ihtiyaç duyduğu görülmektedir. Ayrıca, bu kesimlere rağmen büyük problemlerde MTBA2 algoritması ile elde edilen çözümler MTBA1_maxkap çözümlerinden daha kötüdür. Kesimlerin sebebi incelendiğinde ana-problem ve alt-problemdeki C_{max} hesaplamalarının farklı olduğu görülmüştür. Bu sebeple, iki problem aynı çözümü bile bulsa amaç fonksiyonu değerleri farklı olacağından yine MTBA2 yine kesim yapılmaktadır. En iyi amaç fonksiyonu değeri MTBA2 tarafından kayıt altına alınsa da gereksiz yapılan kesimler algoritma süresini uzatmaktadır.

MTBA2 algoritması detayları için her iki modelde kullanılan toplam çizelge uzunlukları hesaplaması incelenmiştir. TP1 modeli C_{max} hesabı aşağıdaki kısıt (5.44) gibi ve TP2 hesabı ise kısıt (5.37) gibidir.

$$\sum_{(i,k,n)\in Set_1 | n\leq m, n+O_{i_{max}}-1\geq m} x_{ikn} * H_i * U \leq C_{max}, \forall m \in M \tag{5.44}$$

TP2 modeli, kısıt (5.37) ile her makinede başlayan işlerin sürelerini toplar, ve bu toplamda önceki makinelerde başlayıp hesaplama yapılan makineye kadar işleri, yani işlerin operasyon adetlerini dikkate almaz. Dolayısıyla, hesaplamalar gerçek çizelge uzunluklarını yansıtmaz. Bu toplam, Şekil 5.3 ile açıklanabilir. Şekilde de görüldüğü gibi

sadece makinelerde başlayan işlerin süreleri dikkate alınırsa *B* ve *F* işlerini barındıran gerçek çizelge uzunluğu C_{max} hesaplanamaz. Çizelge 5.24'de görüldüğü gibi kısıt (5.37) ve kısıt (5.44) ile her makinedeki işlerin toplam süreleri hesaplanmıştır. Çizelgede ilk sütun ilgili makineleri temsil ederken ikinci ve üçüncü sütunlar incelenen kısıtlarla hesaplanan C_{max} değerlerini göstermektedir. Bu sütunlardaki sıfır değerleri, işlerin operasyon adetlerinin dikkate alınmamasından kaynaklanır. Bu hesaplamalar sonunda TP1 modeli ile C_{max} değeri J_B+J_F olarak belirlenirken kısıt (5.37) ile TP2'de bulunan çizelge uzunluğu J_A+J_D olmaktadır. Bu sonucun gerçek çizelge uzunluğunu yansıtmadığı aşikardır.

Çizelge 5.24. Kısıt (5.44) ve (5.37) karşılaştırması

	C_{max}	
	Kısıt (5.44)	Kısıt (5.37)
M1	J_A+J_D	J_A+J_D
M2	J_A+J_D	0
M3	J_A+J_D	0
M4	J_A+J_D	0
M5	J_A+J_D	0
M6	J_A+J_D	J_B
M7	J_B	0
M8	J_B	0
M9	J_B+J_F	J_F
M10	J_B+J_F	0
M11	J_C+J_F	J_C
M12	J_C+J_F	0
M13	J_C+J_E	J_E

Bu sorunu düzeltmek amacıyla TP1 modeli C_{max} hesabı kısıt (5.44), TP2'deki kısıt (5.37) yerine kullanılmıştır. Kısıt (5.44) hesaplaması periyotlardan bağımsızdır ve işlerin operasyon adetlerini dikkate alır. Bu şekilde mantıklı bir karşılaştırma yapılabilir. Değişiklik sonrasında MTBA2 yönteminin hiç kesim yapmadığı tespit edilmiştir. Bunun sebebi araştırıldığında yapılan değişiklikle TP2 modelinde üretilen optimal çözümün aslında TP1 modelindeki bütün kısıtları sağladığı ve ayrıca bu çözümün TP1 modelinin de optimal çözümü olduğu görülmüştür. Bu sebeple, yapılan C_{max} hesabı değişikliğinden sonra geliştirilen yöntem yeni bir isimle (TP2/TP1 yöntemi) adlandırılmış ve bir sonraki bölümde detaylandırılmıştır.

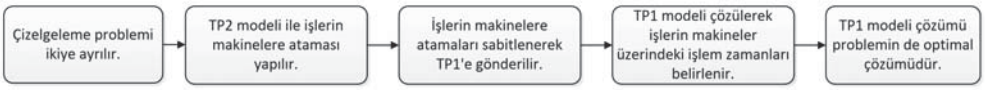
5.6. TP2/TP1 Yöntemi

MTBA algoritmaları ile çizelgeleme problemi ana problem (TP2) ve alt-problem (TP1 ve KP1) olarak iki kısıma ayrılmış, ana problemin bulduğu atamalar alt-probleme sabitlenerek verilmiş ve alt-problem ile işlerin atandıkları makinelerde başlangıç süreleri belirlenmiştir. MTBA modelleri kesimler yaparak ilerlemiş ve uygun çözümlere ulaşabilmiştir. Ancak, yapılan kesimler algoritmaların çözüm sürelerini olumsuz etkilemiş, gereksiz kesimler sebebiyle algoritmalar beklenen performansı sergileyememiştir.

Bu kesimlerin sebepleri araştırıldığında ana-problem TP2'deki amaç fonksiyonu hesaplamasında operasyon adetlerinin dikkate alınmamasının, gereksiz kesimlere sebep olduğu görülmüştür. Ana-problemde çizelge uzunluğu hesaplamasında sadece makinelerde başlayan işler dikkate alınırken, alt-problemde ilave olarak başka makinelerde başlayan ve ilgili makinede de devam eden işler de hesaplamalara dahil edilmiştir. Karşılaştırmaların sağlıklı yapılabilmesi için, Bölüm 5.5'de de bahsedildiği gibi amaç fonksiyonu hesaplaması ana-problemde de güncellenmiştir.

Değişiklik sonrasında MTBA2 yönteminin hiç kesim yapmadığı tespit edilmiştir. Bunun sebebi araştırıldığında yapılan değişiklikle TP2 modelinde üretilen optimal çözümün aslında TP1 modelindeki bütün kısıtları sağladığı ve ayrıca bu çözümün TP1 modelinin de optimal çözümü olduğu görülmüştür. Bu sebeple, yapılan C_{max} hesabı değişikliğinden sonra geliştirilen yöntem yeni bir isimle (TP2/TP1 yöntemi) adlandırılmıştır.

TP2/TP1 metodunda TP2 ve TP1 modelleri bazı değişikliklerle birlikte kullanılmıştır. TP1 alt-problemine TP2'den gelen atamaların sabitlendiği bir kısıt eklenmiş ve TP2 modelinden Benders kesim denklemleri çıkarılmıştır. Metodun optimal çözümü sağladığı ilerleyen kısımda ispatlanmıştır. Yöntemin akışı Şekil 5.6'daki gibidir.



Şekil 5.6. TP2/TP1 akış şeması

TP1 modelinin orijinal TP1 modelinden iki farkı vardır. Bunlardan birincisi, toplam çizelge uzunluğu C_{max} hesaplamasıdır. İkinci fark ise TP2 çözümünün TP1 modeline eklenmesini sağlayan kısıt (5.51)'dir. İspatın takibinin kolaylıkla yapılabilmesi için TP1 modeli güncellenmiş haliyle aşağıdaki kısıtlardan oluşur. Notasyon Bölüm 5.4'deki gibidir.

$$min\ C_{max} \tag{5.45}$$

$$\sum_{(i,k,n)\in Set_1 | n\leq m, n+O_{i_{max}}-1\geq m} x_{ikn} * H_i * U \leq C_{max}, \forall m \in M \tag{5.46}$$

$$\sum_{i\in I} x_{ikm} \leq 1, \forall k \in K, m \in M \mid (i,k,m) \in Set_1 \tag{5.47}$$

$$\sum_{k\in K}\sum_{m\in M} x_{ikm} = 1, \forall i \in I \mid (i,k,m) \in Set_1 \tag{5.48}$$

$$x_{ikm} * (H_i + k - 1) \leq K_{max}, \forall (i,k,m) \in Set_1 \tag{5.49}$$

$$\begin{aligned}
&x_{tzn} \leq 1 - x_{ikm}, \forall (i,k,m),(t,z,n) \in Set_1 \mid \\
&F_0\, AND\, (F_1\, OR\, F_2\, OR\, F_3) \\
&F_0 : t \neq i\, AND\, k \leq z \leq k + H_i - 1 \\
&F_1 : m = n, \\
&F_2 : m \leq n + O_{t_{max}} - 1 \leq m + O_{i_{max}} - 1, \\
&F_3 : n \leq m + O_{i_{max}} - 1 < n + O_{t_{max}} - 1.
\end{aligned} \tag{5.50}$$

$$\sum_{k\in K} x_{ikm} = x^*_{im}, \forall (i,m) \in Set_2 \tag{5.51}$$

$$C_{max} \geq 0 \tag{5.52}$$

$$x_{ikm} \in \{0,1\}, \forall (i,k,m) \in Set_1 \tag{5.53}$$

Amaç fonksiyonu toplam çizelge uzunluğunun (C_{max}) minimizasyonudur (5.45). Her makinedeki işlerin süreleri kısıt (5.46) ile hesaplanır. Bu kısıt, *m*. makinede başlayan veya önceki herhangi bir *n*. makinede başlayıp *m*. makineye kadar devam eden işleri kapsar. Kısıt (5.47) ile her *m* makineye ve her *k* periyoda en fazla bir *i* işi atanabilir. Kısıt (5.48) *i* işinin yalnızca bir makine ve bir periyoda atanmasını garantiler. Kısıt (5.49), toplam periyot uzunluğunu aşan işlerin son periyotlara atanmasını engeller. Kısıt (5.50) ile bir işin başladığı makine ve periyot dikkate alınarak bu işin operasyon adedi ($O_{i_{max}}$) kadar makineye ve periyot adedi (H_i) kadar periyoda başka bir işin atanması engellenir. Kısıt (5.51) ile ana-problemde sabitlenen iş-makine atamaları alt-probleme aktarılmış olur. Hangi işin hangi makinede yapılacağı bellidir, ancak periyoda göre toplam alındığından sadece bir periyotta yapılabilir. Bu periyot alt-problem TP1 tarafından belirlenerek iş sıralama gerçekleştirilir. Kısıt (5.52-5.53) ile değişkenlerin tanım uzayları belirlenir.

TP2 modeli TP1 modelinden Benders kesim denklemlerinin çıkarılması ile oluşturulmuştur. Model notasyonu bölüm 5.5'deki ve kısıtlar aşağıdaki gibidir.

$$min\ C_{max} \tag{5.54}$$

$$\sum_{(i,n) \in Set_2 | n \leq m, n+O_{i_{max}}-1 \geq m} x_{in} * H_i * U \leq C_{max}, \forall m \in M \tag{5.55}$$

$$\sum_{m \in M} x_{im} = 1, \forall i \in I \mid (i,m) \in Set_2 \tag{5.56}$$

$$C_{max} \geq 0 \tag{5.57}$$

$$x_{im} \in \{0,1\}, \forall (i,m) \in Set_2 \tag{5.58}$$

TP2 modelinin amaç fonksiyonu toplam çizelge uzunluğunun minimizasyonudur. Kısıt (5.56) ile her iş alternatif makinelerinden yalnızca bir tanesine atanabilir. Son iki kısıt, karar değişkenlerinin tanımını içerir. TP2/TP1 yöntemi öncelikle TP2 modelini çözer ve x^*_{im} değişkenlerinin optimal değerini belirler. Daha sonra bu atamalar TP1 modeline (5.51) ile gönderilir. TP1 modeli sabitlenmiş iş-makine atamaları ile çözüme başlar ve bu işlerin makineler üzerinde başlama zamanlarını belirler. TP2/TP1 yönteminin optimal çözüm bulduğu aşağıda gösterilmiştir.

Lemma. *TP1 alt-problemindeki optimal iş-makine atamaları doğrudan TP2 modelinin optimal çözümünden elde edilir.*

İspat. Aşağıdaki eşitlik tanımlansın.

$$\sum_{k \in K} \overline{x}_{ikm} = x^*_{im}, \forall (i,m) \in Set_2 \tag{5.59}$$

İlk önce $\overline{x}_{ikm}$ 'nin TP1 modelinin bir uygun çözümü olduğu gösterilecek ve sonrasında bu çözümün aynı zamanda optimal olduğu gösterilecektir. TP2'deki kısıt (5.56) ile $x^*_{im} = 1$ eşitliğini sağlayacak her i için yalnızca bir m (tersi de geçerli) değeri vardır. Böylece (5.59)'da her (i,m) çifti için $\overline{x}_{ikm} = 1$ değerini veren sadece bir *k* vardır. $x^*_{in} = 1$ atamaları (5.55)'i sağlar ve (5.46) denklemi hesabında yalnızca bir *k* değeri için $x^*_{ikn} = 1$ olduğundan $\overline{x}_{ikm}$ değişkenleri (5.46) kısıtını sağlar. Kısıt (5.47) $\overline{x}_{ikm}$ değişkenleri tarafından sağlanır, çünkü her *i* değeri için $\overline{x}_{ikm} = 1$ olacak yalnızca bir (k,m) çifti vardır. Kısıt (5.48) $\overline{x}_{ikm}$ değerleriyle sağlanır, çünkü her *i* için yalnızca bir (k,m) çifti vardır. Kısıt (5.49) sağlanır, çünkü K_{max} değeri tanımı gereği C_{max}'e eşit veya daha büyüktür. Kısıt (5.50)'de yer alan F_0, F_1, F_2 ve F_3 şartları bir iş atandıktan sonra bu işin operasyon adedi kadar makineye ve periyot adedi kadar süreye başka bir işin atanmasını engeller. Bu engelleme kısıt (5.50)'de iki çift iş, makine ve periyot için gösterilmiştir. Aynı şart TP2'de kısıt (5.55) ile C_{max} hesabında her makine için bütün işler dikkate alınarak kullanılmıştır. Böylece kısıt (5.50) $\overline{x}_{ikm}$ değişkenleri ile sağlanır. Bu

gösterimlerle $\overline{x}_{ikm}$ değişkenlerinin TP1'deki bütün kısıtları sağladığı ve böylece TP1'in bir uygun çözümü olduğu gösterilmiştir. Ayrıca bu çözüm optimaldir, çünkü TP1 ve TP2 (TP1'in gevşetilmiş hali) amaç fonksiyonları $\overline{x}_{ikm}$ ve x^*_{im} değişkenleri ile aynı değerde bulunur. □

TP2/TP1 modelinin TP1'in optimal çözümünü verdiğine dair sonuç aşağıdaki gibidir.

Sonuç. *TP1 ve alt-problem TP1 optimal çözümleri aynıdır.*

5.6.1. TP2/TP1 ve TP1 karşılaştırması

TP2/TP1 yöntemi performansı gerçek verilerle elde edilen farklı büyüklükte problemlerle TP1 modeli ile karşılaştırılarak değerlendirilmiştir. Çizelge 5.25'de de görüldüğü gibi ilk üç sütun problemler hakkında bilgi içerirken sonraki kısımlar sırasıyla yöntemler tarafından bulunan çizelge uzunluklarını, bu çözümlere ne kadar sürede ulaşıldığını ve TP2/TP1 yöntemi ile elde edilen iyileşmeyi göstermektedir.

Çizelge 5.25. TP2/TP1 ve TP1 yöntemlerinin küçük boyutlu problemlerle karşılaştırması

			C_{max}(saat)		Çözüm süresi(sn)		%EAOS
Problem	İşler	Periyotlar	TP1	TP2/TP1	TP1	TP2/TP1	TP1-TP2/TP1
1	5	5	16,3	16,3	1,7	1,1	-
2	10	8	29,3	29,3	3,2	2,1	-
3	15	13	43,9	43,9	3,8	3,5	-
4	20	22	73,8	73,8	5,3	3,8	-
5	25	25	78,8	78,8	7,1	5,7	-
6	30	27	89,0	89,0	9,2	8,5	-
7	35	32	98,2	98,2	12,5	12,1	-
8	40	40	121,9	121,9	24,5	20,5	-
9	45	49	156,3	156,3	1080,0	42,3	-
10	52	54	164,0	164,0	325,7	54,4	-
				Ort.	147,3	15,4	

Çizelge incelendiğinde iki yöntem tarafından bulunan çözümlerin aynı olduğu görülmektedir. Bu durum TP2/TP1 yöntemi ile de optimal çözümlerin elde edilebildiğini göstermektedir. Bu problemler için iki yöntem arasındaki fark çözüm süreleri incelendiğinde belirginleşmektedir. TP1 yöntemi ortalama 147,3 saniyede optimal çözümleri üretebilirken TP2/TP1 yöntemi 15,4 saniyede sonuçlara ulaşabilmektedir. Karşılaştırmanın daha detaylı yapılabilmesi için büyük boyutlu problemler ele alınmıştır. Çizelge 5.3'deki düzgün dağılımlı veriler ile üretilen 50-70 iş barındıran problemlerle

gerçekleştirilen deney sonuçları Çizelge 5.26'da detaylandırılmıştır.

Çizelge 5.26. Düzgün dağılımlı veri ile TP2/TP1 ve TP1 karşılaştırması

			C_{max}(s)		Çözüm süresi(sn)		%Fark	%EAOS
Problem	İşler	Periyotlar	TP1	TP2/TP1	TP1	TP2/TP1	TP1	TP1-TP2/TP1
1		66	226,1	226,1	1800	181,9	1,2	0,0
2	50	69	238,4	238,4	1800	229,5	2,0	0,0
3		66	221,3	221,3	1800	229,2	1,4	0,0
1		81	269,3	269,3	1800	317,3	0,6	0,0
2	60	78	262,1	262,1	1800	282,7	1,9	0,0
3		74	244,2	244,2	1800	255,1	2,1	0,0
1		88	296,0	296,0	1800	473,6	0,2	0,0
2	70	90	299,6	**299,2**	1800	622,8	1,0	0,1
3		90	268,4	268,4	1800	554,0	0,2	0,0
				Ort.	1800	349,6		

Çizelge 5.26 incelendiğinde yöntemlerin hemen hemen aynı sonuçları bulduğu, ancak TP1'nin süre limitleri içinde optimali garantileyemediği görülmüştür. Ayrıca, çözüm süresi açısından TP1 modeli bütün problemlerde süre limit 1800 saniyeyi kullanmaktayken, TP2/TP1 ile ortalama 349,6 saniyede büyük boyutlu problemler için optimal çözümler elde edilebilmiştir. Sonuç olarak, TP2/TP1 yöntemi şimdiye kadar geliştirilen yöntemler arasında en hızlı ve aynı zamanda optimal çözümü veren bir yöntemdir. Geliştirilen yöntem gerçek hayat problemlerinde optimal çizelgelerin bulunmasında kullanılmıştır. Gerçekleştirilen çalışmalar sonraki bölümde detaylandırılmıştır.

5.7. Gerçek Verilerin Analizi

Tez çalışması kapsamında geliştirilen modellerin gerçek sistem verileri ile davranışlarını inceleyebilmek için Beyçelik Gestamp Kalıp Fabrikasındaki ardışık 13 presten oluşan manuel pres hattı ele alınmıştır. Paralel çizelgeleme problemi için geliştirilen modelleri test edebilmek amacıyla firmadaki 11.hafta-20.hafta (11.03.2013-19.05.2013) zaman dilimindeki 10 haftalık gerçek çizelgeler incelenmiştir. Firma haftalık taleplerle çalışmaktadır, dolayısıyla incelenen 10 haftalık çizelge ile 10 farklı problem oluşturulmuştur. İşlerin operasyon adetleri, alternatif makine kümeleri ve operasyon süreleri bellidir. Haftalık üretim adetleri ise karmaşık bir konudur. Firmada üretim adetleri

iki farklı veritabanında kaydedilmektedir. Bunlardan ilki, müşteri taleplerini içeren "üretim planları" veritabanıdır. İkinci veritabanı, firmadaki bütün preslerden veri toplayan "Üretim Yönetim Sistemi (ÜYS)" (Manufacturing Execution System-MES) tarafından oluşturulmaktadır. ÜYS bütün preslerdeki hatalı ürünler ve hurdalar da dahil olmak üzere üretim adetlerini veritabanına kaydeder. Bazı durumlarda firma planları dışında stok üretimi yapabilir ve bu durum ÜYS ile takip edilir. Dolayısıyla ÜYS veritabanındaki üretim adetleri genelde üretim planlarından daha fazladır. Deneyler için üretim adetleri belirlenirken ÜYS veritabanı dikkate alınmış, ve böylece hurda ve hatalı ürünlerin üretimi de karşılaştırmalara dahil edilmiştir.

Firmada üretim zamanı boyunca makinelerde arızalar meydana gelebilir, bakımlar gerçekleşebilir veya makineler başka bölümlere tahsis edilebilir. Bütün bu duruşlar, üretim takibinin yapıldığı veritabanlarında belirtilmektedir. Dolayısıyla, incelenen verilerde arıza ve bakım bilgileri mevcuttur. Bu duruşların yöntemler tarafından üretilen çizelgelerde de yansıtılması, gerçek sistemi temsil edebilmek için gereklidir. Duruşların yansıtılabilmesi için gerçekleşme zamanlarının, makinelerinin ve duruş sürelerinin tespit edilmesi ve yapay iş olarak tanımlanıp veri kümesine eklenmesi gerekmektedir. Duruşlar modellere yansıtılırken aşağıdaki adımlar izlenmiştir:

1. Duruşların başlangıç ve bitiş zamanları, toplam süreleri ve hangi makinelerde gerçekleştiği belirlenir.
2. Modeller duruşlar olmadan çalıştırılır ve çizelge uzunlukları belirlenir.
3. Modeller tarafından üretilen çizelgenin uzunluğu duruşun başlama zamanını kapsıyorsa duruş yapay iş olarak veri dosyasına eklenir. Bu eklemede yapay iş tek operasyondan oluşur. Tek alternatifi, duruşun gerçekleştiği makinedir ve üretim süresi, toplam duruş süresidir.
4. Modeller yapay veri ile tekrar çalıştırılır ve modellere ek bir kısıt eklenir. Bu kısıt, yapay işi başlama zamanında ilgili makineye atar. Böylece gerçek sistemde gerçekleşen duruş aynı zamanda aynı makinede temsil edilmiş olur.
5. Eğer model tarafından üretilen çizelgenin uzunluğu duruşun başlama zamanını kapsamıyorsa duruş bilgisi modellere eklenmez. Örneğin, modeller tarafından

üretilen çizelge 120. saatte bitmiş ve 130. saatte herhangi bir makinede bir arıza gerçekleşmiş ise bu duruş modele eklenmez.

Genel olarak *k'*. periyotta *m'*. makinede bir arıza meydana gelmiş ve bu arıza *i'*. iş olarak veri dosyasına kaydedilmiş ise TP1 modeline aşağıdaki kısıt eklenir:

$$x_{i'k'm'} = 1 \tag{5.60}$$

Benzer şekilde aynı arıza KP1 modelinde aşağıdaki kısıtın modele eklenmesiyle temsil edilir:

$$startOf(x_a) = k' * 4 * 3600, \forall a \in Alter | a.m = m', a.i = i' \tag{5.61}$$

MTBA modellerinde farklı bir durum söz konusudur. MTBA algoritması daha önce de anlatıldığı gibi ana problemi iki küçük probleme ayırır. Bunlardan ilki atama problemidir ve zaman indislerini barındırmaz, dolayısıyla zaman indisi olmadığı için duruşların kısıt olarak verilmesi kısıt (5.62) ile mümkündür.

$$x_{i'm'} = 1 \tag{5.62}$$

MTBA algoritmasının alt-problemi zaman indisi barındırır ve ikinci modele duruşların hangi zamanda gerçekleşeceği söylenebilir. MTBA1 algoritmasında alt-problem KP1'ye kısıt (5.61) eklenmiştir. Böylece duruşlar gerçek zamanlarında başlayabilmiştir. MTBA2 algoritmasında ise alt-problem TP1 olduğundan kısıt (5.60) eklenmiştir.

TP2/TP1 algoritmasında da benzer eklemeler mevcuttur. Algoritmanın ana-problemi TP2 modeli sadece iş ve makine indislerini barındırır. MTBA algoritmasındaki gibi zaman indisi yoktur. Dolayısıyla, kısıt (5.62)'nin eklenmesiyle ana-problemde duruşlar temsil edilmiş olur. TP1 modeli alt-problem için tanımlanmıştır ve zaman indisleri de mevcuttur. Bu sebeple kısıt (5.60) eklenmiştir. Bu şekilde karşılaştırılacak tüm modellerde duruşlar gerçek zamanlarında temsil edilebilmiştir.

Firma, problem tanımı aşamasında kalıp değişimlerini vardiya aralarında ve yemek molalarında yaptığını ve kalıp değişimi için aktif üretim süresini kullanmadıklarını belirtmiştir. Etkin kalıp değişim aralığının (periyot uzunluğu) belirlenmesi için yapılan deneyler de dört saatlik üretim sonrası kalıp değişiminin en iyi performansı sağladığını doğrulamıştır. Dolayısıyla, molalar ve vardiya değişimleri hariç firmanın günlük aktif üretim süresi 24 saat yerine 22,5 saattir. Bunların yanında, gerçek hayatta resmi tatillerden dolayı da duruşlar yaşanmaktadır (1 Mayıs). Bu gibi mecburi duruşlar, haftalık aktif üretim süresi hesaplanırken toplam süreye dahil edilmemiştir. Ayrıca, bakım, arıza ve atama gibi duruşların dışında firmanın *planlı duruş* olarak tanımladığı vardiya içi duruşlar mevcuttur. Planlı duruşlar; yönetimsel duruşlar, eğitimler ve toplantılar olarak sınıflandırılmıştır. İncelenen üretim verilerinde haftalık ortalama beş saatlik planlı duruş gerçekleştiği gözlemlenmiştir. Bu süre firmanın haftalık aktif üretim süresinden düşülmüş, böylece gerçek çalışma zamanları temsil edilebilmiştir. Bununla birlikte, şimdiye kadar geliştirilen modellerde kalıp değişimlerinin dört saatlik periyot aralarında (vardiya değişimi/yemek molaları) gerçekleştiği kabul edilmiş ve haricen kalıp değişim süresi modellere eklenmemiştir. Sonuç olarak, modellerin amaç fonksiyon değerlerinde ve mevcut çizelgelerde yalnızca aktif üretim süreleri dikkate alınmıştır. 10 haftalık verilerle gerçekleştirilen deneyler ve sonuçların gerçek çizelgelerle karşılaştırılması bir sonraki bölümde detaylandırılmıştır.

5.8. Gerçek Çizelgelerle Karşılaştırma

Bu bölümde TP1, KP1, MTBA1_maxkap, MTBA2 ve TP2/TP1 modelleri gerçek çizelgelerle karşılaştırılmıştır. Bütün yöntemler 1800 saniye ile sınırlanmış ve duruşlar yapay iş olarak veri kümesine ve sonrasında kısıt olarak yöntemlere eklenmiştir. Haftalık çizelgelerden elde edilen 10 farklı problem ile deneyler yapılmış ve Çizelge 5.27 ve 5.28'de detaylandırılmıştır. MTBA2 algoritması ile bütün deneyler tekrarlanmış, ancak süre limiti içerisinde herhangi bir uygun çözüm bulunamamıştır. Dolayısıyla, Çizelge 5.27'de yalnızca TP1, KP1 ve MTBA1_maxkap algoritmaları karşılaştırmaları vardır. MTBA1_maxkap algoritması sonuçları MTBA1 olarak gösterilmiştir. Bu modeller içinde en başarılı sonuçları üreten TP1 modeli sonraki çizelgeye de taşınmış

ve Çizelge 5.28'de önceki deneylerle başarılı olduğu gösterilen TP2/TP1 yöntemiyle karşılaştırılmıştır. Çizelgelerde ilk üç sütun gerçek veriler hakkında bilgi içermektedir. C_{max} kısmında "*Mevcut*" olarak adlandırılan kısım firmanın manuel olarak geliştirdiği çizelgenin süresini, diğer sütunlar ise modeller tarafından üretilen çizelgelerin sürelerini vermektedir. Sonrasında, çözüm süreleri ve %EAOS ile temsil edilen iyileştirmeler mevcuttur.

Çizelge 5.27. Gerçek çizelge ile TP1, KP1 ve MTBA1 karşılaştırılması

				C_{max}(saat)			Çözüm süresi(sn)		
Hafta	İşler	Periyotlar	Mevcut	TP1	KP1	MTBA1	TP1	KP1	MTBA1
1	63	35	145	102,7	108,1	131,6	1800	1800	1800
2	65	32	137,5	99,3	116,5	128,0	1800	1800	1800
3	60	32	137,5	100,8	105,5	126,6	1800	1800	1800
4	69	36	145	108,2	119,6	131,0	1800	1800	1800
5	60	36	137,5	106,3	111,0	130,8	1800	1800	1800
6	59	38	137,5	106,8	112,2	140,7	1800	1800	1800
7	59	32	145	100,9	118,1	123,4	1800	1800	1800
8	57	33	122,5	109,0	106,7	109,0	59	1800	1800
9	55	33	130	105,7	131,8	132,0	1800	1800	1800
10	49	38	145	124,3	137,0	152,0	73	1800	1800

Çizelge 5.27'de de görüldüğü gibi en iyi haftalık çizelgeler TP1 modeli ile üretilmiştir. KP1 modeli de MTBA1 ve mevcut çizelgelerden daha başarılı çizelgeler elde edebilmiştir. Bu karşılaştırmadaki en başarılı model TP1, bir sonraki çizelgede mevcut çizelgeler ile TP2/TP1 karşılaştırmasında kullanılmıştır. Çizelge 5.28'de ilk üç sütun gerçek veriler hakkında bilgi içermektedir. C_{max} kısmında dördüncü sütun firmanın manuel olarak geliştirdiği mevcut çizelgenin süresini, diğer sütunlar ise modeller tarafından üretilen çizelgelerin uzunluklarını vermektedir. Yedinci ve sekizinci sütunlar modellerin çalışma sürelerini göstermektedir. TP2/TP1 modelinin çalışma zamanı veri dosyasının iki defa okunmasını da içermektedir. Fark sütununda TP2/TP1 ve gerçek çizelgeler arasındaki saatlik fark hesaplanmıştır. Bu analizler detaylandırılarak modellerin buldukları sonuçlar ile mevcut çizelgeler arasındaki iyileşmeyi temsil eden %EAOS değerleri hesaplanmıştır.

Mevcut çizelgelerin toplam üretim zamanları resmi tatiller sebebiyle 122,5-145 saat arasında değişmektedir. Örnek olarak 8. haftada 1 Mayıs tatilinden dolayı gerçek çizelge uzunluğu 122,5 saate düşmektedir. Modeller tarafından üretilen çizelgeler incelendiğinde

Çizelge 5.28. TP1, TP2/TP1 ve gerçek çizelgelerin karşılaştırılması

			C_{max}(saat)			Çalışma Süresi(sn)		C_{max} Farkı	%Fark	%EAOS	
Hafta	İşler	Periyotlar	Mevcut	TP1	TP2/TP1	TP1	TP2/TP1	Mevcut-TP2/TP1	TP1	Mevcut-TP2/TP1	TP1-TP2/TP1
1	63	35	145	102,7	101,9	1800,0	92,9	43,1	1,1	29,7	0,8
2	65	32	137,5	99,3	98,0	1800,0	80,7	39,5	2,0	28,7	1,3
3	60	32	137.5	100,8	100,6	1800,0	134,5	36,9	1,1	26,8	0,2
4	69	36	145	108,2	107,8	1800,0	131,6	37,2	1,0	25,7	0,4
5	60	36	137,5	106,3	106,3	1800,0	81,1	31,2	1,5	22,7	0,0
6	59	38	137,5	106,8	106,8	1800,0	89,8	30,7	1,5	22,3	0,0
7	59	32	145	100,9	100,5	1800,0	66,7	44,5	1,3	30,7	0,4
8	57	33	122,5	109,0	109,0	59,0	58,7	13,6	-	11,1	0,0
9	55	33	130	105,7	105,4	1800,0	59,8	24,6	2,1	18,9	0,3
10	49	38	145	124,3	124,3	73,1	70,8	20,7	-	14,3	0,0
							En İyi	44,5	2,1	30,7	1,3
							En Kötü	13,6	1,0	11,1	0,0
							Ort.	32,2	1,4	23,1	0,3
							Std. Sp.	9,5	0,4	6,3	0,4
							Toplam	322,1			

TP2/TP1 modelinin TP1 ve mevcut çizelgelerden daha iyi bir performans sergilediği ve dakikalar içinde (ortalama 86,7 saniye) optimal çizelgeleri üretebildiği görülmektedir. Mevcut durumda bir mühendis haftanın ilk gününde 8 saat boyunca haftalık manuel çizelgeyi üretmekte ve haftanın geri kalan günlerinde her gün ortalama 4 saat harcayarak üretimin devam etmesini sağlamaktadır. Diğer taraftan TP2/TP1 modeli ile ilk çizelgeler ortalama 86,7 saniyede üretilebilir ve sonrasında duruşlar yaşandığında Bölüm 5.9'daki gibi duruş öncesi atamalar sabitlenerek kalan ürünler için yeni çizelgeler elde edilebilir.

TP2/TP1 ve mevcut çizelgeler arasındaki C_{max} farkları için en iyi, ortalama ve en kötü değerler standart sapmalarıyla hesaplanmıştır. Karşılaştırma sonuçlarına göre en iyi çizelgeler en hızlı şekilde TP2/TP1 ile elde edilmektedir. Örnek olarak, 1. haftada manuel olarak üretilen haftalık çizelge 145 saattir. TP2/TP1 modeli ile aynı üretim adetleri aynı duruşlarla 101,9 saatte gerçekleştirilebilmektedir. Benzer şekilde, TP2/TP1 ile 10. hafta için üretilmiş çizelge Şekil 5.7'deki gibidir. Ancak TP2/TP1 modeli duruşların bilgisi ile çizelgelemeye başlar ve bu durum bir avantaj olarak gözükebilir. Bu avantajı ortadan kaldırmak ve gerçek hayat uygulamasına yaklaşabilmek için duruşlar yaşandığında yeniden çizelgeleme yöntemi ile deneyler tekrarlanmış ve bir sonraki Bölüm 5.9'da detaylandırılmıştır.

	Makineler												
Periyot	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13
P1	48	24	5	5	9	9	9	9	9	28	28		
P2	48	24	5	5	9	9	9	9	9	28	28		
P3	20	20	20	20	9	9	9	9	9	28	28	18	
P4	20	20	20	20	9	9	9	9	9	28	28	18	
P5	20	20	20	20	49		44	46				18	
P6	21	21	21	21	49	45	44	46	14	14	14	14	
P7	21	21	21	21			44	46	14	14	14	14	
P8					27	27	27	27	14	14	14	14	
P9	3	3	3	3	27	27	27	27		29	29		
P10	3	3	3	3	26	26	26	26	26	29	29		
P11	22	22	22	22	26	26	26	26	26	29	29	34	
P12	22	22	22	22	26	26	26	26	26	29	29	34	32
P13	22	22	22	22	26	26	26	26	26	30	30	34	32
P14	23		19		26	26	26	26	26	30	30	34	32
P15	12	12	19	41	11	11		10	10	30	30	43	32
P16	12	12	19	41	11	11		10	10	30	30	43	32
P17	12	12		41	11	11		10	10			43	32
P18	7	7	7	7	11	11			38	38	38	43	
P19	7	7	7	7	2	2	2	2	38	38	38	43	
P20	7	7	7	7	2	2	2	2	37	37	37	43	36
P21	31	31	31	31	31				37	37	37		36
P22	31	31	31	31	31	17	17	17	37	37	37	47	
P23	31	31	31	31	31	17	17	17	37	37	37	47	
P24	39	39	39	39	39	17	17	17	37	37	37	47	
P25	39	39	39	39	39	17	17	17	37	37	37	35	
P26	39	39	39	39	39	17	17	17					
P27	13	13	6	6	6	6	6	6	15	15	15	15	
P28	13	13	6	6	6	6	6	6	15	15	15	15	
P29	13	13	6	6	6	6	6	6	15	15	15	15	
P30	25	42	6	6	6	6	6	6	15	15	15	15	33
P31	25	42	6	6	6	6	6	6	16	16	16	16	
P32	8	42	6	6	6	6	6	6	16	16	16	16	
P33	8	42	6	6	6	6	6	6	16	16	16	16	
P34	8	42		40	1	1	1	1	16	16	16	16	
P35	8	42		40	1	1	1	1	16	16	16	16	
P36	4	4	4		1	1	1	1	16	16	16	16	
P37	4	4	4		1	1	1	1	16	16	16	16	
P38	4	4	4		1	1	1	1	16	16	16	16	

Şekil 5.7. 10. hafta için TP2/TP1 ile üretilen çizelge

5.9. Yeniden Çizelgeleme Yöntemi

Bu bölümde mevcut çizelgeler ile TP2/TP1 modeli sonuçları karşılaştırılmıştır. Önceki bölümlerde detaylandırılan 10 haftalık veri kullanılmıştır. İlk olarak, TP2/TP1 modeli herhangi bir duruş olmadan çalıştırılmış ve bütün hafta için bir çizelge üretilmiştir. Sonrasında ilk duruş gerçekleştiğinde çizelgenin önceki kısmı dondurulmuş, duruş ve geri kalan işler modele tekrar eklenerek yeniden çizelgeleme yapılmıştır. Bir sonraki duruş gerçekleştiğinde aynı süreç tekrarlanmıştır.

Yeniden çizelgelemede bütün bitmiş işler için Bölüm 5.7'de detaylandırıldığı gibi TP2/TP1 alt-problemine kısıt (5.60) ve ana-probleme kısıt (5.62) eklenir. Eğer duruş

bitmemiş bir işin bölünmesine sebep olursa, işin kalan kısmı için yapay bir iş tanımlanır ve kalan kısmın duruştan sonra yapılmasına izin verilir. Ek olarak, bitmiş $i*$ adet iş haricindeki işlerin k_1 ve k_2 periyotları arasındaki atamalarını engellemek amacıyla kısıt (5.63) alt probleme eklenir.

$$\sum_{<i,k,m>\in Set_1 | k_1 \leq k \leq k_2} x_{ikm} = i* \tag{5.63}$$

Bu şekilde gerçek hayat uygulaması ve firma davranışları benzetimi gerçekleştirilmiştir. Örnek olarak, eğer 1. periyot ve 3. periyot arasındaki 7 işin ataması sabitlenmek istenirse alt-probleme kısıt (5.64) eklenir.

$$\sum_{<i,k,m>\in Set_1 | 1 \leq k \leq 3} x_{ikm} = 7 \tag{5.64}$$

Yeniden çizelgeleme yöntemi ile elde edilen deney sonuçları Çizelge 5.29'da detaylandırılmıştır. İlk üç sütun problem bilgilerini içermektedir. C_{max} kısmında TP2/TP1 ve mevcut haftalık çizelgelerin uzunlukları verilmiştir. Çizelgede de görüldüğü gibi TP2/TP1 aynı problem için birden fazla çalıştırılmıştır. Örnek olarak, 2. haftada duruşlar sebebiyle beş defa yeni çizelge üretilmiştir. Çalışma süresi, çizelgelerin oluşturulması için harcanan süreleri göstermektedir. Firma haftalık üretimi devam ettirebilmek için 28 saat (100 800sn) harcamaktadır, fakat TP2/TP1 ortalama 430,8 saniyede yeniden çizelgeleme ile haftalık üretimi devam ettirebilmektedir. Bu durum planlama sürecinde önemli bir iyileştirmedir. Fark sütunu, TP2/TP1 ve mevcut çizelge uzunlukları arasındaki saatlik farkı göstermektedir. TP2/TP1 ortalama 29,8 saat/hafta kısa çizelgeler üretebilmektedir. Geçmiş 10 haftalık çizelgeler göz önüne alındığında TP2/TP1 yöntemi 298 saat daha kısa sürede 10 haftalık üretimi tamamlayabilmektedir. Son sütun mevcut çizelgeler yerine TP2/TP1 yöntemi kullanıldığında elde edilecek iyileştirmeleri yüzdesel olarak temsil etmektedir. TP2/TP1 haftalık çizelgeleri ortalama % 21,4 iyileştirmektedir.

Çizelge 5.29. TP2/TP1 ile yeniden çizelgeleme yöntemi ve mevcut çizelgelerin karşılaştırması

			C_{max}(s)		Çözüm Süresi(sn)		Fark(s)	%EAOS
Hafta	İşler	Periyotlar	Mevcut	TP2/TP1	Mevcut	TP2/TP1	TP2/TP1-Mevcut	Mevcut-TP2/TP1
1	61	35	145.0	97,8	-	91,5	-	-
	62	35	145,0	101,1	-	94,5	-	-
	63	36	145,0	**105,9**	-	325,5	39,1	27,0
				Toplam	100800	**511,5**		
2	60	30	137,5	95,1	-	65,6	-	-
	62	32	137,5	93,7	-	112,6	-	-
	64	34	137,5	97,8	-	134,4	-	-
	65	35	137,5	99,8	-	199,4	-	-
	66	40	137,5	101,6	-	125,2	-	-
	67	40	137,5	**101,8**	-	128,9	35,7	26,0
				Toplam	100800	**766,2**		
3	57	35	137,5	96,1	-	71,9	-	-
	58	35	137,5	97,7	-	72,2	-	-
	60	35	137,5	103,0	-	294,2	-	-
	61	35	137,5	**103,2**		74,4	34,3	24,9
				Toplam	100800	**512,7**		
4	68	36	145,0	105,7	-	111,8	-	-
	69	36	145,0	**107,8**	-	128,5	37,2	25,7
				Toplam	100800	**240,3**		
5	59	36	137,5	103,0	-	79,0	-	-
	60	36	137.5	**117,2**	-	157,5	20,3	14,8
				Toplam	100800	**236,5**		
6	59	38	137,5	**106,8**	-	89,8	30,7	22,3
				Toplam	100800	**89,8**		
7	57	32	145,0	97,5	-	64.8	-	-
	59	33	145,0	101,9	-	90.1	-	-
	60	38	145,0	**101,9**	-	197,7	43,1	29,7
				Toplam	100800	**352,6**		
8	53	33	122,5	104,4	-	62,6	-	-
	54	33	122,5	104,4	-	133,9	-	-
	56	40	122,5	109,0	-	183,3	-	-
	57	40	122,5	109,0	-	353,6	-	-
	58	40	122,5	**109,0**	-	369,4	13,5	11,1
				Toplam	100800	**1102,8**		
9	53	33	130	104,1	-	75,0	-	-
	55	33	130	105,1	-	98,3	-	-
	56	33	130	**106,7**	-	115,8	23,3	17,9
				Toplam	100800	**289,1**		
10	48	38	145,0	124,3	-	92,1	-	-
	50	40	145,0	**124,3**	-	114,8	20,7	14.3
				Toplam	100800	**206,9**		
					Ort.	430,8	29,8	21,4
						Toplam	298,0	

Sonuç olarak, TP2/TP1 ile gerçek çizelgelerin optimal sonuçları dakikalar içinde elde edilebilmektedir. Haftalık üretim 28 saat yerine ortalama 430,8 saniyede devam ettirilebilmekte ve mevcut çizelgelere oranla ortalama %21,4 daha kısa çizelgeler elde edilebilmektedir. TP2/TP1 yönteminin kapasite artışına etkisini detaylı incelemek için bir sonraki bölümde talepler farklı oranlarda arttırılarak deneyler tekrarlanmıştır.

6. İLAVE ÇALIŞMALAR

6.1. Kapasite Artışı Analizi

Önceki bölümde TP2/TP1 ile daha kısa çizelgelerin bulunabileceği gösterilmiştir. TP2/TP1 yönteminin sınırlarını araştırmak için farklı taleplerle deneyler tekrarlanmıştır. İşlerin talepleri sırayla %5, %10, %15, %20 ve %25 oranında arttırılmış ve sonuçlar Çizelge 6.1'de TP2/TP1_5D, TP2/TP1_10D, TP2/TP1_15D, TP2/TP1_20D ve TP2/TP1_25D ile temsil edilmiştir.

Çizelge 6.1. Artan talepler ile TP2/TP1 deneyleri C_{max} sonuçları

			C_{max}(saat)				
Hafta	İşler	Mevcut	TP2/TP1_5D	TP2/TP1_10D	TP2/TP1_15D	TP2/TP1_20D	TP2/TP1_25D
1	63	145	106,9	111,6	116,4	121,2	126,1
2	64	137,5	102,9	107,5	112,3	117,1	121,5
3	55	137,5	105,4	110,1	115,0	119,7	124,3
4	61	145	112,9	118,2	123,4	128,6	133,9
5	59	137,5	111,2	115,8	121,0	126,2	131,4
6	58	137,5	113,1	118,5	123,9	129,3	134,7
7	57	145	105,1	109,9	114,9	119,7	124,4
8	54	122,5	113,9	118,9	**123,8**	**128,8**	**133,7**
9	54	130	110,5	115,6	120,2	125,4	**130,6**
10	49	145	130,5	136,7	142,9	**149,1**	**155,3**
	Ort.	138,3	111,2	116,3	121,4	126,5	131,6

Çizelge 6.1'de de görüldüğü gibi talepler %25 oranında arttırıldığında TP2/TP1 modeli ortalamada firmanın çizelge uzunluğundan daha kısa sürelerde talep artışını karşılayabilmektedir. Bazı problemlerde modelin talep artışlı çizelge uzunlukları mevcut çizelge sürelerini aşmaktadır. Örnek olarak, talep artışı %15 olduğunda her hafta için daha kısa çizelgeler 8. hafta hariç elde edilebilmektedir. Benzer olarak talep artışı %20 olduğunda on haftadan ikisi için mevcut çizelgelerden daha uzun çizelgeler bulunmaktadır. Toplamda eğer talep %25 arttırılırsa mevcut çizelgelerin kırk dört (toplamda elli deneyden) tanesi için daha kısa çizelgeler üretilebilir. Sonuç olarak, TP2/TP1 yöntemi ile en az %10 oranında talep artışları karşılanabilmektedir.

Yöntem ile talep artışlarına yapılabilecek katkı ortaya koyulmuştur. Bu artışlar kullanılarak henüz gerçekleşmemiş üretim haftalarının çizelge uzunlukları ve karşılanabilecek talep miktarları belirlenebilir. Ancak, gerçek sistemde duruşların

meydana gelmesi ve ne kadar süreceği stokastik yapılardır ve çoğu durumda önceden kestirilememektedir. Duruşlar ile ilgili istatistiki bilgiler gerçek çalışma ortamından toplanabilir ve kullanılabilir formata getirilebilirse, henüz gerçekleşmemiş olan üretim haftaları için toplam çizelge uzunlukları gerçeğe daha yakın bir şekilde tahmin edilebilir. Bu amaçla, duruşların istatistiksel olarak analizi gerçekleştirilmiş ve sonraki bölümde anlatılmıştır.

6.2. Duruşların İstatistiksel Analizi

Bu bölümde 10 haftalık çizelgelerde gerçekleşen duruşlar istatistiksel olarak incelenmiştir. Duruşların gerçekleşme sıklıkları, hangi makinelerde ne kadar sürede gerçekleştikleri gibi bilgilerin dağılımları bilinebilirse, bu dağılımlarla yapay duruşlar haftalık taleplere eklenebilir ve henüz gerçekleşmemiş olan üretim haftaları için toplam çizelge uzunlukları gerçeğe yakın tahmin edilebilir. Bu amaçla belirlenmesi gereken dağılımlar sırasıyla, duruşların gerçekleşme adetleri, duruşların hangi makinelerde gerçekleşeceği ve süreleridir.

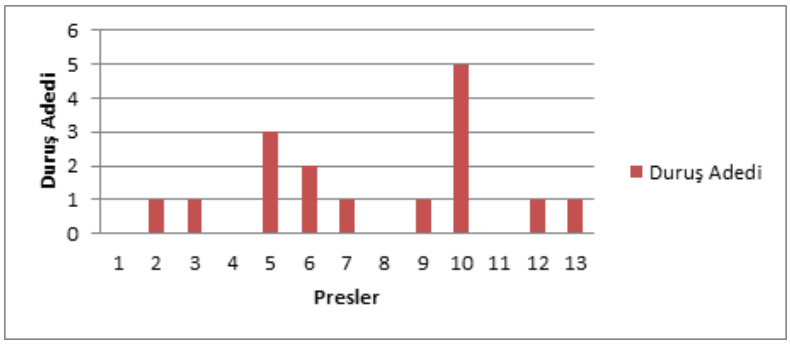
Bütün duruşların haftalara göre dağılımı Çizelge 6.2'deki gibidir. İlgili hafta, haftanın toplam aktif çalışma süresi, duruş tipi, duruşun gerçekleştiği makine, duruşun süresi (saat) ve duruşun ilgili makinenin çalışma süresi içindeki yüzdesi bilgileri çizelgede yer almaktadır. Bu duruşlardan kalıp fabrikasına atamalar firmanın tercihi dahilinde olduğundan analizlerden çıkarılmış ve bir sonraki Çizelge 6.3 elde edilmiştir. Çizelge 6.3 incelendiğinde bakım ve arızadan kaynaklanan duruşların 10 haftalık çalışma süresinin %1,6'sını kapsadığı görülmüştür.

Haftalara göre bakım ve arızaların gerçekleşme sıklıkları incelenmiş ve yapılan analizler sonucu, gerçekleşme sıklıklarının Binom (6, 0,267) dağılımına uyduğu tespit edilmiştir. Bakım ve arızaların sürelerinin haftalara göre dağılımı incelenmiş ve Lognormal (3,46, 1,37, 1,94) dağılıma uyduğu tespit edilmiştir.

Çizelge 6.2. Tüm duruşların haftalara göre dağılımı

Hafta	Süre(saat)	Duruş Tipi	Makine	Süre(saat)	Duruş %
1	145	Kalıp	13	127,5	6,8
1	145	Arıza	6	16,0	0,8
2	137,5	Kalıp	13	16,5	0,9
2	137,5	Kalıp	9	10,8	0,6
2	137,5	Kalıp	8	8,0	0,4
2	137,5	Bakım	6	8,5	0,5
2	137,5	Kalıp	13	12,8	0,7
2	137,5	Kalıp	13	20,3	1,1
2	137,5	Arıza	2	18,0	1,0
3	137,5	Kalıp	13	26,5	1,5
3	137,5	Kalıp	13	30,0	1,7
3	137,5	Arıza	7	4,0	0,2
3	137,5	Kalıp	13	45,0	2,5
4	145	Kalıp	13	46,5	2,5
5	137,5	Arıza	9	27,8	1,6
5	137,5	Bakım	3	3,8	0,2
6	137,5	Bakım	10	41,0	2,3
7	145	Bakım	10	15,0	0,8
7	145	Bakım	10	105,0	5,6
8	122,5	Arıza	10	5,3	0,3
8	122,5	Arıza	10	4,5	0,3
8	122,5	Bakım	12	5,5	0,3
8	122,5	Bakım	13	3,5	0,2
9	130	Arıza	5	7,5	0,4
9	130	Bakım	5	22,5	1,3
10	145	Arıza	5	5,3	0,3

Makineler bazında arıza ve bakımların dağılımı incelediğinde Şekil 6.1'de de görüldüğü gibi en fazla duruşun 10. makinede olduğu görülmüştür. 10. makinedeki duruş, toplam duruşun %58,3'ünü oluşturmaktadır.



Şekil 6.1. Duruşların makinelere göre dağılımı

Çizelge 6.3. Bakım ve arızaların haftalara göre dağılımı

Hafta	Süre(saat)	Duruş Tipi	Makine	Süre(saat)	%Duruş
1	145	Arıza	6	16,0	0,8
2	137,5	Bakım	6	8,5	0,5
2	137,5	Arıza	2	18,0	1,0
3	137,5	Arıza	7	4,0	0,2
5	137,5	Arıza	9	27,8	1,6
5	137,5	Bakım	3	3,8	0,2
6	137,5	Bakım	10	41,0	2,3
7	145	Bakım	10	15,0	0,8
7	145	Bakım	10	105,0	5,6
8	122,5	Arıza	10	5,3	0,3
8	122,5	Arıza	10	4,5	0,3
8	122,5	Bakım	12	5,5	0,3
8	122,5	Bakım	13	3,5	0,2
9	130	Arıza	5	7,5	0,4
9	130	Bakım	5	22,5	1,3
10	145	Arıza	5	5,3	0,3
Toplam	1382,5			293,0	1,6

Ayrıca duruşların gerçekleştiği makinelere göre dağılımı Çizelge 6.4'de detaylandırılmıştır. Duruş yüzdeleri kesikli olarak incelenerek kümülatif toplamları alınmıştır. Bu toplamlarda duruş adedi sıfır olan makinelere de olasılık vermek amacıyla bir *epsilon* değeri $\varepsilon = 0,01$ kullanılmıştır. Kümülatif toplamlar bulunmuş ve sonrasında duruş adedi kadar rassal sayı üretilmiştir. Bu rassal sayıların denk geldiği aralığa göre duruşun gerçekleştiği makine belirlenmiştir. Öncelikle Binom dağılımı kullanılarak 10

Çizelge 6.4. Mevcut duruşların makinelere dağılımı

Makine	Duruş Adedi	Yüzde	Düzenlenmiş	Kümülatif
1	0	0	0,01	0,01
2	1	0,06	0,06	0,07
3	1	0,06	0,06	0,14
4	0	0	0,01	0,15
5	3	0,19	0,18	0,33
6	2	0,13	0,12	0,45
7	1	0,06	0,06	0,51
8	0	0	0,01	0,52
9	1	0,06	0,06	0,58
10	5	0,31	0,28	0,86
11	0	0	0,01	0,87
12	1	0,06	0,06	0,93
13	1	0,06	0,06	1
Toplam	16	1	1	

hafta için duruş adetleri belirlenmiş ve toplamda 17 adet duruş haftalara dağıtılmıştır. Duruş sürelerinin dağılımı kullanılarak 17 duruş için süreler belirlenmiştir. Son olarak, 17 adet duruşun gerçekleştiği makineleri belirlemek için 17 adet (0,1) aralığında rassal sayı üretilmiş ve kümülatif sütunda denk geldiği aralıktan makineler belirlenmiştir. Dağılımlara uygun olarak üretilen duruşlar Çizelge 6.5'de detaylandırılmıştır.

Çizelge 6.5. Dağılımlara uygun olarak üretilen duruşlar

Hafta	Duruş	Rassal sayılar	Makineler	Duruş Süresi
1	1	0,9448	13	2,5
2	2	0,3234	5	1,04
2	3	0,7609	10	87,45
3	4	0,1733	5	4,59
4	5	0,1985	5	23,39
5	6	0,203	5	4,84
5	7	0,53	9	5,01
6	8	0,2075	5	0,34
7	9	0,5058	7	0,98
7	10	0,3808	6	2,9
7	11	0,303	5	17,37
8	12	0,6629	10	0,75
8	13	0,7203	10	10,79
9	14	0,8109	10	112,89
10	15	0,5609	9	37,68
10	16	0,6857	10	0,81
10	17	0,6828	10	3,77

10 hafta için elde edilen duruşlar haftalık verilere yapay iş olarak eklenmiş ve deneyler tekrarlanmıştır. Deney sonuçları Çizelge 6.6'da detaylandırılmıştır.

Çizelge 6.6. Dağılımlara uygun üretilen yapay duruşlarla deney sonuçları

			C_{max}(saat)		Çözüm Süresi(sn)	Fark (saat)	%EAOS
Hafta	İşler	Periyotlar	Mevcut	TP2/TP1	TP2/TP1	Mevcut-TP2/TP1	Mevcut-TP2/TP1
1	62	40	145	97,8	97,9	47,2	32,5
2	62	50	137,5	**170,3**	139,9	-32,8	-23,8
3	58	40	137,5	96,4	77,1	41,1	29,9
4	69	40	145	108,0	142,9	37,0	25,5
5	61	40	137,5	103,5	104,5	34,0	24,7
6	60	40	137,5	106,9	86,5	30,6	22,3
7	60	40	145	98,4	89,1	46,6	32,1
8	55	40	122,5	110,8	67,1	11,8	9,6
9	54	55	130	**192,7**	153,0	-62,7	-48,2
10	50	40	145	128,9	75,3	16,1	11,1
					En İyi	47,2	32,5
					En Kötü	-62,7	-48,2
					Ort.	16,9	11,6

Çizelge sonuçları incelendiğinde yöntemin yine hızlı şekilde optimal sonuçları ürettiği görülmüş, fakat bazı haftalarda mevcut çizelge sürelerinden uzun süreler elde edilmiştir. Bunun sebebi, ikinci haftaya eklenen 87,5 saatlik duruş ile dokuzuncu haftaya eklenen 112,9 saatlik duruştur. Buna rağmen, yapılan testlerin %80'inde mevcut çizelge uzunluklarından daha kısa çizelgelere ortalama 103,3 saniyede ulaşılabilmiştir. Yapılan çalışmalar ilerletilerek sonraki bölümde farklı amaç fonksiyonları ile elde edilen çizelgeler karşılaştırılmıştır.

6.3. Farklı Amaç Fonksiyonları ile Üretilen Çizelgelerin Karşılaştırılması

Bu bölümde haftalık çizelgelerin üretilmesinde maksimum tamamlanma zamanının minimizasyonu (min C_{max}) yanında üç farklı amaç fonksiyonu geliştirilmiştir. Toplamda dört farklı amaç fonksiyonu ile üretilen çizelge uzunlukları ve yapılan atamalar karşılaştırılmıştır. Bu amaç fonksiyonları aşağıdaki gibidir:

1. Mevcut amaç fonksiyonu C_{max} değerinin minimizasyonudur. Bu hesaplamada işlerin başlangıç periyotları önemli değildir, makinelere atanan işlerin işlem süreleri dikkate alınır.

$$min\, C_{max} \tag{6.1}$$

2. C_{max} değerinin yanında işlerin başlangıç periyotları da dikkate alınmıştır. Burada C_{max} değerinin ilk hedef olduğunu vurgulamak için büyük bir L sayısı ile çarpılmıştır.

$$min\, L * C_{max} + \sum_{<i,k,m>\in Set2} x_{ikm} * k \tag{6.2}$$

3. İşlerin makineler üzerinde dengeli dağılımını sağlamak amacıyla makinelerdeki işlerin tamamlanma zamanlarının ortalamaları alınmış ve C_{ort} değişkeninde tutulmuştur. Ortalama tamamlanma zamanının minimizasyonu hedeflenmiştir.

$$min\, C_{ort} \tag{6.3}$$

4. Her makine için tamamlanma zamanı cap_m hesaplanmaktadır. Dördüncü amaç fonksiyonunda C_{ort} değerinin makineler arasında dengeli bir dağılımı temsil edebilmesi için cap_m değişkeni ile farkının minimizasyonu amaçlanmıştır. Ayrıca işlerin son periyotlara ötelenmesini engellemek için başlangıç periyotlarının önemli olduğu ikinci kısım da amaç fonksiyonuna eklenmiştir. Yapılan deneyler, ikinci ifadenin çok büyük bir sayıyla çarpılmasının sonucu değiştirmediğini göstermiştir.

$$min \mid cap_m - C_{ort} \mid + \sum_{<i,k,m> \in Set2} x_{ikm} * k \tag{6.4}$$

Farklı amaç fonksiyonlarının karşılaştırılması deneylerinde TP1 kullanılmıştır. Deneyler ilk olarak küçük boyutlu örneklerle gerçekleştirilmiştir. Farklı amaç fonksiyonları ile elde edilen tamamlanma zamanları TP1_1, TP1_2, TP1_3 ve TP1_4 ile temsil edilmiştir. Deney sonuçları Çizelge 6.7'de detaylandırılmıştır. Anlamlı bir karşılaştırma için bütün deneylerde amaç fonksiyonlarına ilave olarak C_{max} değerleri hesaplanmış ve karşılaştırmalarda kullanılmıştır. Hesaplanan amaç fonksiyonları ve gerçek C_{max} değerleri arasındaki farkı gösterebilmek için ortalama tamamlanma zamanlarının minimize edildiği 3. amaç fonksiyonu ele alınmıştır. Bu amaç fonksiyonu kullanıldığında model oldukça hızlanmıştır, ancak ortaya çıkan çizelgeler incelendiğinde C_{max} değerlerinin diğer modellere göre büyük çıktığı görülmektedir. Benzer sonuçlar 4. amaç fonksiyonu için de söylenebilir. Çizelge 6.7 incelendiğinde ilk beş problem için

Çizelge 6.7. 4 farklı amaç fonksiyonu ile TP1 modeli sonuçları

		TP1_1	TP1_2	TP1_3	TP1_4	Çözüm Süresi (sn)			
İşler	Periyotlar	C_{max}	C_{max}	$C_{max}(C_{ort})$	C_{max}	TP1_1	TP1_2	TP1_3	TP1_4
5	5	16,3	16,3	16,3(6,9)	16,3	1,2	2,1	2,8	3,3
10	8	29,3	29,3	29,3(15,0)	29,3	3,4	3,2	3,8	3,8
15	13	43,9	43,9	43,9(27,3)	44,3	3,4	3,5	3,8	5,0
20	22	73,8	73,8	73,8(46,6)	73,8	4,3	3,9	4,4	4,0
25	25	78,8	78,8	78,8(51,9)	78,8	6,5	5,3	5,9	5,4
30	27	**89,0**	**89,0**	**89,0**(73,6)	91,8	9,4	9,2	8,9	9,8
35	32	**98,2**	**98,2**	105,0(84,0)	105,0	12,2	12,1	11,8	13,6
40	40	**121,9**	**121,9**	137,9(109,0)	134,0	22,5	35,1	20,1	45,1
45	49	**156,3**	**156,3**	171,6(139,0)	174,5	1015,9	180,0	30,8	102,3
52	54	**164,0**	**164,0**	178,1(150,1)	193,1	311,9	1800,0	45,4	198,8

aynı çizelge uzunluklarının elde edildiği görülmektedir. Daha büyük problemlerde amaç fonksiyonları arasındaki fark da artmaktadır. En kısa çizelgeler 1. ve 2. amaç fonksiyonu

ile elde edilmiştir. En uzun çizelgeler 4. amaç fonksiyonu ile bulunmuştur. Bu çizelgeleri detaylı olarak incelemek için farkın ilk ortaya çıktığı 6. problem (30 iş) ele alınmıştır. 6. problemde ilk üç amaç fonksiyonu ile 89 saat olarak bulunan çizelgeler, 4. amaç fonksiyonu ile 91,8 saat olarak bulunmuştur. 2. amaç fonksiyonu ile 30 işin ataması Şekil 6.2'deki gibi ve 4. amaç fonksiyonu ile elde edilen atama Şekil 6.3'deki gibidir.

	Makineler												
Zaman(s)	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13
3,5	14	14	14	6	15	15	15	15	15	24	9	1	10
	25	25	25	6	15	15	15	15	15	24	9	1	10
	25	25	25	6	15	15	15	15	15	5	9		7
	2	2	21	21	15	15	15	15	15	5	9		7
	2	2	21	21	4	13	13	13		5	9		7
	2	2	21	21	4	13	13	13	8	8	8	8	22
	19	19	19	19	4	28	26	26	8	8	8	8	22
27,7	19	19	19	19	23	28	26	26	8	8	8	8	22
	16	16	16	16	23	28	26	26	8	8	8	8	22
	16	16	16	16	23	28	26	26	29	29	29	29	22
	3	3	3	3	23	28	26	26	29	29	29	29	
	3	3	3	3	23	28	26	26	29	29	29	29	
	3	3	3	3	27	28	26	26	29	29	29	29	
	12	12	12	12	27	28	26	26	30	30	30	30	
	12	12	12	12	27	28	26	26	30	30	30	30	
50,3	12	12	12	12	27	28	26	26	30	30	30	30	
	17	17	17	17	27	28	26	26	30	30	30	30	
	17	17	17	17	27		26	26	30	30	30	30	
	17	17	17	17	27				20	20	20	20	20
	11	11	11	11	27				20	20	20	20	20
	11	11	11	11	27				20	20	20	20	20
71,6	11	11	11	11	27				20	20	20	20	20
	18	18	18	18	27				20	20	20	20	20
	18	18	18	18									
	18	18	18	18									
	18	18	18	18									
89,02	18	18	18	18									

Şekil 6.2. 2. amaç fonksiyonu ile 30 iş ataması

	Makineler												
Zaman(s)	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13
3,5	14	14	14	6	4	13	13	13	9	24	10	1	7
	25	25	25	6	4	13	13	13	9	24	10	1	7
	25	25	25	6	4	28	26	26	9	5			7
	2	2	21	21	23	28	26	26	9	5			22
	2	2	21	21	23	28	26	26	9	5			22
	2	2	21	21	23	28	26	26	30	30	30	30	22
	16	16	16	16	23	28	26	26	30	30	30	30	22
	16	16	16	16	23	28	26	26	30	30	30	30	22
	19	19	19	19	27	28	26	26	30	30	30	30	
	19	19	19	19	27	28	26	26	30	30	30	30	
	11	11	11	11	27	28	26	26	8	8	8	8	
	11	11	11	11	27	28	26	26	8	8	8	8	
46	11	11	11	11	27	28	26	26	8	8	8	8	
	17	17	17	17	27		26	26	8	8	8	8	
	17	17	17	17	27				29	29	29	29	
	17	17	17	17	27				29	29	29	29	
	12	12	12	12	27				29	29	29	29	
	12	12	12	12	27				29	29	29	29	
	12	12	12	12	27				20	20	20	20	20
	3	3	3	3					20	20	20	20	20
	3	3	3	3					20	20	20	20	20
79	3	3	3	3					20	20	20	20	20
	18	18	18	18					20	20	20	20	20
	18	18	18	18	15	15	15	15	15				
	18	18	18	18	15	15	15	15	15				
	18	18	18	18	15	15	15	15	15				
91,8	18	18	18	18	15	15	15	15	15				

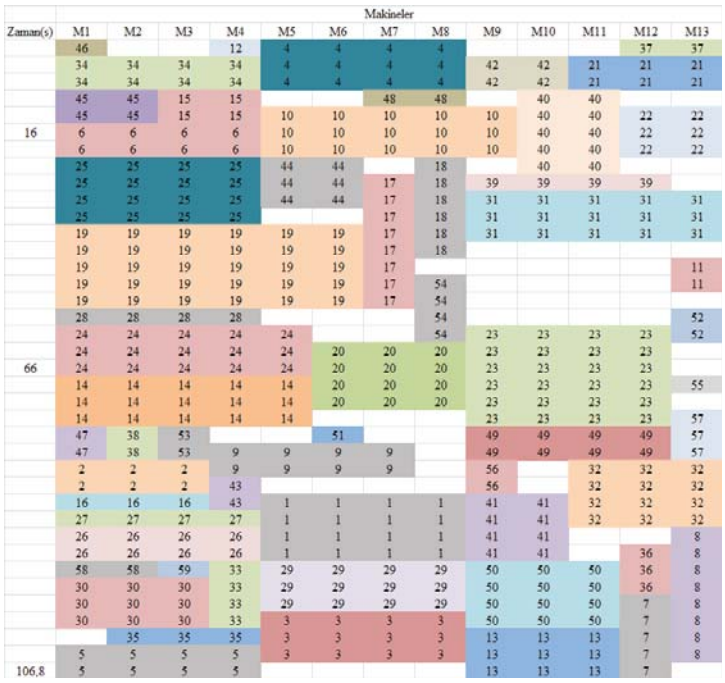
Şekil 6.3. 4. amaç fonksiyonu ile 30 iş ataması

İki çizelgedeki fark M9 makinesine atanan işlerden kaynaklanmaktadır. 4. amaç fonksiyonu ile çalışan modelde M9'a fazladan 9. iş yüklenmiştir. Bu da çizelge uzunluğunun 91,8 saat olmasına sebep olmuştur. Farklı amaç fonksiyonlarının etkilerini incelemek için 10 haftalık gerçek veriler ile deneyler tekrarlanmıştır. Deney sonuçları Çizelge 6.8'de detaylandırılmıştır. Sonuçlar incelendiğinde, ilk iki amaç fonksiyonu ile bulunan çizelge uzunluklarının diğer amaç fonksiyonları ile elde edilen çözümlerden daha kısa olduğu görülmektedir. Bununla birlikte problemlerin çözüm süreleri incelendiğinde 3. amaç fonksiyonunun TP1'i hızlandırdığı görülmektedir. Ancak, 1. amaç fonksiyonlu çizelgeler 3. amaç fonksiyonu kullanılarak elde edilen çizelgelerden daha iyidir. Bu

Çizelge 6.8. Haftalık çizelgelerle 4 farklı amaç fonksiyonu karşılaştırması

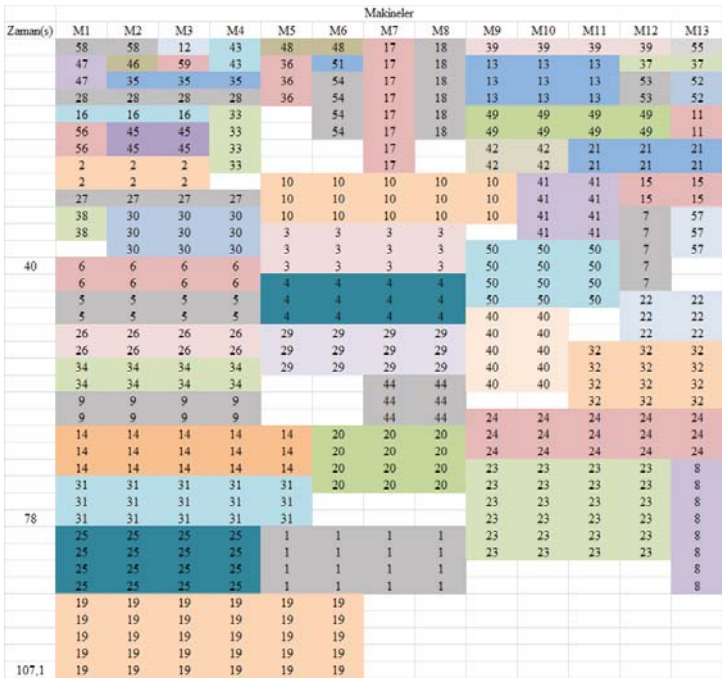
			TP1_1	TP1_2	TP1_3	TP1_4	Çözüm Süresi (sn)				%Fark			
Hafta	İşler	Periyotlar	C_{max}	C_{max}	$C_{max}(C_{ort})$	C_{max}	TP1_1	TP1_2	TP1_3	TP1_4	TP1_1	TP1_2	TP1_3	TP1_4
1	63	35	**102,7**	102,8	113,2(101,6)	113,6	1800,0	1800,0	70,5	196,7	1,1	1,1	-	-
2	65	32	99,3	**98,8**	109,1(97,3)	109,8	1800,0	1800,0	56,4	1800,0	2,0	1,5	-	0,4
3	60	32	**100,8**	101,4	107,6(99,7)	105,8	1800,0	1800,0	48,1	1007,6	1,1	1,6	-	-
4	69	36	**108,2**	108,6	123,2(107,2)	119,1	1800,0	1800,0	89,1	1800,0	1,0	1,3	-	1,4
5	60	36	**106,3**	**106,3**	119,3(101,3)	121,7	1800,0	1800,0	61,3	238,0	1,5	1,6	-	-
6	59	38	**106,8**	107,1	127,0(104,6)	124,7	1800,0	1800,0	71,9	1800,0	1,5	1,8	-	1,0
7	59	32	**100,9**	101,3	110,6(99,6)	110,7	1800,0	1800,0	49,2	858,2	1,3	1,4	-	-
8	57	33	**109,0**	**109,0**	110,5(87,3)	111,0	59,0	44,5	43,3	48,6	-	-	-	-
9	55	33	**105,7**	**105,7**	113,7(103,2)	110,9	1800,0	1800,0	46,9	1727,5	2,1	2,2	-	-
10	49	38	**124,3**	**124,3**	131,3(114,1)	135,1	73,1	66,5	66,1	1316,5	-	-	-	-

sonuçları detaylı incelemek için farkın en büyük olduğu 6. hafta çizelgeleri ele alınmıştır. Üç amaç fonksiyonu kullanılarak üretilen 6. hafta çizelgeleri şekillerle detaylandırılmıştır. 2. amaç fonksiyonunun hedefi 1. amaç fonksiyonu ile elde edilen çizelgelerdeki boşlukların azaltılmasıdır. Şekil 6.5'de 2. amaç fonksiyonu ile üretilen çizelge, Şekil 6.4'de 1. amaç fonksiyonlu çizelge ile karşılaştırıldığında işlerin üst periyotlara doğru sıkıştırıldığı görülmektedir. Örnek olarak, Şekil 6.4'de 23. iş 31. işten hemen sonra işlenebiliyorken iki iş arasında gereksiz, toplam çizelge uzunluğunu etkilemeyen bir boşluk bırakılmıştır. Bir diğer deyişle, 1. amaç fonksiyonu ile toplam çizelge uzunluğu aynı kaldığı sürece ardışık işler arasında boşluklar meydana gelebilmektedir. Ancak 2. amaç fonksiyonu ile bu gereksiz boşluklar kaldırılmıştır. 3. amaç fonksiyonu ile üretilen çizelge 2. amaç fonksiyonu ile elde edilen çizelgeye benzemesine rağmen karşılaştırma tablosunda da görüldüğü gibi aralarında yaklaşık 20 saat bir fark vardır. 3. amaç fonksiyonunda makinelerdeki işlerin tamamlanma zamanları dengelenmek istenmiştir ve yapılan atamalar incelendiğinde dengeli bir yerleşim yapıldığı görülmektedir. Ancak bu amaç, haftalık çizelgenin tamamlanma zamanını olumsuz etkilemektedir.



	Makineler												
Zaman(s)	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13
	46			12	4	4	4	4				37	37
	34	34	34	34	4	4	4	4	42	42	21	21	21
	34	34	34	34	4	4	4	4	42	42	21	21	21
	45	45	15	15			48	48		40	40		
	45	45	15	15	10	10	10	10	10	40	40	22	22
16	6	6	6	6	10	10	10	10	10	40	40	22	22
	6	6	6	6	10	10	10	10	10	40	40	22	22
	25	25	25	25	44	44		18		40	40		
	25	25	25	25	44	44	17	18	39	39	39	39	
	25	25	25	25	44	44	17	18	31	31	31	31	31
	25	25	25	25			17	18	31	31	31	31	31
	19	19	19	19	19	19	17	18	31	31	31	31	31
	19	19	19	19	19	19	17	18					
	19	19	19	19	19	19	17						11
	19	19	19	19	19	19	17	54					11
	19	19	19	19	19	19	17	54					
	28	28	28	28				54					52
	24	24	24	24	24			54	23	23	23	23	52
	24	24	24	24	24	20	20	20	23	23	23	23	
66	24	24	24	24	24	20	20	20	23	23	23	23	
	14	14	14	14	14	20	20	20	23	23	23	23	55
	14	14	14	14	14	20	20	20	23	23	23	23	
	14	14	14	14	14				23	23	23	23	57
	47	38	53			51			49	49	49	49	57
	47	38	53	9	9	9	9		49	49	49	49	57
	2	2	2	9	9	9	9		56		32	32	32
	2	2	2	43					56		32	32	32
	16	16	16	43	1	1	1	1	41	41	32	32	32
	27	27	27	27	1	1	1	1	41	41	32	32	32
	26	26	26	26	1	1	1	1	41	41			8
	26	26	26	26	1	1	1	1	41	41		36	8
	58	58	59	33	29	29	29	29	50	50	50	36	8
	30	30	30	33	29	29	29	29	50	50	50	36	8
	30	30	30	33	29	29	29	29	50	50	50	7	8
	30	30	30	33	3	3	3	3	50	50	50	7	8
		35	35	35	3	3	3	3	13	13	13	7	8
	5	5	5	5	3	3	3	3	13	13	13	7	8
106,8	5	5	5	5					13	13	13	7	

Şekil 6.4. 1. amaç fonksiyonu ile 6. hafta ataması



	Makineler												
Zaman(s)	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13
	58	58	12	43	48	48	17	18	39	39	39	39	55
	47	46	59	43	36	51	17	18	13	13	13	37	37
	47	35	35	35	36	54	17	18	13	13	13	53	52
	28	28	28	28	36	54	17	18	13	13	13	53	52
	16	16	16	33		54	17	18	49	49	49	49	11
	56	45	45	33		54	17	18	49	49	49	49	11
	56	45	45	33			17		42	42	21	21	21
	2	2	2	33			17		42	42	21	21	21
	2	2	2		10	10	10	10	10	41	41	15	15
	27	27	27	27	10	10	10	10	10	41	41	15	15
	38	30	30	30	10	10	10	10	10	41	41	7	57
	38	30	30	30	3	3	3	3		41	41	7	57
		30	30	30	3	3	3	3	50	50	50	7	57
40	6	6	6	6	3	3	3	3	50	50	50	7	
	6	6	6	6	4	4	4	4	50	50	50	7	
	5	5	5	5	4	4	4	4	50	50	50	22	22
	5	5	5	5	4	4	4	4	40	40		22	22
	26	26	26	26	29	29	29	29	40	40		22	22
	26	26	26	26	29	29	29	29	40	40	32	32	32
	34	34	34	34	29	29	29	29	40	40	32	32	32
	34	34	34	34			44	44	40	40	32	32	32
	9	9	9	9			44	44			32	32	32
	9	9	9	9			44	44	24	24	24	24	24
	14	14	14	14	14	20	20	20	24	24	24	24	24
	14	14	14	14	14	20	20	20	24	24	24	24	24
	14	14	14	14	14	20	20	20	23	23	23	23	8
	31	31	31	31	31	20	20	20	23	23	23	23	8
	31	31	31	31	31				23	23	23	23	8
78	31	31	31	31	31				23	23	23	23	8
	25	25	25	25	1	1	1	1	23	23	23	23	8
	25	25	25	25	1	1	1	1	23	23	23	23	8
	25	25	25	25	1	1	1	1					8
	25	25	25	25	1	1	1	1					8
	19	19	19	19	19	19							
	19	19	19	19	19	19							
	19	19	19	19	19	19							
	19	19	19	19	19	19							
107,1	19	19	19	19	19	19							

Şekil 6.5. 2. amaç fonksiyonu ile 6. hafta ataması

Yapılan karşılaştırmalar sonucu en iyi çizelgelerin 1. ve 2. amaç fonksiyonları ile elde edildiği ve 2. amaç fonksiyonu ile boşluksuz, firma tarafından uygulanabilir makul yerleştirmeler yapıldığı gözlenmiştir.

	Makineler												
Zaman(s)	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13
	27	27	27	27			43		10	10	10	10	10
	16	16	16				43		10	10	10	10	10
	56			33	29	29	29	29	10	10	10	10	10
	56			33	29	29	29	29	39	39	39	39	
	30	30	30	33	29	29	29	29	31	31	31	31	31
25	30	30	30	33			48	48	31	31	31	31	31
	30	30	30	12					31	31	31	31	31
	25	25	25	25	1	1	1	1	41	41	32	32	32
	25	25	25	25	1	1	1	1	41	41	32	32	32
	25	25	25	25	1	1	1	1	41	41	32	32	32
	25	25	25	25	1	1	1	1	41	41	32	32	32
	5	5	5	5		21	21	21	34	34	34	34	57
	5	5	5	5		21	21	21	34	34	34	34	57
	28	28	28	28		20	20	20	38	40	40	7	57
	24	24	24	24	24	20	20	20	38	40	40	7	52
	24	24	24	24	24	20	20	20		40	40	7	52
	24	24	24	24	24	20	20	20		40	40	7	
	2	2	2	9	9	9	9			40	40	7	
	2	2	2	9	9	9	9	18	47				
	6	6	6	6			17	18	47	37	37	22	22
	6	6	6	6			17	18	42	42		22	22
	19	19	19	19	19	19	17	18	42	42		22	22
	19	19	19	19	19	19	17	18	23	23	23	23	
75	19	19	19	19	19	19	17	18	23	23	23	23	
	19	19	19	19	19	19	17		23	23	23	23	
	19	19	19	19	19	19	17		23	23	23	23	8
	58	58					17		23	23	23	23	8
				59	3	3	3	3	23	23	23	23	8
	26	26	26	26	3	3	3	3	49	49	49	49	8
	26	26	26	26	3	3	3	3	49	49	49	49	8
						54	44	44					8
						54	44	44	13	13	13		8
	14	14	14	14	14	54	44	44	13	13	13		8
	14	14	14	14	14	54	15	15	13	13	13	36	
	14	14	14	14	14		15	15	50	50	50	36	
	45	45	53		4	4	4	4	50	50	50	36	11
	45	45	53		4	4	4	4	50	50	50	55	11
	46	35			4	4	4	4	50	50	50	51	
126,9	46												

Şekil 6.6. 3. amaç fonksiyonu ile 6. hafta ataması

7. DEĞERLENDİRME VE SONUÇ

Doktora tezi kapsamında NP-zor sınıfı bir bağımsız paralel makine çizelgeleme problemi ele alınmıştır. Bu problem, makine elverişlilik ve kaynak kısıtları altında ardışık makinelerde işlenmesi gereken farklı sayıda operasyona sahip işlerin bağımsız paralel makine ortamında eş zamanlı olarak çizelgelenmesi ($R_m|M_i; res; chains|C_{max}$) şeklinde sınıflandırılmıştır.

Çalışmada problem çözümüne yönelik farklı yaklaşımlar geliştirilmiştir. Bu yaklaşımlardan ilki KTP1 modeli olarak adlandırılan karışık tamsayılı matematiksel modeldir ve problemi mevcut durumun varsayımlarını da test edecek şekilde yansıtmaktadır. İkinci olarak, kısıt programlamanın çizelgelemeye yönelik özel değişken ve kısıt tanımları kullanılmış ve KP1 geliştirilmiştir. Bu model en az KTP1 kadar iyi bir model olmasına rağmen bir alt sınır vermemesi ve optimalden uzaklık bilgisine sahip olmaması bir dezavantaj olmuştur. KP1'in KTP1'den hızlı çalışmasının sebepleri araştırılmış ve KP1 mantığında, bir alt sınır verebilen TP1 modeli geliştirilmiştir. TP1 modeli KTP1 ve KP1'den daha iyi olmasına rağmen gerçek boyutlu çizelgelerin optimal sonuçlarına ulaşmada sıkıntılarla karşılaşılmıştır.

Bu aşamadan sonra kısıt programlama ve matematiksel programlamanın avantajlı yönlerini birleştiren yöntemler araştırılmıştır. Mantık-tabanlı Benders ayrıştırma yöntemi ile MTBA1 ve MTBA2 geliştirilmiştir. MTBA yöntemi var olan bir problemi iki kısma ayırır ve her bir kısmı ayrı ayrı çözerek büyük problemin çözümüne ulaşır. Bu amaçla çizelgeleme problemi iki kısma ayrılmıştır: ana-problem, işlerin makinelere atanması ve alt-problem, atanan işlerin makinelerde çizelgelenmesidir. Ana-problem yeni bir matematiksel model kullanılarak (TP2) çizelgelenmiştir. Alt-problem olarak MTBA1'de KP1 kullanılmış, ancak MTBA1 KP1'nin dezavantajlarını barındırdığından MTBA2 geliştirilmiştir. MTBA2'de alt-problem TP1 tarafından çözülmektedir. MTBA modellerinde üretilen kesimler, modellerin uygun çözümlere ulaşmasını geciktirmiştir.

Benders ayrıştırma yönteminden esinlenerek TP2/TP1 yöntemi geliştirilmiş ve yöntemin optimum sonucu garantilediği ispatlanmıştır. Uygulama problemi olarak bir pres hattı ele alınmıştır. Gerçek hayattaki çizelgeleme problemi verileri analiz edilerek üretim verileri oluşturulmuş ve bu veriler yöntemlerin test edilmesinde kullanılmıştır. TP2/TP1 yöntemiyle duruşların, gerçekleştiği anda temsil edilmesi için yeniden çizelgeleme yapılmıştır. Haftalık çizelgelerin birden fazla üretilmesine rağmen TP2/TP1 yöntemi ile dakikalar içinde optimal çizelgeler oluşturulabilmiştir. Yöntemin kapasite artışına sağlayabileceği katkılar incelenmiş ve duruşların dağılımları belirlenerek henüz gerçekleşmemiş üretim çizelgeleri için tahmini süreler elde edilebilmiştir. Son olarak, farklı amaç fonksiyonları değerlendirilerek elde edilen çizelgeler karşılaştırılmıştır.

Tez çalışması ile ortaya koyulan faydalar incelendiğinde ilk önce problem tanımının özgünlüğü akla gelir. Makine elverişlilik ve kaynak kısıtları altında ardışık makinelerde işlenmesi gereken farklı sayıda operasyona sahip işlerin bağımsız paralel makine ortamında eş zamanlı olarak çizelgelenmesi $R_m|M_i; res; chains|C_{max}$ problemi literatürde bilindiği kadarıyla incelenmemiştir. İlk defa bu problem tanımı yapılarak literatürdeki bağımsız paralel makine çizelgeleme problemlerine bir yenisi eklenmiştir. Problem tanımı için ilk karışık tamsayılı model, ilk kısıt programlama modeli ve ilk mantık-tabanlı Benders ayrıştırma yöntemleri geliştirilmiştir. Ayrıca, ayrıştırma tekniklerinden esinlenilerek optimum sonucu garantileyen TP2/TP1 algoritması geliştirilmiştir. Algoritmanın hızından yararlanarak yeniden çizelgeleme yöntemi geliştirilmiş ve makine arızaları durumunda kalan üretim için yeniden çizelgeleme yapılarak uygun olmayan makinelerdeki atamalar güncellenmiştir. Ek olarak, geliştirilen yöntemin mevcut probleme katkılarının incelenmesi için kapasite artışları araştırılmıştır. Son olarak, duruşların istatistiksel analizi yapılmış ve farklı amaç fonksiyonları modellerde kullanılarak elde edilen çizelgeler karşılaştırılmıştır.

Sonuç olarak, gerçek-boyutlu bir bağımsız paralel makine çizelgeleme probleminin optimal çözümü TP2/TP1 algoritması ile dakikalar içinde elde edilebilmiştir. Bu başarının bir devamı olarak, bağımsız paralel makine çizelgeleme problemlerine benzer farklı çizelgeleme problemleri için TP2/TP1 algoritması güncellenebilir ve

farklı sınıfta problemlere uygulanarak bu problemlerin de optimal çözümlerine ulaşılabilir. Tez çalışması kapsamında geliştirilen ve sezgisel yerine optimali sağlayan bu algoritma, literatürdeki benzer yapıdaki NP-zor problemlere uygulanarak, henüz çözümü sağlanamamış problemler için bir çözüm yöntemi olarak kullanılabilir.

KAYNAKLAR

Apt, K. 2003. *Principles of constraint programming*, Cambridge University Press.

Baker, K. R., Trietsch, D. 2009. *Principles of sequencing and scheduling*, John Wiley & Sons.

Baptiste, P., Le Pape, C., Nuijten, W. 2001. *Constraint-based scheduling: applying constraint programming to scheduling problems*, Vol. 39, Springer.

Barlatt, A. Y., Cohn, A., Gusikhin, O., Fradkin, Y., Davidson, R., Batey, J. 2012. Ford motor company implements integrated planning and scheduling in a complex automotive manufacturing environment, *Interfaces* 42(5): 478–491.

Bazaraa, M. S., Jarvis, J. J., Sherali, H. D. 2011. *Linear programming and network flows*, John Wiley & Sons.

Blazewicz, J., Lenstra, J. K., Kan, A. 1983. Scheduling subject to resource constraints: classification and complexity, *Discrete Applied Mathematics* 5(1): 11–24.

Brearley, A., Mitra, G., Williams, H. P. 1975. Analysis of mathematical programming problems prior to applying the simplex algorithm, *Mathematical programming* 8(1): 54–83.

Brucker, Peter ve Knust, S. 2006a. Complexity results for scheduling problems.
URL: *http://www.informatik.uni-osnabrueck.de/knust/class/*

Brucker, Peter ve Knust, S. 2006b. The scheduling zoo.
URL: *http://www-desir.lip6.fr/∼ durrc/query/*

Cambazard, H., Hladik, P.-E., Déplanche, A.-M., Jussien, N., Trinquet, Y. 2004. Decomposition and learning for a hard real time task allocation problem, *Principles and Practice of Constraint Programming–CP 2004*, Springer, pp. 153–167.

Canto, S. P. 2008. Application of benders' decomposition to power plant preventive maintenance scheduling, *European journal of operational research* 184(2): 759–777.

Chen, B., Potts, C. N., Woeginger, G. J. 1999. A review of machine scheduling: Complexity, algorithms and approximability, *Handbook of combinatorial optimization*, Springer, pp. 1493–1641.

Cheng, T., Sin, C. 1990. A state-of-the-art review of parallel-machine scheduling research, *European Journal of Operational Research* 47(3): 271–292.

Chu, Y., Xia, Q. 2004. Generating benders cuts for a general class of integer programming problems, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, Springer, pp. 127–141.

Chu, Y., Xia, Q. 2005. A hybrid algorithm for a class of resource constrained scheduling problems, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, Springer, pp. 110–124.

Coban, E., Hooker, J. N. 2010. Single-facility scheduling over long time horizons by logic-based benders decomposition, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, Springer, pp. 87–91.

Cohen, J. 1988. A view of the origins and development of prolog, *Communications of the ACM* 31(1): 26–36.

Cohen, J. 1990. Constraint logic programming languages, *Communications of the ACM* 33(7): 52–68.

Conway, R., Maxwell, W. 1967. L. miller, theory of scheduling, *AW Pub* .

Darby-Dowman, K., Little, J., Mitra, G., Zaffalon, M. 1997. Constraint logic programming and integer programming approaches and their collaboration in solving an assignment scheduling problem, *Constraints* 1(3): 245–264.
Edis, E. B., Oguz, C., Ozkarahan, I. 2013. Parallel machine scheduling with additional resources: Notation, classification, models and solution methods, *European Journal of Operational Research* 230(3): 449–463.
Edis, E. B., Ozkarahan, I. 2011. A combined integer/constraint programming approach to a resource-constrained parallel machine scheduling problem with machine eligibility restrictions, *Engineering Optimization* 43(2): 135–157.
Edis, E. B., Ozkarahan, I. 2012. Solution approaches for a real-life resource-constrained parallel machine scheduling problem, *The International Journal of Advanced Manufacturing Technology* 58(9-12): 1141–1153.
Çetinkaya, F. C. 2009. *İngilizce-türkçe endüstri mühendisliği ve mühendislik yönetimi terimleri sözlüğü*, TMMOB Makine mühendisleri odası.
Fazel-Zarandi, M. M., Beck, J. C. 2009. Solving a location-allocation problem with logic-based benders' decomposition, *Principles and Practice of Constraint Programming-CP 2009*, Springer, pp. 344–351.
Flood, M. M. 1954. Application of transportation theory to scheduling a military tanker fleet, *Journal of the Operations Research Society of America* 2(2): 150–162.
Framinan, J. M., Ruiz, R. 2010. Architecture of manufacturing scheduling systems: Literature review and an integrated proposal, *European Journal of Operational Research* 205(2): 237–246.
Gantt, H. L. 1919. *Organizing for work*, Harcourt, Brace and Howe.
Garey, M. R., Johnson, D. S. 1978. "strong"np-completeness results: Motivation, examples, and implications, *Journal of the ACM (JACM)* 25(3): 499–508.
Garey, M. R., Johnson, D. S. 1979. *Computers and intractability*, Vol. 174, freeman San Francisco.
Gomes, C. P. 2004. Randomized backtrack search, *Constraint and Integer Programming*, Springer, pp. 233–291.
Graham, R. L., Lawler, E. L., Lenstra, J. K., Kan, A. 1979. Optimization and approximation in deterministic sequencing and scheduling: a survey, *Annals of discrete mathematics* 5: 287–326.
Harjunkoski, I., Grossmann, I. E. 2002. Decomposition techniques for multistage scheduling problems using mixed-integer and constraint programming methods, *Computers & Chemical Engineering* 26(11): 1533–1552.
Heinz, S., Beck, J. C. 2012. Reconsidering mixed integer programming and mip-based hybrids for scheduling, *Integration of AI and OR Techniques in Contraint Programming for Combinatorial Optimzation Problems*, Springer, pp. 211–227.
Held, M., Karp, R. M. 1962. A dynamic programming approach to sequencing problems, *Journal of the Society for Industrial & Applied Mathematics* 10(1): 196–210.
Hentenryck, P. V. 2002. Constraint and integer programming in opl, *INFORMS Journal on Computing* 14(4): 345–372.
Hillier, F. S., Lieberman, G. J. 2001. *Introduction to operations research*, Tata McGraw-Hill Education.
Hooker, J. 2011. *Logic-based methods for optimization combining optimization and constraint satisfaction*, Vol. 2, John Wiley & Sons.

Hooker, J. N. 2002. Logic, optimization, and constraint programming, *INFORMS Journal on Computing* 14(4): 295–321.
Hooker, J. N. 2005. Planning and scheduling to minimize tardiness, *Principles and Practice of Constraint Programming-CP 2005*, Springer, pp. 314–327.
Hooker, J. N. 2007. Planning and scheduling by logic-based benders decomposition, *Operations Research* 55(3): 588–602.
Hooker, J. N., Ottosson, G. 2003. Logic-based benders decomposition, *Mathematical Programming* 96(1): 33–60.
Hooker, J. N., Yan, H. 1995. Logic circuit verification by benders decomposition, *Principles and practice of constraint programming: The Newport Papers* pp. 267–288.
Jackson, J. R. 1956. An extension of johnson's results on job idt scheduling, *Naval Research Logistics Quarterly* 3(3): 201–203.
Jain, V., Grossmann, I. E. 2001. Algorithms for hybrid milp/cp models for a class of optimization problems, *INFORMS Journal on computing* 13(4): 258–276.
Jordan, C., Drexl, A. 1995. A comparison of constraint and mixed-integer programming solvers for batch sequencing with sequence-dependent setups, *ORSA Journal on Computing* 7(2): 160–165.
Kanet, J. J., Ahire, S. L., Gorman, M. F. 2004. Constraint programming for scheduling.
Karp, R. M. 1972. *Reducibility among combinatorial problems*, Springer.
Khayat, G. E., Langevin, A., Riopel, D. 2006. Integrated production and material handling scheduling using mathematical programming and constraint programming, *European journal of operational research* 175(3): 1818–1832.
Kravchenko, S. A., Werner, F. 2011. Parallel machine problems with equal processing times: a survey, *Journal of Scheduling* 14(5): 435–444.
Lageweg, B., Lenstra, J. K., Lawler, E., Kan, A. 1981. Computer aided complexity classification of deterministic scheduling problems, *Report BM 138, Centre for Mathematics and Computer Science* .
Lageweg, B., Lenstra, J. K., Lawler, E., Kan, A. 1982. Computer-aided complexity classification of combinational problems, *Communications of the ACM* 25(11): 817–822.
Leung, J. Y. 2004. *Handbook of scheduling: algorithms, models, and performance analysis*, CRC Press.
Liao, T., Chang, P., Kuo, R., Liao, C.-J. 2014. A comparison of five hybrid metaheuristic algorithms for unrelated parallel-machine scheduling and inbound trucks sequencing in multi-door cross docking systems, *Applied Soft Computing* 21: 180–193.
Li, K., Yang, S.-l. 2009. Non-identical parallel-machine scheduling research with minimizing total weighted completion times: Models, relaxations and algorithms, *Applied mathematical modelling* 33(4): 2145–2158.
Lin, Y., Pfund, M. E., Fowler, J. W. 2011. Heuristics for minimizing regular performance measures in unrelated parallel machine scheduling problems, *Computers & Operations Research* 38(6): 901–916.
Liu, C. 2013. A hybrid genetic algorithm to minimize total tardiness for unrelated parallel machine scheduling with precedence constraints, *Mathematical Problems in Engineering* 2013.
Lustig, I. J., Puget, J.-F. 2001. Program does not equal program: Constraint programming and its relationship to mathematical programming, *Interfaces* 31(6): 29–53.
Malapert, A., Guéret, C., Rousseau, L.-M. 2012. A constraint programming approach

for a batch processing problem with non-identical job sizes, *European Journal of Operational Research* 221(3): 533–545.
Meyr, H., Mann, M. 2013. A decomposition approach for the general lotsizing and scheduling problem for parallel production lines, *European Journal of Operational Research* .
Mokotoff, E. 2001. Parallel machine scheduling problems: a survey, *Asia Pacific Journal of Operational Research* 18(2): 193–242.
Neos Server **2014**. http://www.neos-server.org/neos/solvers/go:scip/CPLEX.html.
Nuijten, W. P., Aarts, E. H. 1996. A computational study of constraint satisfaction for multiple capacitated job shop scheduling, *European Journal of Operational Research* 90(2): 269–284.
Pfund, M., Fowler, J. W., Gupta, J. N. 2004. A survey of algorithms for single and multi-objective unrelated parallel-machine deterministic scheduling problems, *Journal of the Chinese Institute of Industrial Engineers* 21(3): 230–241.
Pinedo, M. 2012. *Scheduling theory, algorithms, and systems*, Springer.
Prosser, P. 1993. Hybrid algorithms for the constraint satisfaction problem, *Computational intelligence* 9(3): 268–299.
Reisman, A., Kumar, A., Motwani, J. 1997. Flowshop scheduling/sequencing research: A statistical review of the literature, 1952-1994, *Engineering Management, IEEE Transactions on* 44(3): 316–329.
Shahzad, A. 2010. A single machine scheduling problem with individual job tardiness based objectives.
URL: *http://oro.univ-nantes.fr/sujets-09-10/shahzad.pdf*
Slowinski, R. 1980. Two approaches to problems of resource allocation among project activities–a comparative study, *Journal of the Operational Research Society* pp. 711–723.
Smith, B. M., Brailsford, S. C., Hubbard, P. M., Williams, H. P. 1996. The progressive party problem: Integer linear programming and constraint programming compared, *Constraints* 1(1-2): 119–138.
Spielman, D. A., Teng, S.-H. 2004. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time, *Journal of the ACM (JACM)* 51(3): 385–463.
Tavakkoli-Moghaddam, R., Taheri, F., Bazzazi, M., Izadi, M., Sassani, F. 2009. Design of a genetic algorithm for bi-objective unrelated parallel machines scheduling with sequence-dependent setup times and precedence constraints, *Computers & Operations Research* 36(12): 3224–3230.
Tran, T. T., Beck, J. C. 2012. Logic-based benders decomposition for alternative resource scheduling with sequence dependent setups., *ECAI*, pp. 774–779.
Van Hentenryck, P. 1999. *The OPL optimization programming language*, Mit Press.
Vazsonyi, A. 1955. The use of mathematics in production and inventory control-ii (theory of scheduling), *Management Science* 1(3-4): 207–223.
Ventura, J. A., Kim, D. 2000. Parallel machine scheduling about an unrestricted due date and additional resource constraints, *Iie Transactions* 32(2): 147–153.
Wagner, H. M. 1959. An integer linear-programming model for machine scheduling, *Naval Research Logistics Quarterly* 6(2): 131–140.
Williams, H. P., Wilson, J. M. 1998. Connections between integer linear programming and constraint logic programming—an overview and introduction to the cluster of articles, *INFORMS Journal on Computing* 10(3): 261–264.

Xiaolu, L., Baocun, B., Yingwu, C., Feng, Y. 2014. Multi satellites scheduling algorithm based on task merging mechanism, *Applied Mathematics and Computation* 230: 687–700.

ÖZGEÇMİŞ

Adı Soyadı	: Burcu ÇAĞLAR GENÇOSMAN
Doğum Yeri ve Tarihi	: Bursa, 02.05.1984
Yabancı Dili	: İngilizce.
Eğitim Durumu (Kurum ve Yıl)	
Lise	: Şükrü Şankaya Anadolu Lisesi 2002
Lisans	: Uludağ Üniversitesi 2006
Yüksek Lisans	: Uludağ Üniversitesi 2009
Çalıştığı Kurum/Kurumlar ve Yıl	: Uludağ Üniversitesi Müh. Fak. 2006-...
İletişim (e-posta)	: burcucaglar@uludag.edu.tr

ULUSLAR ARASI YAYINLARI

SCI ve SCI–Expanded indekslerince taranan dergilerde yayınlanan veya yayına kabul edilen tam metin özgün araştırma makaleleri

1. Çağlar, B., Özmutlu, H. C., Özmutlu, S. (Basımda) Character N-Gram Application for Automatic New Topic Identification. *Information Processing& Management*, http://dx.doi.org/10.1016/j.ipm.2014.06.005

2. Eroğlu, D. Y., Gençosman, B. Ç., Çavdur, F., Özmutlu, H. C.. Introducing the MCHF/OVRP/SDMP: Multi-Capacitated Heterogeneous Fleet Open Vehicle Routing Problems with Split Deliveries and Multi-Products. *The Scientific World Journal*, Volume 2014, Article ID 515402, http://dx.doi.org/10.1155/2014/515402.

Diğer indekslerce taranan dergilerde yayınlanan veya yayına kabul edilen tam metin özgün araştırma makaleleri

1. Gençosman, B. Ç., Özmutlu, H. C., Hnich, B. 2012. Production Scheduling using Constraint Programming. *The 18th International Conference on Principles and Practice of Constraint Programming*, sf.1-6.

2. Çavdur, F., Özmutlu, H.C., Özmutlu, S., Eroğlu, D., Gençosman, B.Ç. 2012. Automatic new topic identification using goal programming, *3rd International Conference on Industrial Engineering and Operations Management*, sf:1103-1110.

Değerlendirme Aşamasındaki Yayınlar

1. Gençosman, B. Ç., Özmutlu, H. C., Özkan, H., Beğen, M. Automative stamping scheduling using mixed integer programming and constraint programming. *Non-Traditional Applications of IE.* (Kitap Bölümü).

2. Gençosman, B. Ç., Özmutlu, H. C., Özkan, H., Beğen, M. Scheduling of Stamping Operations at Beyçelik using Mathematical Programming. *Manufacturing & Service Operations Management*.

ULUSAL YAYINLARI

1. Özmutlu, H.C., Çağlar, B. 2009. Arama Motorlarında Yeni Konu Tanılamada Karakter N-gram ve Yapay Sinir Ağları Uygulaması. *Mühendislik-Mimarlık Fakültesi Dergisi*, Cilt:14, Sayı:2 sf: 75-91.

KONGRE FAALİYETLERİ (ULUSLAR ARASI)

1. Gençosman, B. Ç., Özmutlu, H. C., Özkan, H., Beğen, M. 2013. Automotive Stamping Scheduling using Mixed Integer Programming and Constraint Programming. *International IIE Conference*, İstanbul.

2. Gençosman, B. Ç., Eroğlu, D. Y., Özmutlu, H. C. 2013. Performance comparison between Character N-gram and Levenshtein edit-distance for automatic topic identification in search engines. *International Software Conference and Exhibition*, Bursa.

3. Gençosman, B. Ç., Eroğlu, D. Y., Özmutlu, H. C. 2013. Estimation of topic changes in search engines: Improved character n-gram method by pre-processed spelling detection methods. *International Software Conference and Exhibition*, Bursa.

4. Gençosman, B. Ç., Eroğlu, D. Y., Özmutlu, H. C. 2013. Factor Analysis of the character n-gram method for topic identification in search engines. *International Software Conference and Exhibition*, Bursa.

5. Gençosman, F., Aygül, U., Gençosman, B. Ç. 2013. Positive effect of implementing the production management system (PMS) to companies. *International Software Conference and Exhibition*, Bursa.

6. Eroğlu Y. D., Gençosman, B. Ç., Kılıç, K. 2013. Improved Hybrid Genetic-Local Search Algorithm for Simultaneous Feature Selection and Weighting. *International Software Conference and Exhibition*, Bursa.

7. Eroğlu Y. D., Gençosman, B. Ç., Kılıç, K. 2013. A Comparative Study on Feature Selection Methods: Techniques in Weka & Hybrid Genetic-Local Search Algorithm. *International Software Conference and Exhibition*, Bursa.

8. Eroğlu Y. D., Gençosman, B. Ç., Kılıç, K. 2013. A Genetic Algorithm Based Model for Innovation Management. *International Software Conference and Exhibition*, Bursa.

9. Gençosman, B. Ç., Eroğlu Y. D., Beğen, M. 2013. Combined use of Constraint Programming and Mixed Integer Programming: A case study in Stamping Scheduling. *International Conference on Business, Management, Economics and Finance*, İzmir.

10. Eroğlu Y. D., Gençosman, B. Ç., Özmutlu, H. C. 2013. Improved Mixed Integer Programming models for the unrelated parallel machine scheduling problem with job splitting and sequence dependent setup times. *International Conference on Business, Management, Economics and Finance*, İzmir.

11. Gençosman, B. Ç., Orbak, Y. A., Orbak, İ. 2012. Factor Analysis and Response Surface Optimization for Copper Removal from Aqueous Solutions. *3rd International Conference on Industrial Engineering and Operations Management*, İstanbul.

12. Çavdur, F., Özmutlu, H. C., Özmutlu, S., Eroğlu, D., Gençosman, B. Ç. 2012. The Open Vehicle Routing Problem with Multiple Products and Vehicles. *Workshop on Women in Industrial Engineering Academia*, İstanbul.

13. Çağlar, B., Özmutlu, H. C., Orbak, Y. A., Özmutlu, S. 2010. Statistical Procedures for Robotic Assembly Line Balancing Problems. *24rd European Conference on Operational Research*, Portekiz.

KONGRE FAALİYETLERİ (ULUSAL)

1. Gençosman, B. Ç., Özmutlu, H. C., Özkan, H. 2013. Kalıp Fabrikası Pres Hatlarının Çizelgelenmesi. *Uludağ Üniversitesi III. Bilgilendirme ve AR-GE Günleri*, Bursa.

2. Gençosman, B. Ç., Orbak, Y. A., Orbak, İ. 2012. Endüstride Kullanılan Sulu Çözeltilerden Kadmiyum Ayrışımı için Faktör Analizi ve Cevap Yüzeyi Yöntemi Kullanımı. *Yöneylem Araştırması ve Endüstri Mühendisliği 32. Ulusal Kongresi (YA/EM 2012)*, İstanbul.

3. Gençosman, B.Ç., Eroğlu, Y. D., Çavdur, F., Özmutlu, S., Özmutlu, C. 2012. Açık Uçlu Araç Rotalama Probleminde Genetik Algoritma Yaklaşımı. *Yöneylem Araştırması ve Endüstri Mühendisliği 32. Ulusal Kongresi (YA/EM 2012)*, İstanbul.

4. Eroğlu, Y. D., Gençosman, B.Ç., Çavdur, F., Özmutlu, S., Özmutlu, C. 2012. Açık Uçlu Araç Rotalama Probleminde İyileştirilmiş Genetik Algoritma Yaklaşımı. *Uludağ Üniversitesi II. Bilgilendirme ve AR-GE Günleri*, Bursa.

5. Özmutlu, H.C., Çağlar, B. 2008. Yeni Konu Tanılamada Karakter N-gram Yaklaşımı. *Yöneylem Araştırması ve Endüstri Mühendisliği 28. Ulusal Kongresi (YA/EM 2008)*, İstanbul.