KIRIKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI
DOKTORA TEZİ

Öğrenme-Unutma Etkili ve Ayar Süreli Tek Makine Çizelgeleme Problemleri İçin Yeni Çözüm Yaklaşımları

Settar MUŞTU

OCAK 2020

Endüstri Mühendisliği Anabilim Dalında Settar MUŞTU tarafından hazırlanan ÖĞRENME-UNUTMA ETKİLİ VE AYAR SÜRELİ TEK MAKİNE ÇİZELGELEME PROBLEMLERİ İÇİN YENİ ÇÖZÜM YAKLAŞIMLARI adlı Doktora Tezinin Anabilim Dalı standartlarına uygun olduğunu onaylarım.

Prof. Dr. Süleyman ERSÖZ
Anabilim Dalı Başkanı

Bu tezi okuduğumu ve tezin **Doktora Tezi** olarak bütün gereklilikleri yerine getirdiğini onaylarım.

Prof. Dr. Tamer EREN
Danışman

Jüri Üyeleri

Başkan	: Prof. Dr. Hadi GÖKÇEN
Üye (Danışman)	: Prof. Dr. Tamer EREN
Üye	: Doç. Dr. Talip KELLEGÖZ
Üye	: Dr. Öğr. Üyesi Suna ÇETİN
Üye	: Dr. Öğr. Üyesi Hacı Mehmet ALAKAŞ

13/01/2020

Bu tez ile Kırıkkale Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu Doktora derecesini onaylamıştır.

Prof. Dr. Recep ÇALIN
Fen Bilimleri Enstitüsü Müdürü

Bu tezin hazırlanış sürecinde
fazlasıyla fedakârlık yapan
kıymetli aileme…

ÖZET

ÖĞRENME-UNUTMA ETKİLİ VE AYAR SÜRELİ TEK MAKİNE ÇİZELGELEME PROBLEMLERİ İÇİN YENİ ÇÖZÜM YAKLAŞIMLARI

MUŞTU, Settar
Kırıkkale Üniversitesi
Fen Bilimleri Enstitüsü
Endüstri Mühendisliği Anabilim Dalı, Doktora tezi
Danışman: Prof. Dr. Tamer EREN
Ocak 2020, 190 sayfa

Bu tez çalışmasında, değişken işlem sürelerinin ve beraberinde ayar sürelerinin olduğu iki farklı tek makine çizelgeleme problemi ele alınmıştır. İlk olarak; sıra bağımlı ayar süreleri ve pozisyona bağlı öğrenme etkisi olan toplam gecikme problemi tanımlanmıştır. NP-zor olan bu problemin kesin çözümleri için dal-sınır algoritması, yaklaşık çözümleri için ise iki adet genetik, iki adet de değişken komşu arama algoritması geliştirilmiştir. Dal-sınır algoritmasında kullanılmak üzere polinom zamanda türetilebilen bir alt sınır ve algoritmanın etkinliğini artırmak için iki farklı baskınlık kuralı ortaya konmuştur. Genetik algoritmanın uygunluk fonksiyonuna tavlama benzetimindeki sıcaklık parametresi ve değişken komşu arama algoritmasının karıştırma operatörüne de baskınlık kuralları eklenerek işlevsellikleri artırılmıştır. Rasgele türetilen problem örnekleri ile yapılan uygulama deneyleri neticesinde, çözüm yöntemlerinin performansları değerlendirilmiştir. Deney sonuçları, dal-sınır algoritmasının küçük boyutlu problem örneklerini makul sürelerde çözebildiğini, sezgisel yöntemlerin de yaklaşık çözümleri elde etme konusunda başarılı olduklarını göstermiştir. Bunun yanında, sezgisel yöntemlere uyarlanmış ilave özelliklerin performans üzerindeki olumlu etkileri de gözlemlenmiştir.

İkinci olarak; sıra bağımsız ayar süreleri, zamana bağlı öğrenme etkisi ve yine zamana bağlı unutma etkisi olan maksimum tamamlanma zamanı problemi

tanımlanmıştır. Tanımlanan problemde, bir işe yansıyan öğrenme şiddeti kendisinden önce çizelgelenmiş olan işlerin normal işlem süreleri toplamına bağlı bir fonksiyonla ifade edilmiştir. Her işlemden önce üretimin durmasına yol açan ayar süreleri, unutma etkisinin temel kaynağı olarak ele alınmıştır. Bu nedenle, bir işe yansıyan unutma şiddeti, hem duruştan önceki öğrenme şiddetine hem de o işe ait olan ayar süresine bağlı bir fonksiyonla ifade edilmiştir. İşlem süreleri öğrenme etkisinin, ayar süreleri de unutma etkisinin parametresi olduğundan, iki değişkenli yeni bir öğrenme-unutma modeli tanımlanmıştır. Probleme ait bir takım özellikler sayesinde çözüm karmaşıklığının normal şiddette NP-zor olduğu ıspatlanmıştır. Bu nedenle, kesin çözüm yöntemi olarak tam sayılı doğrusal olmayan programlama modeli ve dinamik programlama algoritması geliştirilmiştir. Önerilen dinamik programlama algoritmasının sözde-polinom zamanlı bir çözüm yöntemi olması, tanımlanan problemin normal şiddette NP-zor olduğunu desteklemiştir. Rasgele türetilen problem örnekleri ile yapılan uygulama deneyleri neticesinde, önerilen her iki yöntemin de etkinliği gösterilmiştir.

Anahtar Kelimeler: Tek Makine Çizelgeleme, Öğrenme Etkisi, Unutma Etkisi, Ayar Süresi, Dal-Sınır Algoritması, Genetik Algoritma, Değişken Komşu Arama Algoritması, Dinamik Programlama, Tam Sayılı Doğrusal Olmayan Programlama, Çözüm Karmaşıklığı

ABSTRACT

NEW SOLUTION APPROACHES FOR SINGLE MACHINE SCHEDULING PROBLEMS WITH LEARNING-FORGETTING EFFECTS AND SETUP TIMES

MUŞTU, Settar
Kırıkkale University
Graduate School of Natural and Applied Sciences
Department of Industrial Engineering, Ph. D. Thesis
Supervisor: Prof. Dr. Tamer EREN
January 2020, 190 pages

In this thesis, two different single machine scheduling problems with variable processing times and setup times are discussed. At first; the total tardiness problem with sequence-dependent setup times and a position-dependent learning effect is defined. Since the problem is NP-hard, a branch&bound algorithm is proposed to obtain exact solutions. Furthermore, genetic and variable neighborhood search algorithms are proposed to obtain approximate solutions. A lower bound which can be calculated in polynomial time and two different dominance rules are employed in the branch&bound. The functionality of heuristics is increased by utilizing dominance rules in the shaking operator of the variable neighborhood search and adapting the temperature parameter of the simulated annealing to the fitness function of the genetic algorithm. Some computational experiments are performed on randomly generated test problems to evaluate the performance of solution approaches. Results show that the branch&bound can optimally solve the small-sized problem samples in a reasonable time and heuristics are successful in obtaining approximate solutions. Furthermore, the positive effects of additional features adapted to heuristics are also observed.

Secondly; the makespan problem with sequence-independent setup times, time-dependent learning effect and time-dependent forgetting effect is defined. The learning amount of a job is represented by a function of the-sum-of-normal-

processing-times of jobs scheduled before. Setup times are assumed to be the reason for forgetting since they lead to interruptions in the production period before each job. Therefore, the forgetting amount of a job is represented by a function of the setup time and existing amount of learning before the interruption. A new bivariate learning-forgetting model is defined since processing times are the parameter of the learning effect and setup times are the parameter of the forgetting effect. By means of problem properties, it is proved that the computational complexity of the defined problem is ordinary NP-hard. Therefore, an integer-nonlinear-programming-model and a dynamic programming algorithm are proposed as exact solution methods. The fact that the proposed dynamic programming is a pseudo-polynomial time algorithm supports the ordinary NP-hardness of the described problem. Some computational experiments with randomly generated problem samples are performed and results indicate the effectiveness of solution methods.

Key Words: Single Machine Scheduling, Learning Effect, Forgetting Effect, Setup Time, Branch&Bound Algorithm, Genetic Algorithm, Variable Neighborhood Search Algorithm, Dynamic Programming, Integer Nonlinear Programming, Computational Complexity

TEŞEKKÜR

Tezimin hazırlanması esnasında hiçbir yardımı esirgemeyen ve biz genç araştırmacılara büyük destek olan, bilimsel deney imkanlarını sonuna kadar bizlerin hizmetine veren, tez yöneticisi hocam, Sayın Prof. Dr. Tamer EREN' e, tez çalışmalarım esnasında, bilimsel konularda daima yardımını gördüğüm Sayın Doç. Dr. Talip KELLEGÖZ' e, Sayın Dr. Öğr. Üyesi Suna ÇETİN' e ve son olarak Dokuz Eylül Üniversitesi' nden yakın arkadaşım Doç. Dr. Şener AKPINAR' a verdiği destek ve motivasyondan dolayı teşekkür ederim.

İÇİNDEKİLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

Çizelge Sayfa

ŞEKİLLER DİZİNİ

Şekil Sayfa

KISALTMALAR DİZİNİ

ATC	Apparent Tardiness Cost
ATCS	Apparent Tardiness Cost With Setups
BA	Bal Arısı Algoritması
BİS	Bulanık İşlem Süreleri
BLAS	Sıra Bağımlı Ayar Süresi
BTT	Bulanık Teslim Tarihleri
BZAS	Sıra Bağımsız Ayar Süresi
ÇMK	Çoklu Müsaitlik Kısıtı
ÇOİ	Eşit Sayıda Birden Fazla Operasyonu Olan İşler
ÇÖ	Çok Ölçütlü Çizelgeleme
DA	Dağınık Arama
DKA	Değişken Komşu Arama Algoritması
DKİ	Değişken Komşu İniş Algoritması
DMH	Değişken Makine Hızları
DP	Dinamik Programlama Algoritması
DRC	Dual Resource Constrained System
DS	Dal-Sınır Algoritması
EDD	Earliest Due Date
EMQ	Economic Manufacturing Quantity
FGZ	Farklı Geliş Zamanları
FMH	Farklı Makine Hızları
GA	Genetik Algoritma
GÇ	Grup Çizelgeleme
GZAS	Geçmiş Zamana Bağlı Ayar Süresi
GZTT	Geçmiş Zamana Bağlı Teslim Tarihi
HAÇ	Hibrid Akış Tipi Çizelgeleme
İA	İteratif Arama
İM	İki Farklı Müşteri
İR	İş Reddetme
KI	Karmaşıklık İspatı

KİM	Makineler İçin Kesintisiz İşlem
Kİİ	İşler İçin Kesintisiz İşlem
KKE	Kesikli Kademeli Evrim Algoritması
KKO	Karınca Kolonisi Optimizasyonu
KT	Kaynak Tahsisi
LFCM	Learn-Forget Curve Model
LFL	Learn-Forget-Learn Model
MM	Matematiksel Model
NP	Polinom Zamanlı Olmayan Problemler
NP-tam	Polinom Zamanlı Olmayan Karar Problemleri
NP-zor	Polinom Zamanlı Olmayan Optimizasyon Problemleri
OA	Operatör Atama
OAÇ	Orantılı Akış Tipi Çizelgeleme
ODT	Orta Değer Teoremi
OT	Ortak Teslim Tarihi
ÖDO	Özel Durumda Optimizasyon
Öİ	Özdeş İşler
ÖOM	Özdeş Olmayan Makineler
P	Polinom Zamanlı Problemler
PID	Power Integration Diffusion
PSO	Parçacık Sürüsü Optimizasyonu
PZ	Polinom Zamanlı Algoritma
RC	Recency Index
SPT	Shortest Processing Time
TA	Tabu Arama Algoritması
TAA	Teslim Tarihi Aralığı Atama
TB	Tavlama Benzetimi Algoritması
TDOP	Tam Sayılı Doğrusal Olmayan Programlama Modeli
TTA	Teslim Tarihi Atama
TZ	Tamirat Zamanı
UE	Unutma Etkisi
VRVF	Variable Regression Variable Forgetting
YE	Yaşlanma Etkisi

1. GİRİŞ

Çizelgeleme teorisi, teorik bilgisayar biliminin bir parçası olarak işlerin makinelere belirli bir zaman aralığında atanmasını ifade eden ve optimum çizelgelerin aranmasıyla ilgilenen bir çalışma alanıdır. Çizelgeleme, üretim ve servis sistemlerinde kritik bir role sahip, önemli bir karar verme sürecidir. Çizelgeleme operasyonel olarak, matematiksel teknikleri ve sezgisel yöntemleri kullanarak, kısıtlı kaynakların çeşitli görevlere uygun biçimde dağıtılmasını sağlar. Tipik bir çizelgeleme problemi, işlem görecek olan bir grup işi, bu işlemleri yapacak olan bir grup makineyi ve işlerin oluşturduğu olurlu bir çizelgenin kalitesini ifade eden bir amaç fonksiyonunu tüm yönleriyle tanımlayarak formülüze edilebilir. Bu problemde ulaşılmak istenen hedef, tanımlanan problemin belirttiği şartları ve kısıtları ihlal etmeden amaç fonksiyonunu optimize eden çizelgenin mümkün olan en kısa zamanda bulunmasıdır. Çizelgeleme teorisi, 1950 yılından bugüne yoğun bir şekilde geliştirilmiştir [1–3]. O zamandan beri, yönetim, üretim, lojistik, taşımacılık ve bilgisayar sistemlerinde sayısız pratik uygulamaları yapılarak büyük ilgi görmüştür [4, 5]. Yöneylem araştırmacıları, endüstri mühendisleri ve yöneticiler çalışma alanındaki çeşitli aktivitelerin yönetim problemiyle karşılaşmışlardır. Bu problemleri aşmak için geliştirilen etkili çizelgeleme yöntemleri üretim maliyetlerinin azalmasına ve firmanın rekabet gücünü korumasına yardımcı olmuştur. Bilgisayar uzmanları 1960' ların sonunda işletim sistemleri geliştirirken bir takım çizelgeleme problemlerini aşmak zorunda kalmışlardır. O günlerde verimli donanımsal kaynaklar oldukça kıt olduğundan, mevcut donanımların verimli kullanımı uygulamaların çalıştırılması için gereken maliyeti düşürmüştür [4].

Üretim endüstrisinde, çizelgeleme problemlerinin yapısını belirleyen ana unsur genellikle atölye tipi olmaktadır. Tüm süreçleri olumsuz etkileyebilen makine darboğazları, üretim sistemlerinde zaman zaman karşılaşılan bir durumdur. Bu darboğazların doğru yönetimi firmalar için oldukça kritik bir süreçtir. Çünkü firmalar, müşteri siparişlerinin teslim edilmesinde oluşacak gecikmelerden dolayı sorumludurlar [6]. Dolayısıyla, üretim süreçlerinde hedeflenen çıktıyı elde etmek için ve ayrıca çok makineli atölye tiplerinde darboğazları da ifade ettiği için, tek makine

çizelgeleme üzerinde dikkatle çalışılan önemli bir problem olmaya devam etmektedir. Üretim süreçlerinde hedeflenen çıktı, sistemden beklenen performans değerine göre değişiklik gösterir. Tedarik zinciri yönetimi çerçevesinde firmalar, müşterilerine taahhüt edilen teslim tarihlerine uymak konusunda hassasiyet gösterirler. Taahhüt edilen bu tarihlerde herhangi bir aksama meydana geldiğinde, temeli müşteri ilişkilerine dayanan çeşitli maliyetlerle karşılaşmak, firmalar için kaçınılmaz olmaktadır. Serbest piyasa koşullarında, verilen taahhütlerin yerine getirilmesi önemli olduğundan, teslim tarihlerine uymayı önceleyen performans göstergeleri önem kazanmaktadır. Dolayısıyla, toplam ağırlıklı gecikmenin minimizasyonu, geride bıraktığımız yıllarda çoğu araştırmacının dikkati çeken önemli bir performans göstergesi olmuştur [7, 8].

Makine çizelgeleme problemleri bazı yapısal kısıtlar, öncelik ilişkileri ve özel durumları içerebilir. Ayar süreleri, çizelgeleme problemlerinde dikkate alınan özel durumlardan biridir [9]. Ayar sürelerinin ihmal edilmesi veya işlem sürelerinin bir parçası olarak kabul edilmesi, çizelgeleme problemlerinin çoğunda karşılaşılan bir durumdur. Bu varsayımlar, gerçek uygulamalar için hazırlanan çizelgeleme modellerinin bazılarında kaliteli çözüm elde etmeyi kolaylaştırırken, bazılarında çözüm kalitesini olumsuz etkiler. Çözüm kalitesinin olumsuz etkilendiği bu gibi uygulamalar için geliştirilen modellerde, ayar süreleri de ayrı olarak ele alınmalıdır [10]. Gerçek üretim sistemlerinde, ardışık işler arasındaki ayar süreleri farklılık gösterebilir. Bu şekilde olan ayar sürelerine sıra bağımlı ayar süreleri (BLAS) denir ve bir önceki işe göre değeri değişebilir [11]. Panwalkar vd.' ne [12] göre, çizelgeleme konusunda çalışanların %70' i tanımlanan operasyonların en az birkaç tanesi için, kalan %30' u ise tanımlanan tüm operasyonlar için BLAS' nin geçerli olduğunu belirtmiştir [13].

Çizelgeleme problemleri, üretim kaynaklarının birden fazla ürün tarafından kullanılması sonucu meydana gelir. Bu problemlerin çözümüne yönelik geliştirilen karar verme süreçleri, var olan kısıtları ihlal etmeden, belirli zaman aralıklarının kullanıcılara atanmasını uygulanabilir biçimde sağlayan işlemlerden oluşur [14]. Dolayısıyla, işlem sürelerini sabit kabul etmek, birçok çizelgeleme probleminde karar verme sürecinin hassasiyetini olumsuz etkiler. İşçiler, aynı işi çok kez yapıp

tecrübe kazanarak, iş üzerindeki kabiliyetlerini geliştirdiklerinden dolayı işçi performansındaki bu artış, işlem sürelerinde azalmaya yol açar [15]. Çizelgeleme literatüründe kabul görmüş bu olgu öğrenme etkisidir [16–18]. Gelişim fonksiyonu olarak da bilinen öğrenme eğrisi, işlem tekrarından sağlanan verimlilik artışını ifade eder. Öğrenme etkisi genellikle, benzer ürünler için gerçekleşen maliyet bilgisinin istatistiksel analizi ile belirlenir. Azalan maliyet ile ilgili olarak; tekrar eden elektriksel operasyonlarda %75-85, gemi inşasında %80-85, tekrar eden elektronik üretiminde %90-95 ve satın alma operasyonlarında %85-88 gibi oranlar tespit edilmiştir [19]. Yelle [20] geniş çaplı çalışmasında, endüstrinin farklı alanlarına ait sayısız pratik durumlarda ve çeşitli kurumsal faaliyetlerde öğrenme etkisinin olduğunu gözlemlemiştir. Öğrenme etkisi, makine çizelgeleme problemlerinde ilk olarak Biskup [21] tarafından yakın geçmişte kullanılmasına rağmen, çizelgeleme literatüründe oldukça dikkat çekmiştir.

Üretim sürecindeki çeşitli sebeplere bağlı olarak meydana gelen ve unutma etkisi olarak bilinen, öğrenmenin tersi bir olgu daha vardır. Öğrenme ve unutma, üretim planlamada, çizelgelemede, hatta fiyatlandırmada ve maliyet tahmininde önemli ve etkili parametreler olduğundan dolayı, unutma etkisinin daha net ifade edilmesi ve anlamlaştırılması gerekmektedir [22]. Gerçek üretim sistemlerinin çoğu, çeşitli yapısal özellikler ve özel durumlardan dolayı genellikle kesintiye uğrar. Bazı planlamacılar bu durumu göz ardı etse de, bu kesintiler performans ölçütlerinde sapmaya neden olur. Kesintilere bağlı olarak öğrenme ile birlikte gerçekleşen unutma etkisi, performans ölçütlerinde meydana gelen bu sapmanın nedenlerinden biridir. Buradan yola çıkarak, bir işleme başlamadan önceki ayar sürelerinin de unutmaya yol açtığı düşünülebilir, çünkü ayar süreleri de üretim sürecini bölen duraksamalardır [23]. Bu nedenle, öğrenme ve unutma etkisi içeren çizelgeleme problemlerini yüksek hassasiyetle çözebilmek için ayar sürelerinin başlıca bir problem parametresi olarak dikkate alınması gerekmektedir.

Bu tezde, yukarıda anlatılan tüm unsurların dikkate alındığı iki farklı çizelgeleme problemi ele alınmıştır. İlk olarak, işlem süreleri üzerinde öğrenme etkisi olan ve her işlemden önce BLAS bulunan tek makine çizelgeleme probleminde toplam gecikmenin minimizasyonu ele alınmıştır. Tanımlanan problem için bir takım

baskınlık kuralları ve alt sınır belirlenmiştir. Bu bilgiler ışığında belirli boyuta kadar optimum çözümlerin bulunabildiği bir dal-sınır algoritması (DS) geliştirilmiştir. Zaman ve hafıza kısıtları nedeniyle büyük boyutlu problemler için sezgisel çözüm yöntemlerine başvurulmuştur. Büyük boyutlu problemlerin yaklaşık çözümlerini elde etmek için genetik algoritma (GA) ve değişken komşu arama algoritması (DKA) tasarlanmış ve problem özelliklerine uygun çalışmalarını sağlamak için bir takım deneysel çalışmalar yapılmıştır. Her iki sezgisel yöntem de bazı operatörleri bakımından farklılaştırılarak iki adet GA ve iki adet DKA önerilmiştir. Bu problem kapsamında tezin literatüre katkıları aşağıdaki gibidir.

- ✓ BLAS ve işlem süreleri üzerinde öğrenme etkisi bir arada ele alınmıştır.
- ✓ Tanımlanan alt sınır ve baskınlık kuralları vasıtasıyla, küçük boyutlu problemlerin optimum çözümlerini makul sürelerde elde edebilen etkin bir DS geliştirilmiştir.
- ✓ Büyük boyutlu problemlerin yaklaşık çözümlerini makul sürelerde elde edebilen etkin sezgisel yöntemler (GA ve DKA) geliştirilmiştir.
- ✓ DKA için baskınlık kurallarını içeren etkin bir karıştırma operatörü ve GA için adaptif değerlendirme yapma imkanı sağlayan etkin bir uygunluk fonksiyonu önerilmiştir.

İkinci olarak, işlem süreleri üzerinde öğrenme ve unutma etkisi olan ve her işlemden önce sıra bağımsız ayar süresi (BZAS) bulunan tek makine çizelgeleme probleminde maksimum tamamlanma zamanının minimizasyonu ele alınmıştır. Bu problemde iki değişkenli bir öğrenme-unutma modeli tanımlanmıştır. Her iki özelliği barındıran modelin öğrenme faktörü önceki işlerin işlem süreleri toplamıyla ifade edilirken, unutma faktörü ayar süresiyle ifade edilmiştir. Bu özellikler neticesinde yeni bir problem yapısı elde edilmiştir ve bu yüzden, tanımlanan problemin çözüm karmaşıklığı ilk kez bu tezde belirlenmiştir. Optimum çözüme ait bir takım özellikler ortaya çıkarılmış ve bu özelliklerden yola çıkılarak, karmaşıklık teorisinde belirtilen yöntemlerle problemin NP-zor olduğu gösterilmiştir. Buradan hareketle, optimum çözümleri elde etmek için tam sayılı doğrusal olmayan programlama modeli (TDOP) ve dinamik programlama algoritması (DP) geliştirilmiştir. Bu problem kapsamında tezin literatüre katkıları aşağıda belirtilmiştir.

- ✓ Unutma etkisinin oluşma nedenlerine dayanarak, ayar süreleri unutma fonksiyonunun bağımsız değişkeni olarak kullanılmıştır.
- ✓ Bir çizelgeleme probleminde ilk defa zamana bağlı olan 2 değişkenli öğrenme-unutma modeli kullanılmıştır.
- ✓ Tanımlanan problemin optimum çözümüne ait bir takım temel özellikler ortaya çıkarılmıştır.
- ✓ Benzer yapıdaki problemlerin aksine, tanımlanan problemin NP-zor (normal şiddette) olduğu tespit edilerek çözüm karmaşıklığı belirlenmiştir.
- ✓ Tanımlanan problemin optimum çözümlerini elde etmek için TDOP ve sözde-polinom zamanlı DP önerilmiştir.

Tezin devamında, tüm konular ayrılmış bölümlerde detaylı bir şekilde anlatılmıştır. Bölüm 2' de, genel hatlarıyla çizelgeleme teorisi, çizelgeleme problemi, algoritma kavramı, çözüm kavramı ve karmaşıklık teorisi anlatılmıştır.

Bölüm 3' de, çalışmanın dayandığı iki temel yapı taşı olan değişken işlem süresi ve ayar süresi kavramları literatürdeki örnekleriyle detaylı bir şekilde anlatılmıştır Değişken işlem süresi kapsamında öğrenme ve unutma etkileri, ayar süresi kapsamında ise sıra bağımlı ve sıra bağımsız ayar süreleri incelenmiştir.

Bölüm 4' de, literatür taraması verilmiştir. Farklı alt bölümlerde olmak üzere, öğrenme ve unutma etkili makine çizelgeleme problemleri incelenmiştir. Devamında, yine farklı bir alt bölümde ayar süreli toplam gecikme problemleri ve son alt bölümde raporlanan literatüre ait bir değerlendirme yapılmıştır.

Bölüm 5' de, yukarıda bahsedilen çizelgeleme problemlerinin tüm unsurları, kısıtları ve yapısal özellikleri matematiksel olarak tanımlanmıştır. Her iki problemde kullanılan ifadeler için ortak notasyon belirlenmiştir.

Bölüm 6' da, her iki problem için geliştirilen çözüm yöntemleri verilmiştir. İlk problemin optimum çözümleri için geliştirilen DS' nın operatörleri, aşamaları ve çalışma şekli detaylı olarak anlatılmıştır. DS' da kullanılmak üzere alt sınır ve baskınlık kuralları ifade edilmiştir. Beraberinde, yaklaşık çözümler için geliştirilen

GA’ lar ve DKA’ ları için önerilen operatörlerin ve fonksiyonların tanımı, uygulama şekli ve birbirleriyle olan ilişkileri tüm yönleriyle ele alınmıştır. İkinci problemin çözüm karmaşıklığı ispatlanmıştır ve buna bağlı olarak, optimum çözümleri elde etmek için geliştirilen TDOP ve DP’ nın modelleri ve formülasyonları verilmiştir.

Bölüm 7’ de, her iki problem için geliştirilen çözüm yöntemlerinin performans değerlendirmesi yapılmıştır. İlk problemin sezgisel yöntemleri için bir takım parametre ayarlamaları yapılmıştır. Her iki problem için küçük ve büyük boyutlarda ayrı problem setleri oluşturulmuş ve tüm yöntemler için uygun olan problem örneklerinin çözümleri elde edilmiştir. Bulunan sonuçlar ilgili çizelgelerle kıyaslamalı olarak verilmiştir.

Bölüm 8’ de, tezin genel değerlendirmesi ve uygulamalar neticesinde gözlemlenen bulgular ifade edilmiştir. Son olarak, sonuçlara dayanan önerilerden ve gelecek çalışmalardan bahsedilmiştir.

2. ÇİZELGELEME

Çizelgeleme, yapılması gereken işlemlerin tam olarak gerçekleşmesi için uygun bir sıralama oluşturulması ve bu sıralamanın bazı kısıtlar altında çeşitli performans ölçütlerini iyilemesi faaliyetidir. Belirli zaman aralıklarının kısıtlar göz önünde bulundurularak kullanıcılara makul bir şekilde atanması, çizelgeleme problemlerinin özelliklerini belirler. Çizelgeleme için literatürde birçok farklı tanım, açıklama ve ifade bulunmaktadır. Örneğin, Coffman ve Bruno' ya [24] göre bir çizelge, işlem yapmaya müsait ayrı zaman dilimlerine her bir işin atanmasıyla oluşan bir sıralamadır. Bu tanıma göre, atanan işlere ait bir takım özellikler çizelgenin yapısını belirleyen ana unsurlardır. Gawiejnowicz' in [25] tanımına göre uygulanabilir bir çizelge, makinelerin (kaynakların) işlem yapmaya müsait zamanlarını yapılacak işler için rezerve etmektir. Bu tanımda ise, işlem yapmaya müsait zamanlarda geçerli olmak üzere, makinelere (kaynaklara) ait bir takım özellikler çizelgenin yapısını belirleyen ana unsurlardır.

Görüldüğü üzere çizelgeleme için yapılan tanımların hepsi kendine özgü durumlarla açıklanabilmektedir ve en tamamlayıcı tanım, üzerinde çalışılan problemin modeliyle yapılabilmektedir. Bu nedenle, Pinedo' nun [5] belirttiği şekliyle, genel üretim ortamını ve çizelgeleme modelinin bu ortamdaki rolünü tanımlamak daha açıklayıcı olacaktır. Bir üretim ortamındaki çizelgeleme modelini tanımlamak için sipariş olarak gelen bilgilerin öncelikle işlem süreleri, ağırlıkları (önem dereceleri), öncelik ilişkileri, ayar süreleri, geliş zamanları ve teslim tarihleri gibi özellikleri belirlenmiş işlere dönüştürülmesi gerekmektedir.

İşler genellikle üretim ortamındaki makineler vasıtasıyla, belirlenen sıraya göre işlem görmelidirler. Öyle ki, çeşitli nedenlerden dolayı herhangi bir işin tamamlanması gecikebilir ve işlemin bir kısmı veya tamamı önceden belirlenen teslim tarihiden sonraya sarkabilir. Makine arızaları, kapasite yetersizliği, önem derecesine göre bazı işlere öncelik tanınması, işin yapılma sırasına veya zamanına bağlı değişen işlem süreleri gibi planlı veya beklenmeyen şekilde gerçekleşen olaylar gecikmeye neden olan unsurlardan bazılarıdır. Dolayısıyla, bir üretim sistemine ait çizelgeleme modeli

benzer unsurları gözeterek hazırlandığında daha etkili ve tutarlı olacaktır. Modelin etkinlik ve tutarlılığı ise, modele ait performans ölçütünün gerçekleşen durumdan sapmasına bağlıdır.

Çizelgelemede yaygın olarak kullanılan performans ölçütleri üç farklı grupta toplanmıştır. Bunlar çizelgenin değişik yönlerini ifade eden; tamamlanma süresi, zamanında tamamlanma ve çıktı miktarıdır [26]. Çizelgeleme modelinin performans ölçütü, ilgili üretim sistemi bileşenlerinin tüm niteliklerini (modelin yapısını belirleyen özellikler) kapsayacak yapıda olmalıdır. Ancak, gerçek endüstride bu niteliklerin tam olarak tespit edilip istenen hassasiyetle ifade edilmesi çoğu zaman mümkün olmamaktadır. Bu durumda uygulanan en pratik yöntem, bu niteliklere yönelik varsayımlar belirlemektir. Gerçekte, çizelgeleme modeline kolayca adapte edilebilecek temel özellikler göz önünde bulundurulurken, performans ölçütüne bağlı ve izole edilmesi güç olan diğer özellikler bazı temel varsayımlarla kullanılabilir hale getirilmektedir. Örneğin, bazı sistemlerde malzeme taşıma süreleri ayrı bir unsur olarak modele eklenirken, bazı sistemlerde işlem sürelerine eklenerek kullanılabilmektedir. Benzer şekilde, üretim sürecindeki duruşların bazı sistemlerde modele eklenmesi gerekirken, bazı sistemlerde göz ardı edilmesi mümkün olabilmektedir. Görüldüğü üzere, karar verme sürecinde performans ölçütünün beklenen değerden sapmasına yol açan durumlardan biri de modelin yapısı ve kapsamıdır. Bu bölümde, çizelgeleme için yapılan genel tanımlar ve çizelgeleme problemlerinin özellikleri Gawiejnowicz' in [27] belirttiği gibi ele alınmıştır.

2.1. Çizelgeleme Problemlerinin Temel Özellikleri

Problem (Π), değerleri belirtilmemiş çeşitli parametreler veya bağımsız değişkenler içeren ve cevaplanması gereken genel bir sorudur. Bir problem, parametreleriyle ilgili bir takım tanımlamalar ve çözümün (sonucun) sağlaması gereken çeşitli özelliklere ait açıklamalarla ifade edilmelidir. Bir problem örneği ($\mathcal{I} \in \Pi$) ise, tanımlanan problem parametrelerine belirli uygun değerlerin atanmasıyla elde edilir. Bu tezde iki çeşit problem dikkate alınmıştır. Bunlardan biri karar problemleri, bir diğeri de optimizasyon problemleridir. Karar problemlerinde aranan çözüm,

problemin tanımında belirtilen özellikleri birebir yansıtan bir çözümdür [4]. Optimizasyon problemlerinde aranan çözüm, problem için tanımlanmış olan amaç fonksiyonunu optimize eden bir çözümdür. Optimizasyon ve karar problemleri birbirleriyle yakından ilgilidir. Mesela, bir optimizasyon probleminde amaç fonksiyonunu minimize eden çözüm aranırken, aynı özellikteki karar probleminde ise, amaç fonksiyonunu belirli bir eşik değerine eşit veya eşik değerinden daha küçük yapan bir çözümün olup olmadığı araştırılır.

Bir optimizasyon problemi olan çizelgeleme problemi Π, tanımlandığı ortamdan bağımsız olarak yapısal özelliklerini yansıtan dört temel simgeyle $\Pi = (N, M, R, \gamma)$ ifade edilebilir. Buradaki N tamamlanması gereken faaliyetlerin oluşturduğu bir seti, M faaliyetleri gerçekleştirecek olan işlem merkezlerinin oluşturduğu bir seti, R faaliyetlerin tamamlanması için işlem merkezlerinden başka gerekli olan diğer tüm kaynakların oluşturduğu bir seti ve γ probleme ait çözümlerin kalite düzeyini belirleyen bir fonksiyonu ifade etmektedir.

2.1.1. Faaliyet Setinin Özellikleri

Faaliyet setini N oluşturan elemanlar iş olarak adlandırılır. Bu tezde, aksi belirtilmediği sürece, $|N| = n$ olarak kabul edilmiştir. İşler $N(1), N(2), \cdots, N(n)$ ile ve herhangi bir $N(j)$ işi de j ile ifade edilmiştir. Bir j işinin $1 \leq j \leq n$ operasyon sayısının n_j olduğu varsayımıyla, operasyonlar $O_{1,j}, O_{2,j}, \cdots, O_{n_j,j}$ ile ifade edilmiştir. Eğer operasyon sayısı 1 ise, j işinin operasyonu için O_j kullanılabilir. Her operasyon için önceden belirlenmiş bir işlem süresi bulunmaktadır. Bir işin $1 \leq j \leq n$ operasyonlarına $1 \leq i \leq n_j$ ait işlem süreleri p_{ij} $(p_j) \geq 0$ ile gösterilmektedir. Bir j işi için $1 \leq j \leq n$, bir geliş zamanı $r_j \geq 0$, bir teslim tarihi $d_j \geq 0$ ve bir ağırlık $w_j \geq 0$ tanımlanmış olabilir. Bu j işinin ilk operasyonu $O_{1,j}$ geliş zamanından r_j önce başlayamaz ve son operasyonun $O_{n_j,j}$ teslim tarihinden d_j önce tamamlanması istenir. Ağırlık w_j diğer işlere nispeten j işinin önemini belirtmektedir. Tanımlanan Π probleminde, eğer $\forall j \in N$ için $r_j = 0$ ise, tüm işler $t = 0$ anında hazır bulunmaktadır ve eğer $\forall j \in N$ için $w_j = 1$ ise, işlerin önem derecesi eşittir.

Bir j işi için $1 \leq j \leq n$, bir ayar süresi $s_j \geq 0$ veya $s_{ij} \geq 0$ tanımlanmış olabilir. Ayar süresi, bir işe ait operasyonun ilgili makinede işleme başlamadan önce yapılması gereken ayar faaliyetleri için gereken süreyi temsil eder. Bir j işinin i işlem merkezindeki $O_{i,j}$ operasyonuna ait ayar faaliyetleri tamamlanmadan, $O_{i,j}$ operasyonu başlayamaz. Ayar süresi, bir önceki tamamlanan işe (operasyona) göre değişkenlik gösteriyorsa, s_{ij} gibi iki indisli gösterimle ifade edilir. Bu gösterimde i tamamlanan işi (operasyonu), j ise başlayacak olan işi (operasyonu) temsil eder.

Bunun yanında, bazı işlerin diğerlerinden önce yapılması gerektiğini yansıtan öncelik ilişkileri olabilir. Öncelik ilişkileri, işlerden oluşan kısmi sıralamalarla ifade edilir $\prec \subseteq N \times N$. Öncelik ilişkileri zincirler, ağaç gösterimi, tek yönlü şebeke gibi birçok farklı formda ifade edilebilir [27].

Bazı çizelgeleme problemlerinde iş kesme üstünlüğünden de bahsedilebilir. Eğer bir problemde iş kesme üstünlüğü söz konusu ise, bir işin işlemine herhangi bir anda ara verilip bir başka işin işlemine başlanabilir. İşlemi bölünen işler ekstra bir maliyete maruz kalabilir. Bu maliyet unsuru tanımlanan problem için ölçülebilir bir niteliktir. Yani bazı durumlarda gereken işlem süresinin uzamasına yol açarken, bazı durumlarda da fazladan ayar işlemlerinin yapılması gerekebilir.

Çizelgeleme literatüründe, işlere ait operasyon sürelerinin değişkenlik gösterdiği durumlar da bulunmaktadır. Bu gibi durumlarda değişken işlem süresi söz konusudur. Bir j işinin $1 \leq j \leq n$ operasyonlarına $1 \leq i \leq n_j$ ait işlem süreleri p_{ij} ($n_j = 1$ için p_j) önceki işlerin tamamlanması için harcanan (gereken) zamana göre, işlem sırasına göre, her ikisine göre veya bunlara bağlı tanımlanan çeşitli fonksiyonlara göre değişkenlik gösterebilir. Bu değişkenliğin yönü ve şiddetine göre çeşitli tanımlamalar yapılmıştır. Buna göre, işlem süresinin azalma eğilimindeki değişimi öğrenme etkisiyle, artma eğilimindeki değişimi ise unutma etkisi veya yaşlanma etkisi ile ifade edilmiştir. Değişimin şiddeti matematiksel olarak belirtilir ve problem yapısını etkileyen önemli bir özelliktir.

2.1.2. İşlem Merkezi Setinin Özellikleri

İşlem merkezi setini M makine, operatör, istasyon, işlemci, tesis vb. gibi problem yapısına göre belirlenen elemanlar oluşturur. Bu tezde, aksi belirtilmediği sürece işlem merkezi setinin makinelerden oluştuğu ve $|M| = m$ olduğu kabul edilmiştir. Makineler $M(1), M(2), \cdots, M(m)$ ile ve herhangi bir $M(i)$ makinesi de i ile ifade edilmiştir.

Eğer $m = 1$ ise, tek makine çizelgeleme problemi söz konusudur. Tek makine çizelgeleme problemi en yalın durumu ifade etse de, metnin devamında belirtilen karmaşık atölye ortamlarının en küçük yapı taşı olduğu için dikkate değer bir problem yapısıdır. Yani, her atölye ortamından tek makine çizelgeleme problemi türetilebilir veya mevcut problem tek makine problemine indirgenebilir.

Eğer $m \geq 2$ ve tüm makineler aynı özelliklere sahip ise, özdeş paralel makine çizelgeleme problemi söz konusudur. Bu problemde, bir j işine ait sadece O_j operasyonu vardır ve bu operasyon herhangi bir makinede p_j süreyle yapılır. Eğer makinelerin işlem hızları farklı ve bu farklılık işe göre değişmiyor ise, uniform paralel makine çizelgeleme problemi söz konusudur. Bu problemde, bir j işine ait sadece O_j operasyonu vardır ve bu operasyon, $1 \leq i \leq m$ için işlem hızı v_i olan i makinesinde p_j/v_i süreyle yapılır. Eğer makinelerin işlem hızları farklı ve bu farklılık işlere göre değişiyor ise, ilişkisiz paralel makine çizelgeleme problemi söz konusudur. Bu problemde, bir j işine ait sadece O_j operasyonu vardır ve bu operasyon, $1 \leq i \leq m$ için i makinesinde p_{ij} süreyle yapılır. İlişkisiz paralel makine çizelgeleme probleminde, makine hızları tüm işlere göre farklı olduğu için tüm işlerin tüm makinelerdeki işlem süreleri problem girdisi olarak belirtilmelidir.

Yukarıda anlatılanlar bir işin sadece bir operasyona sahip olduğu ve tüm makinelerin bu operasyonu yapabilecek kabiliyette olduğu durumu ifade etmektedir. Eğer mevcut makineler aynı işlemleri yapamayacak özellikte farklı fonksiyonlara sahip ise, işlere ait operasyonlara göre tahsis edilmiş makineler ve dolayısıyla akış tipi, iş atölyesi tipi veya açık atölye tipi çizelgeleme problemi söz konusudur. Bu tip atölye ortamlarında

bir işin birden fazla operasyonu vardır ve her operasyon farklı makinelerde yapılmaktadır. Akış tipi çizelgeleme probleminde $M(1), M(2), \cdots, M(m)$ olmak üzere $m \geq 2$ adet makine ve her j işine ait $O_{1,j}, O_{2,j}, \cdots, O_{n_j,j}$ sırasıyla yapılmak üzere $n_j = m$ adet operasyon bulunur. Akış tipi atölyede $O_{i,j}$ operasyonundaki i indisi hem operasyonun hangi makinede yapılacağını, hem de j işinin operasyon sırasını ifade eder. Yani, j işinin $r-1$ makinesindeki $O_{r-1,j}$ operasyonu tamamlanmadan, r makinesindeki $O_{r,j}$ operasyonu başlayamaz ve operasyonların yapılmasına dair belirlenen bu sıralama tüm işler için aynıdır ($j \in N$, $2 \leq r \leq n_j = m$, $i \in M$).

Açık atölye tipi çizelgeleme probleminde de $m \geq 2$ ve $\forall j \in N$ için $n_j = m$ olur. Akış tipi atölyede olduğu gibi, $O_{i,j}$ operasyonundaki i indisi operasyonun hangi makinede yapılacağını ifade eder. Ancak, j işinin operasyon sırasını ifade etmez. Dolayısıyla açık atölyede, işlere ait operasyonların hangi sırayla yapılacağı konusunda belirlenmiş bir kural bulunmaz ve dolayısıyla, tüm işler için operasyonlar herhangi bir sırada yapılabilir.

İş atölyesi tipi çizelgeleme probleminde $m \geq 2$ ancak, $\forall j \in N$ için $n_j = m$ olmak zorunda değildir. Tüm işler, kendine özgü belirlenmiş bir operasyon sırasına göre işlem görür. Bir iş bazı makinelerde işlem görmeyebilir ve bazı makinelerde birden fazla kez işlem görebilir. Bu nedenle, her işin operasyon sayısı farklı ve makine sayısından fazla veya az olabilir.

2.1.3. Kaynak Setinin Özellikleri

Bazı problemlerde, işlerin tamamlanması makinelerden başka ek varlıklar da gerektirebilir. Bu tip varlıklara kaynak (R) adı verilir. Kaynaklar sürekli veya kesikli olabilir. Bir işe gerçek sayılarla ifade edilebilir miktarda kaynak tahsisi yapılabiliyorsa, kaynak süreklidir. Bir işe yalnızca negatif olmayan tam sayılarla ifade edilebilir miktarda kaynak tahsisi yapılabiliyorsa, kaynak kesiklidir. Bunun yanında, kaynaklar sınırlı (kısıtlı) veya sınırsız da olabilir.

2.1.4. Amaç Fonksiyonunun Özellikleri

Bir çizelgeleme probleminde, işlerin planlama periyodundaki durumlarını temsil eden bazı özellikleri vardır. Bu özellikler, $\forall j \in N$ için başlama zamanı (ST_j), bekleme zamanı (WT_j), tamamlanma zamanı (C_j), akış zamanı (F_j), erken tamamlanma (E_j), geç tamamlanma (L_j), gecikme (T_j) ve geciken iştir (U_j). Belirtilen bu özelliklerin matematiksel ifadesi tanımlanan probleme göre değişiklik gösterebilir. Bundan dolayı, birçok çalışmada kabul edilen bazı genel varsayımlar dikkate alınmıştır [28].

1. Tüm işler, bölünmeyen ve çakışmayan bir zaman aralığında sadece bir makinede işlem görebilir.
2. Her makine, bölünmeyen ve çakışmayan bir zaman aralığında sadece bir işle meşgul olabilir.
3. Eğer bir işin herhangi bir makinede operasyonu başlamış ise, operasyon tamamlanana kadar işlem devam eder.
4. Tüm makineler planlama periyodu boyunca işlem yapmaya müsaittir, yani arıza ve bakım-onarım faaliyeti bulunmamaktadır.
5. Kısıtlı kaynak sadece makinedir, bunun dışında operatör, araç ve gereç, malzeme, takım gibi kaynaklar sınırsızdır.
6. İşlem süreleri, bir iş için gerekli olan ayar faaliyeti dışında diğer tüm faaliyetlere (taşıma, istifleme vs.) ait zamanları da içermektedir.
7. İşler ardışık olarak sıralanmış operasyonlarla ifade edilir, yani montaj işlemi barındırmamaktadır.

Bu varsayımlar doğrultusunda, t bir önceki işin tamamlanma zamanı olmak üzere başlama zamanı $ST_j \geq \max\{r_j, t + s_j\}$ ve bekleme zamanı $WT_j = ST_j - \max\{r_j, t + s_j\}$ ile hesaplanır. Tamamlanma zamanı $C_j = ST_j + p_j = \max\{r_j, t + s_j\} + WT_j + p_j$ olur, çünkü 6 numaralı varsayıma göre bir iş ya makinede işlem görür, ya da işlem kuyruğunda bekler. Bir işin sistemde kalma süresini ifade eden akış zamanı $F_j = WT_j + p_j = C_j - \max\{r_j, t + s_j\}$ formülüyle, erken tamamlanması $E_j = \max\{d_j - C_j, 0\}$ formülüyle, geç tamamlanma zamanı $L_j = C_j - d_j$ formülüyle,

gecikmesi $T_j = \max\{0, L_j\}$ formülüyle ve geciken iş $U_j = \begin{cases} 0, & \text{eğer } L_j \leq 0 \text{ ise} \\ 1, & \text{eğer } L_j > 0 \text{ ise} \end{cases}$ formülüyle belirlenir. İşler için hesaplanabilen bu değerler çerçevesinde, maksimum değerin ölçüt olduğu maksimum tamamlanma zamanı (C_{max}), maksimum bekleme zamanı (WT_{max}), maksimum akış zamanı (F_{max}), maksimum erken tamamlanma (E_{max}), maksimum geç tamamlanma zamanı (L_{max}) ve maksimum gecikme (T_{max}) minimizasyonu amaç fonksiyonu olarak dikkate alınmaktadır. Genel bir ifadeyle $\zeta \in \{C, WT, F, E, L, T\}$ olduğuna göre, maksimum formdaki bu amaç fonksiyonları denklem 2.1 ile hesaplanır.

$$\zeta_{max} = \max_{j \in N}\{\zeta_j\} \tag{2.1}$$

İşlere ait bu değerlerin toplamının ölçüt olduğu toplam tamamlanma zamanı ($\sum_j C_j$), toplam akış zamanı ($\sum_j F_j$), toplam bekleme zamanı ($\sum_j WT_j$), toplam erken tamamlanma ($\sum_j E_j$), toplam geç tamamlanma zamanı ($\sum_j L_j$), toplam gecikme ($\sum_j T_j$) ve toplam geciken iş sayısı ($\sum_j U_j$) minimizasyonu amaç fonksiyonu olarak dikkate alınmaktadır. Genel bir ifadeyle $\xi \in \{C, WT, F, E, L, T, U\}$ olduğuna göre, toplam formdaki bu amaç fonksiyonları denklem 2.2 ile hesaplanır.

$$\sum_j \xi_j = \sum_{j \in N} \xi_j \tag{2.2}$$

Eğer tanımlanan çizelgeleme probleminde işlere ait ağırlıklar (w_j) söz konusu ve $\forall j \in N,\ i \neq j$ için $w_j \neq w_i$ olan $\exists i \in N$ ise, toplam formdaki amaç fonksiyonları $\sum_j w_j \xi_j$ şeklinde ifade edilir ve denklem 2.3 ile hesaplanır.

$$\sum_j w_j \xi_j = \sum_{j \in N} w_j \xi_j \tag{2.3}$$

Bir çizelgeleme problemine ait çözümlerin kalite düzeyini belirleyen amaç fonksiyonlarının en temel formları denklem 2.1-2.3 ile verilmiştir. Buna göre, tanımlanan bir çizelgeleme probleminin amaç fonksiyonu $\gamma \in \{\zeta_{max}, \sum_j \xi_j, \sum_j w_j \xi_j\}$ ile ifade edilebilir.

2.2. Çizelgeleme Problemlerinin Gösterimi

Çizelgeleme problemlerini tanımlamak ve ifade etmek için literatürde kabul görmüş sistematik, Graham vd. [29]' nin ortaya koyduğu $\alpha|\beta|\gamma$ üçlü gösterimidir. Bu temsil sistematiğine göre α ile atölye yapısı (M), β ile iş özellikleri (N) ve eğer varsa kaynak tanımları (R), son olarak γ ile amaç fonksiyonu ifade edilir.

Üçlü gösterimdeki $\alpha = \alpha_1\alpha_2$ formundadır ve $\alpha_1 \in$ {"", P, Q, R, F, O, J} simgesi en temel atölye yapılarını içermektedir. Buna göre, tanımlanan Π problemindeki atölye yapısını belirtmek üzere, tek makine çizelgeleme için boşluk (""), özdeş paralel makine çizelgeleme için 'P', uniform paralel makine çizelgeleme için 'Q', ilişkisiz paralel makine çizelgeleme için 'R', akış tipi çizelgeleme için 'F', açık atölye tipi çizelgeleme için 'O' ve iş atölyesi çizelgeleme için 'J' kullanılmaktadır. Atölye yapısının diğer bileşeni $\alpha_2 \in$ {1, |m|, m} makine sayısını belirtmektedir. Tek makine çizelgeleme için '1', çok makineli diğer atölye ortamları için makine sayısı '|m|' doğrudan yazılmaktadır. Eğer tek makine dışındaki atölye ortamlarında makine sayısı bir problem girdisi olarak verilmemiş ise, 'm' simgesi kullanılmaktadır.

Üçlü gösterimdeki $\beta \in \{$prec, pmtn, res, s_{ij}, s_j, r_j, L, F, A, D$\}$ simgesi en temel iş özelliklerini ve yapısal durumları içermektedir. Buna göre, tanımlanan Π problemindeki iş özelliklerini ve yapısal kısıtları belirtmek üzere, $\prec \neq \emptyset$ durumu için 'prec' ifadesi ve iş kesme üstünlüğü için 'pmtn' ifadesi kullanılmaktadır. Kaynak tahsisi içeren durumlar için 'res' ifadesi veya kaynak tahsisini açıklayan matematiksel ifade kullanılmaktadır. Ayar süresinin bir önceki tamamlanan işe göre değişkenlik gösterdiği durumlar için 's_{ij}' ve ayar süresinin değişkenlik göstermediği durumlar için 's_j' ifadeleri kullanılmaktadır. Farklı geliş zamanlarının olduğu durumlar için 'r_j' ifadesi kullanılmaktadır. Değişken işlem süreleri için bu değişkenliğin şiddetini gösteren matematiksel formül veya öğrenme için 'L', unutma için 'F' ve yaşlanma için 'A' veya 'D' ifadeleri kullanılmaktadır. Üçlü gösterimdeki $\gamma \in \{\zeta_{max},\ \sum_j \xi_j,\ \sum_j w_j\xi_j\}$ simgesi en temel amaç fonksiyonlarını içermektedir. Amaç fonksiyonlarının matematiksel ifadeleri Bölüm 2.1.4' de verilmiştir.

2.3. Çizelgeleme Problemlerinde Çözüm Kavramı

Çizelgeleme problemlerinin çözümü, işlerin hangi makinede ne zaman başlayıp ne zaman biteceğini gösteren bir yapıyla ifade edilebilir. Çözümü temsil eden bu yapıya çizelge denir ve bu tezde aksi belirtilmedikçe çizelge için π simgesi kullanılmıştır. Eğer tanımlanan çizelgeleme problemine ait herhangi bir çizelge biliniyorsa, problemdeki işlerin $(1 \leq j \leq n)$ bazı temel özellikleri kolayca tespit edilebilir. Çizelgeye göre değer alan bu özellikler işlerin başlama ve tamamlanma zamanlarıdır. Bu tezde, başlama zamanı ayrı bir parametre olarak verilmemiştir ve tüm hesaplar tamamlanma zamanlarına göre yapılmıştır. Bu nedenle, bir makineye atanmış işlerin $1 \leq j \leq n$ oluşturduğu π çizelgesinde $C_0^\pi = t_0$ olmak üzere, herhangi bir j işinin tamamlanma zamanı $C_j^\pi = C_i^\pi + t + p_j$ ile hesaplanmaktadır. Burada, i indisi ile j işinden hemen önce çizelgelenmiş olan iş, t_0 ile çizelgenin başladığı zaman, $t \geq 0$ ile i işinin tamamlanmasıyla j işinin başlaması arasında geçen zaman (ayar süresi s_j veya s_{ij}, geliş zamanı r_j, mola, bakım onarım zamanı, makine boşta bekleme zamanı vs.) ifade edilmiştir. Jouglet ve Carlier [30], kombinatoryal optimizasyon probleminin çözümü ile amaç fonksiyonu arasındaki ilişkiyi matematiksel olarak göstermişlerdir. Bu ifade, çizelgeleme problemleri için aşağıdaki gibi uyarlanabilir.

Π	Tanımlanan çizelgeleme problemi.
x_j	Problemin çözümünü oluşturan değişkenler, $\forall j \in N$.
$\Upsilon(x_j)$	Çözümdeki herhangi bir x_j değişkeninin alabileceği tüm değerlerin oluşturduğu küme, $\forall j \in N$.
π	Problemin çözümü, $\pi = \{x_1, x_2, \cdots, x_n\}$.
S	Probleme ait çözüm uzayı, $S = \Upsilon(x_1) \times \Upsilon(x_2) \times \cdots \times \Upsilon(x_n)$.
$\mathcal{C}^\Pi$	Probleme ait kısıtlar.
א	Kısıtların sağlandığı çözümleri ifade eden bir fonksiyon, $א(\mathcal{C}^\Pi): S \rightarrow D$.
D	Probleme ait olurlu çözüm uzayı, $D \subseteq S$.
ב	İşlerin tamamlanma zamanını hesaplayan bir fonksiyon, $\forall j \in N$ için $ב(j): \pi \rightarrow C_j^\pi$.
γ	Amaç fonksiyonu, $\gamma(\pi): C_1^\pi, C_2^\pi, \cdots, C_n^\pi \rightarrow \mathbb{R}$.

Yukarıda verilenlere göre, bir çizelgeleme probleminin çözümü $\Upsilon(x_1) \times \Upsilon(x_2) \times \cdots \times \Upsilon(x_n)$ kartezyen çarpımından elde edilen bir vektördür. Elde edilen vektör (çizelge) için $\pi \in D \subseteq S$ ilişkisi söz konusu ise, bulunan π çizelgesi $\mathcal{C}^{\Pi}$ ile ifade edilen kısıtları ihlal etmeyen olurlu bir çözümdür. Bu çizelgenin $\beth$ fonksiyonu ile çözümlenmesiyle her $j \in N$ için C_j^{π} değerleri ve dolayısıyla, amaç fonksiyon değeri $\gamma(\pi)$ elde edilir.

Tanımlanan çizelgeleme probleminde $\pi^* \in B \subset D$ ve $\forall \pi \in B \subset D$ için eğer $\gamma(\pi^*) \leq \gamma(\pi)$ ise, π^* bir lokal optimum çözümdür (yaklaşık çözüm). Eğer $\forall \pi \in D$ için $\gamma(\pi^*) \leq \gamma(\pi)$ ise, π^* bir global optimum çözümdür (kesin çözüm). Tüm bunların yanında, bir probleme ait kısmi çözümler de oluşturulabilir. Kısmi çözümler, π çizelgesindeki x_j değişkenlerinden bazılarının oluşturduğu ve boyutu n'den küçük olan çözümlerdir. Bu tezde, kısmi çözümler alt çizelge olarak adlandırılmıştır.

Kesin çözümde bulunan özelliklerin bir kısmını veya tamamını barındıran olurlu bir çözüme, baskın çözüm denilmektedir. Tanımlanan çizelgeleme probleminde π^* bir kesin çözüm ve $B^* \subseteq D$ kesin çözümlerin oluşturduğu bir olurlu çözüm kümesi olsun. Bu durumda $\pi^* \in B^*$ ve $\forall \pi \in B^*$ için $\gamma(\pi^*) = \gamma(\pi)$ olur. Eğer tanımlanmış bir baskınlık özelliğini barındıran çözümlerin kümesi $\ddot{B} \subseteq S$ ise, $B^* \cap \ddot{B} \neq \emptyset$ olur.

2.4. Çözüm Yöntemi ve Algoritma Kavramı

Tanımlanan bir problemin herhangi bir örneğine ait çözümü oluşturan sistematik uygulamalara algoritma denilmektedir. Algoritmalar, sınırlı sayıdaki adımlarla uygun bir girdiyi deterministik bir şekilde beklenen çıktıya dönüştüren açıkça tanımlanmış genel prosedürlerdir. Bu anlamda algoritmalar; girdi, operasyonel adımlar ve çıktıdan meydana gelir. Algoritmanın girdisi, üzerinde çalışılan problem örneğidir ($\mathcal{I} \in \Pi$). Operasyonel adımların her biri ise, ilgili problem örneğinin çözümünü elde etmek için gerçekleştirilen ardışık sınırlı eylemlerdir. Bu eylemler genellikle bilinen basit aritmetik, mantıksal, atama ve kontrol işlemleridir. Bu tezde, algoritmalar bir adım

listesi şeklinde verilmiştir ve her adımda gerçekleştirilen işlemler matematiksel olarak ifade edilmiştir.

Algoritmaların girdisi, bir problem örneğine ait gerçek parametre değerlerine net ve düzenli bir kodlama sistematiği uygulanarak oluşturulmuş standart yapılardır. Bu bakımdan, bir algoritma için uygun olan herhangi bir girdi, önceden tanımlanmış sonlu bir sembol kümesine ait elemanlarla sistematik şekilde oluşturulmuş sıralı yapılardır [27]. Yaygın olarak kullanılan kodlama yöntemi, ikili sayı sistemidir (binary encoding scheme). Bu sistemde, tüm sayılar iki tabanına göre 0 ve 1' lerden oluşan çeşitli uzunluktaki dizilerden meydana gelir. Ayrıca, sayıların ardışık 1' lerle ifade edildiği birli kodlama sistemi de bulunmaktadır (unary encoding scheme). Önceden tanımlanmış bir sisteme göre kodlanmış bir girdinin karakter uzunluğu, girdi boyutunu temsil eder. Garey ve Johnson' ın [31] ifade ettiği şekliyle Length $(\mathcal{I}) \in \mathbb{Z}^+$ fonksiyonu, $\mathcal{I} \in \Pi$' yi temsil etmek için gerekli olan sembollerin sayısını, yani girdi boyutunu temsil etmektedir.

2.5. Algoritmanın Zaman ve Hafıza Karmaşıklığı

Bir problem farklı algoritmalar ile çözülebilir. Bu durumda farklı algoritmaların etkinliği, girdi boyutuna bağlı olarak, problemi çözerken uygulanan temel operasyonların miktarındaki değişimle kıyaslanabilir. Uygulanan operasyonların miktarı, algoritmanın koşum süresi ve kullandığı hafıza alanıyla ölçülür. Dolayısıyla, operasyon sayısı algoritma girdisinin bir fonksiyonudur. Yani, girdi boyutu büyüdükçe algoritma çözüm üretmek için daha fazla operasyon yapmak durumunda kalır. Buradan hareketle, p ve q problem boyutunun bir fonksiyonu olmak üzere, boyutu Length $(\mathcal{I})$ olan bir $\mathcal{I} \in \Pi$ için algoritmanın zaman karmaşıklığı $O: p\big(\text{ Length }(\mathcal{I})\big) \rightarrow$ CPU zamanı ve hafıza karmaşıklığı $O: q\big(\text{ Length }(\mathcal{I})\big) \rightarrow$ RAM miktarı fonksiyonlarıyla tanımlanabilir.

Bu tezde algoritmalar, zaman karmaşıklığı veya koşum süresi ile değerlendirilmiştir. Algoritma karmaşıklığını ifade eden O fonksiyonunun çıktısı, algoritmanın oluşturulduğu ve çalıştırıldığı ortamlara göre değişiklik gösterebilir. Mesela,

Length $(\mathcal{I}) = n$ olan bir $\mathcal{I} \in \Pi$ için zaman karmaşıklığı $O(p(n))$ olan bir algoritmanın koşum süresi, işlemcisi nispeten daha hızlı olan bir bilgisayarda daha küçük olur. Bir başka açıdan bakıldığında; Length $(\mathcal{I}) = n$ olan bir $\mathcal{I} \in \Pi$ için geliştirilen A ve B algoritmalarının operasyon sayılarını ifade eden fonksiyonlar $p\_A(n) \leq p\_B(n)$ ise, her iki algoritmanın aynı bilgisayardaki koşum süreleri için $O(p\_A(n)) \leq O(p\_B(n))$ olur. Algoritmalar zaman karmaşıklığına göre, polinom zamanlı, üstel zamanlı ve sözde-polinom zamanlı olmak üzere üçe ayrılmıştır [27].

2.5.1. Polinom Zamanlı Algoritma

Polinom zamanlı algoritmalarda temel işlemlerin ulaşabileceği en büyük değer, girdi boyutunun polinomuyla ifade edilir. Bu, $p(n)$ fonksiyonunun bir polinom olduğu anlamına gelir. Böylece, boyutu Length $(\mathcal{I}) = n$ olan bir $\mathcal{I} \in \Pi$ için geliştirilen algoritmanın zaman karmaşıklığı $O(p(n))$ ise, $p(n)$ fonksiyonu bir polinomdur.

2.5.2. Üstel Zamanlı Algoritma

Üstel zamanlı algoritmalar, zaman karmaşıklığı polinomiyal olmayan algoritmalardır. Tam olarak, boyutu Length $(\mathcal{I}) = n$ olan bir $\mathcal{I} \in \Pi$ için geliştirilen algoritmanın zaman karmaşıklığı $O(p(n))$ ise, $p(n)$ fonksiyonu polinom değildir. Üstel zamanlı algoritmaların zaman karmaşıklığı ya üstel fonksiyonla, ya da üstel hızda artan bir fonksiyonla ifade edilir. Bu nedenle, polinom zamanlı algoritmalara nazaran oldukça verimsiz çözüm yöntemleridir.

2.5.3. Sözde-polinom Zamanlı Algoritma

Sözde-polinom zamanlı algoritmalarda temel işlemlerin ulaşabileceği en büyük değer, girdi boyutunun ve girdideki en büyük değerin polinomuyla ifade edilir. Garey ve Johnson' ın [31] ifade ettiği şekliyle Max $(\mathcal{I}) \in \mathbb{Z}^+$ fonksiyonu, $\mathcal{I} \in \Pi$' nin

girdileri arasındaki maksimum değerin büyüklüğünü temsil etmektedir. Buradan hareketle, p fonksiyonunun her iki parametreyi de içerdiği düşünülürse $p\big(\text{Length}\,(\mathcal{I}), \text{Max}\,(\mathcal{I})\big)$ olur ve Max $(\mathcal{I})$ fonksiyonu birli kodlama sistemine göre düzenlenirse, p girdi boyutuna göre polinomiyal fonksiyon haline gelir [4]. Böylece, girdi boyutu Length $(\mathcal{I}) = n$ olan ve girdiler arasındaki maksimum değerin büyüklüğü Max $(\mathcal{I}) = m$ olan bir $\mathcal{I} \in \Pi$ için geliştirilen sözde-polinom zamanlı bir algoritmanın zaman karmaşıklığı $O\big(p(n, m)\big)$ olmaktadır.

2.6. Çizelgeleme Problemlerinin Karmaşıklığı

Bir problemin kesin çözümünü elde etmek için öncelikle uygun bir çözüm yöntemi belirlemek gerekmektedir. Bölüm 2.4' de bahsedildiği üzere, algoritmalar çeşitli problemlerin çözümünü elde etmek için tasarlanmış sistematik işlemler topluluğudur ve ilgili problemin karmaşıklığı çözüm yönteminin de karmaşıklığını belirlemektedir. Karmaşıklık teorisi temel olarak karar problemleri üzerine kurulmuştur. Bilindiği üzere bir optimizasyon problemi, tanımlanmış amaç fonksiyonuna bir hedef değer atamasıyla karar problemi olarak ifade edilebilmektedir. Böylece, karar problemi için tespit edilen karmaşıklık, optimizasyon problemi için de geçerli olmaktadır.

Teorik olarak, deterministik Turing makinesi (Deterministic Turing Machine) ile polinom zamanda kesin çözümüne ulaşılabilen tüm karar problemleri P ile gösterilen problem kümesini oluşturur. NP ile gösterilen problem kümesi ise, deterministik olmayan Turing makinesi (Non-deterministic Turing Machine) ile çözümü türetilen ve polinom zamanda kontrol edilebilen karar problemlerinden oluşur. Turing makinelerini, P ve NP kümelerini daha detaylı incelemek isteyen okuyucular için [31–33] ile belirtilen kaynaklar faydalı olabilir.

Burada biraz daha üzerinde durulması gereken konu, NP kümesini oluşturan karar problemleridir. Leung [4] tarafından belirtilen şekliyle, NP kümesine ait bir karar problemi 'özlü' bir sertifikaya sahiptir. Bu sertifikayla kastedilen, tanımlanan karar problemine ait cevabın (evet veya hayır) test edilmesi için gereken maksimum sürenin, problem boyutunun polinomuyla sınırlı olmasıdır. Başka bir ifadeyle, NP

kümesinde bulunan bir karar problemi polinom zamanda çözülemezken, mevcut bir çözümün cevabı polinom zamanda kontrol edilebilmektedir. Bu iki küme arasında P = NP ilişkisini destekleyen bir kanıt olmazken, P ≠ NP ilişkisi de henüz aksi ispatlanmamış olan bir durumdur.

Karar problemleri arasında tanımlı bir dönüşüm söz konusudur. Tanımlanan dönüşüm fonksiyonu vasıtasıyla, bir karar problemine ait girdilerle bir başka karar problemine ait girdiler arasında sistematik bir ilişki oluşturulur. Bahsedilen dönüşüm; Π_1 ve Π_2 iki farklı karar problemi olmak üzere, $\forall \mathcal{I}_1 \in \Pi_1$ için $\vartheta: \mathcal{I}_1 \rightarrow \mathcal{I}_2 \in \Pi_2$ fonksiyonu ile gerçekleşir ve $\Pi_1 \propto \Pi_2$ şeklinde gösterilir. Dönüşüm fonksiyonu ϑ, aşağıdaki temel iki özelliğe sahip olmalıdır.

1. Dönüşüm fonksiyonu ϑ polinom zamanda hesaplanabilmelidir.
2. Tüm $\mathcal{I}_1 \in \Pi_1$ için; ancak ve ancak $\vartheta(\mathcal{I}_1)$' nin cevabı 'evet' ise, $\mathcal{I}_1$' nin cevabı 'evet' olur.

Karar problemlerinin asgari zorluk durumunu ifade etmek için karmaşıklık uzayında NP-tam kümesi tanımlanmıştır. Buna göre, eğer $\Pi_1 \in$ NP ve $\forall \Pi \in$ NP için $\Pi \propto \Pi_1$ ise, $\Pi_1 \in$ NP-tam olur. Bir karar probleminin ($\Pi_1 \in$ NP) NP-tam olduğunu kanıtlamak için NP kümesindeki tüm karar problemlerinden ($\forall \Pi \in$ NP) ilgili karar problemine polinomiyal dönüşüm olduğunu ($\Pi \propto \Pi_1$) göstermek gerekmektedir. NP kümesi sınırsız sayıda problem içerdiğinden dolayı, bir problemin karmaşıklığını bu yöntemle belirlemek oldukça zor görünmektedir. Teorik olarak bu zorluğu aşmak [31]' de belirtilen yardımcı önerme (Lemma 2.3) ile mümkündür. Yardımcı önermede (Lemma 2.3) anlatılana göre; eğer $\Pi_1, \Pi_2 \in$ NP, $\Pi_1 \in$ NP-tam ve $\Pi_1 \propto \Pi_2$ ise, $\Pi_2 \in$ NP-tam olur. Ancak bu durumda da, NP-tam kümesine ait olduğu kesinleşmiş bir problem bulunması gerekmektedir. Cook [34], deterministik olmayan Turing makinesi programı ile 'Gerçeklenebilirlik' (Satisfiability) problemi arasında bir dönüşüm oluşturarak diğer tüm dönüşümlere referans olacak bir NP-tam problem tanımlamıştır. Böylece, Gerçeklenebilirlik probleminden herhangi bir karar problemine dönüşüm yaparak ilgili problemin NP-tam olduğunu göstermek mümkün olmuştur. Buradan hareketle, Karp [35] birçok kombinatoryal problemin NP-tam olduğunu göstermiştir.

Anlatıldığı üzere bir karar problemi NP-tam ise, ikili kodlama sistemine göre polinomiyal bir yöntemle kesin çözümünü elde etmek mümkün değildir. Ancak NP-tam olmasına rağmen, birli kodlama sistemine göre polinomiyal yöntemle kesin çözümü elde edilebilen problemler de bulunmaktadır. Bu türdeki kesin çözüm yöntemleri, ikili kodlama sistemine göre sözde-polinom zamanlı olarak ifade edilmektedir. Buradan hareketle, ikili kodlama sistemine göre NP-tam olan ancak birli kodlama sistemine göre olmayan karar problemleri 'normal şiddette' NP-tam (ordinary NP-complete veya NP-complete in the ordinary sense) sınıfında yer almaktadır. Hem ikili hem de birli kodlama sistemlerine göre NP-tam olan karar problemleri de 'güçlü şiddette' NP-tam (strong NP-complete veya NP-complete in the strong sense) sınıfında yer alır.

Bir karar probleminin normal veya güçlü şiddette olduğunu tespit etmek için bazı temel kavramların bilinmesi gerekmektedir. Daha önce bahsedildiği üzere, bir problem örneğini $\mathcal{I} \in \Pi$ tanımlamak için Length $(\mathcal{I})$ ve Max $(\mathcal{I})$ fonksiyonları kullanılmaktadır. Eğer $\mathcal{I}$' de bulunan sayılar ikili kodlama sistemine göre düzenlendiğinde, Max $(\mathcal{I})$' nin alabileceği en büyük değer Length $(\mathcal{I})$' nin bir polinomu ile sınırlandırılmamış ise, Π bir sayı problemidir. Tanımlanan p bir polinomiyal fonksiyon olmak üzere, Π_p problem yapısı, $\forall \mathcal{I} \in \Pi_p$ için Max $(\mathcal{I}) \leq p\big(\text{Length } (\mathcal{I})\big)$ şartını sağlayan Π problem yapısının bir özel durumudur. Böylece, Π bir sayı problemi olsa bile, Π_p sayı problemi değildir ve eğer herhangi bir p polinomiyal fonksiyonu için $\Pi_p \in$ NP-tam ise, $\Pi \in$ NP-tam (güçlü şiddette) olur.

Görüldüğü üzere, bir karar problemi sayı problemi değil ve NP-tam ise, bu problem güçlü şiddette NP-tam olmaktadır. Eğer karar problemi sayı problemi ve NP-tam ise, güçlü şiddette NP-tam olup olmadığı belirsizdir. Bu belirsizlik durumunda, sözde-polinom zamanlı dönüşüm kavramı ön plana çıkmaktadır. Buna göre, yukarıda 1 ve 2 numaralı özelliklerle tarif edilen ϑ dönüşümünde $\Pi_1 \in$ NP-tam (güçlü şiddette) ve Max $(\mathcal{I}_2) \leq p\big(\text{Length } (\mathcal{I}_1), \text{Max } (\mathcal{I}_1)\big)$ ise, $\vartheta: \mathcal{I}_1 \rightarrow \mathcal{I}_2$ dönüşüm fonksiyonu sözde-polinom zamanda gerçekleşir ve $\Pi_2 \in$ NP-tam (güçlü şiddette) olur ([31]' deki yardımcı önerme - Lemma 4.1). Dolayısıyla faydalı bir sonuç olarak, eğer bir karar problemi güçlü şiddette NP-tam sınıfında yer alıyor ise, ilgili problemin kesin

çözümü sözde-polinom zamanlı bir yöntemle elde edilemez ([31]' de gözlemlenen bulgu - Observation 4.2). Bu durumun bir başka doğal sonucu olarak, NP-tam sınıfında yer alan bir karar probleminin kesin çözümü sözde-polinom zamanlı bir yöntemle elde edilebiliniyor ise, ilgili problem normal şiddette NP-tam sınıfındadır.

2.7. Kesin Çözüm Yöntemleri

Kesin çözüm, bir probleme ait olurlu çözüm uzayındaki en iyi çözüm, yani optimum çözümdür. P sınıfında yer alan problemlerin kesin çözümü polinom zamanlı yöntemlerle elde edilebilirken, NP sınıfında yer alan problemlerin kesin çözümü daha kompleks çözüm yöntemleriyle bulunabilmektedir. Çizelgeleme problemlerinin büyük kısmı NP sınıfında yer aldığından dolayı, kesin çözümlerini elde etmek için genellikle tam sayım yöntemi, DS ve DP kullanılmaktadır.

2.7.1. Tam Sayım Yöntemi

Tam sayım yöntemi, bir problemin olurlu çözüm uzayındaki tüm çözümleri değerlendirerek optimum sonucu bulmaya çalışır. Problem boyutunun yeterince küçük olduğu ve etkin bir çözüm yönteminin uygulanamadığı durumlarda tam sayım yöntemi uygulanabilir. Ancak, optimizasyon problemlerinin olurlu çözüm uzayı problem boyutuna göre genellikle üstel arttığı için tam sayım yöntemlerinin uygulanması genellikle mümkün olmamaktadır. Bu nedenle, etkin bir kesin çözüm yönteminin geliştirilemediği durumlarda genellikle yaklaşık çözüm yöntemleri uygulanmaktadır.

2.7.2. Dal-Sınır Algoritması

DS, olurlu çözümlerin daha küçük parçalarını dikkate alarak, o küçük parçayı barındıran sayısız çözümle ilgili değerlendirme yapan bir kesin çözüm yöntemidir. Optimum sonuca ulaşma ihtimali olmayan çözümleri uygun tekniklerle önceden

elimine ederek, tüm olurlu çözümleri dolaylı olarak değerlendirmiş olur. Algoritmanın temel yapısı dallanma ve sınırlama olmak üzere iki prosedürden oluşur. Dallanma aşamasında, tanımlanan problemin kısmi çözümleri oluşturulur. Kısmi çözümler belirli bir sistematikle elde edilir. Genellikle, çözümü temsil eden yapının anlamlı en küçük parçasına göre dallanma yapılır. Bu yüzden, ardışık olarak dallanan kısmi çözümler bir ağaç yapısı teşkil eder ve herhangi bir seviyedeki kısmi çözüm, bağlı olduğu önceki kısmi çözümlerin devamıdır. Bu şekilde ağaç yapısı oluşturarak devam eden kısmi çözümler, her bir dallanma işlemi sonrasında daha da genişleyerek olurlu bir çözüm oluştururlar ve elde edilen bu olurlu çözümler yaprak olarak adlandırılır.

Sınırlama işlemiyle kısmi çözümlerin oluşturduğu ağaç yapısının bazı bölümleri budanır ve budanan bölümler için çözüm süreci sonlanmış olur. Bir kısmi çözümü sonlandırma kararı için iki değer arasında kıyaslama yapılır. Bunlardan biri kısmi çözümlerin her biri için hesaplanan alt sınır değerleri, bir diğeri de üst sınır olarak bilinen en iyi amaç fonksiyon değeridir. Böylece, bir kısmi çözümün alt sınır değeri üst sınır değerinden daha kötü ise, bu kısmi çözümün oluşturduğu dallanma yapısı budanır ve değerlendirme dışı bırakılır. Budanmamış ve yaprak oluşturmuş tüm olurlu çözümlerin en iyisi optimum çözümdür.

2.7.3. Dinamik Programlama Algoritması

DP, birbiriyle bağlantılı ardışık kararlar neticesinde optimum sonuca ulaşmayı hedefleyen yinelemeli bir yöntemdir. Algoritmanın karar aşamaları problem evreleri olarak ifade edilir ve nihai çözüm, koşulları belirlenmiş bir başlangıç evresinin üzerine kurulur. Evreleri ifade eden yapı matematiksel olarak tanımlanır ve yinelemeli bir fonksiyon ile her evredeki kısmi çözümün değeri tespit edilir. DP' nda, herhangi bir evredeki kısmi çözüm kendisinden önceki çeşitli evrelerde oluşturulan kısmi çözümlerle ilişkilidir ve bu ilişki zinciri başlangıç evresine kadar mutlaka devam eder. Bu nedenle, nihai çözümün oluştuğu evre ile başlangıç evresi arasında optimum çözümü ifade eden bir yol bulunmaktadır [36].

2.8. Yaklaşık Çözüm Yöntemleri

Çizelgelemede karşılaşılan problemlerin birçoğu için etkin kesin çözüm yöntemleri uygulanamamaktadır. Sadece NP sınıfına ait problemleri dikkate aldığımızda, yukarıda anlatılan kesin çözüm yöntemleri sınırlı boyuttaki problem örneklerine ancak uygulanabilmektedir. Bu durumlarda yaklaşık çözüm yöntemleri devreye girmektedir. Yaklaşık çözüm yöntemleri optimum sonuca ulaşmayı garanti etmeseler de, karar vericilerin kabul edebileceği düzeyde sapmayla optimuma yakın sonuçlar verebilmektedirler. Bazı yöntemlerin en büyük sapma oranı tespit edilebilmektedir ve genellikle bu yöntemlerin zaman karmaşıklığı da hesaplanabilmektedir. Yaklaşık çözüm yöntemlerinin bazıları ise daha kompleks yapıdadır. Bu yöntemlerin optimum sonuçtan sapması ve beraberinde zaman karmaşıklığı genellikle önceden tespit edilememektedir. Sezgisel yöntemler, bu yapıda olan yaklaşık çözüm algoritmalarıdır ve NP sınıfındaki problemlerin büyük boyutlu olanları için sıklıkla tercih edilirler.

3. DEĞİŞKEN İŞLEM SÜRESİ VE AYAR SÜRESİ

Bu tez çalışmasında üzerinde durulan temel konulardan biri, makine çizelgeleme problemlerinde yaygın olarak kullanılan değişken işlem sürelerinin çeşitleri, özellikleri ve uygulama alanlarıdır. Değişken işlem süresi, genel anlamda denklem 3.1' de verilen eşitlikle tanımlanabilir. Buradaki p_j^A gerçekleşen işlem süresini, $\mathcal{P}$ problemin bir özelliği olarak tanımlanan değişim fonksiyonunu, p_j problemin bir girdisi olarak verilen normal işlem süresini ve $k = 1,2,\cdots,m$ için geçerli olan x_k da değişim fonksiyonunun normal işlem süresi dışındaki diğer tüm bağımsız değişkenlerini ifade etmektedir. İşlem süresindeki değişimin yönü ve şiddeti tanımlanan $\mathcal{P}$ fonksiyonuna bağlıdır ve dolayısıyla, ilgili problemi tanımlayan baskın özelliklerden biridir.

$$p_j^A = \mathcal{P}\left(p_j, x_1, x_2, \cdots, x_m\right) > 0 \tag{3.1}$$

Literatürdeki değişken işlem süresi içeren çizelgeleme problemleri, öğrenme (learning), unutma (forgetting) ve yaşlanma (aging, deteriorating) etkilerinin ayrı veya bir kaçının beraber kullanımıyla ifade edilmiştir. Öğrenme etkisi genel olarak, bir işe ait gerçekleşen işlem süresinin, tamamlanmış işlere ait bir takım parametrelere bağlı olarak azalmasıdır. Bu tanımdaki parametrenin x_k olduğunu varsaydığımızda, $\mathcal{P}$ fonksiyonunun x_k' ya göre kısmi türevi $\partial\mathcal{P}\left(p_j, x_1, x_2, \cdots, x_m\right)/\partial x_k \leq 0$ olur. Unutma ve yaşlanma etkileri ise, bir işin gerçekleşen işlem süresinin çeşitli nedenlerle artmasıdır. Bu tanımdaki çeşitli nedenlerin x_k olduğunu varsaydığımızda ise, $\mathcal{P}$ fonksiyonunun x_k' ya göre kısmi türevi $\partial\mathcal{P}\left(p_j, x_1, x_2, \cdots, x_m\right)/\partial x_k \geq 0$ olur. Unutma etkisindeki artış, tecrübeye (öğrenme) bağlı kazanılan verimliliğin azalmasından kaynaklanırken, yaşlanma etkisindeki artış, tamamen yapılan işin veya işlem sürecinin niteliğinden kaynaklanmaktadır. Örneğin, çelik haddeleme işletmelerinde, fırın ile hadde arasında beklemek zorunda kalan çelik kütüklerin sıcaklığı haddeleme için gereken sıcaklığın altına düşer ve tekrar ısıtılmaya ihtiyaç duyulur. Bu durum tekrar ısıtılan çelik kütüğün işlem süresini artırdığından dolayı, yaşlanma etkisi (deteriorating effect) olarak ifade edilebilir [37]. Benzer şekilde, bir

atölyedeki bir takım işlerin, havanın kararmasından veya mevsimsel şartlardan etkilenerek daha yavaş yapılması işlem sürelerini artıracağı için, yaşlanma etkisi (aging effect) olarak ifade edilebilir. Görüldüğü üzere, yaşlanma etkisi bir sistemde tek başına gözlemlenebilir ve gerçekleşen işlem süresi, yaşlanma etkisine bağlı olarak normal işlem süresinin üstüne çıkabilir. Ancak, unutma etkisi beraberinde öğrenme etkisinin de olduğu durumlarda geçerlidir ve bu nedenle öğrenmeyle birlikte kullanımı yaygındır. Daha açık ifadeyle, unutma etkisinin oluşması için beraberinde öğrenme etkisinden kaynaklanan bir takım tecrübelerin edinilmesi gerekmektedir [38].

Öğrenme, unutma ve yaşlanma etkilerinin her biri için farklı tanımlar ifade edilmiş olsa da, kullanılan modeller ve uygulama şekli bakımından ortak yönleri oldukça fazladır. İlgili literatürde, değişken işlem süreleri üzerine yapılan analizlerin kökeni Gupta ve Gupta [39] ve Browne ve Yechiali [40] tarafından yapılan iki bağımsız çalışmaya dayanmaktadır. Bunların devamı niteliğinde, deneysel, pratik veya teorik çalışmalar ile formülüze edilen değişken işlem süreli modellerin analizini içeren geniş çaplı çalışmalar yapılmıştır [15, 27, 37, 41, 42]. Bu çalışmaların ortak özelliği, öğrenme etkili problemleri ve ilgili öğrenme modellerini de içermeleridir. Dolayısıyla, öğrenme etkisi kapsamında, unutma ve yaşlanma etkilerinin de incelenmesi mümkündür. Bu bölümde, değişken işlem süreli çizelgeleme ile ilgili raporlanan literatür araştırması, öğrenme ve unutma etkileri kapsamında yapılmıştır. Beraberinde, ayar süreleri içeren çizelgeleme modelleri üzerinde de durulmuştur.

3.1. Öğrenme Etkisi

Yaparak öğrenmek insan verimliliğini artıran doğal bir süreçtir. Bu süreçte, bir işi tamamlamak için gereken süre, aynı işin tekrarlanmasına bağlı olarak azalmaktadır. Öğrenme, zamana bağlı ve haricen kontrol edilebilir bir olgudur. Çeşitli araştırmalar insan performansının takviyelerle veya sık görülen tekrarlarla arttığını doğrulamıştır [20, 43]. Öğrenme etkisinin tarihsel durumunu gösteren ve ulaşılacak amaç doğrultusunda yapılan planların performans parametreleriyle öğrenme fonksiyonu arasındaki ilişkileri anlatan çeşitli çalışmalar yapılmıştır [44]. Öğrenme etkisine bağlı

olarak işlem sürelerindeki düşüş, işverenler tarafından doğrudan mali tasarruf olarak algılanırken, çalışanlara da moral artışı olarak yansımaktadır. Anzanello ve Fogliatto' ya [17] göre öğrenme fonksiyonu, çalışanların yeni bir görev karşısındaki performansını gösteren etkili bir araç olarak ifade edilmiştir.

Wright [45], birçok araştırmacı tarafından üretimde öğrenme etkisinin var olduğunu gözlemleyen ve analiz eden ilk kişi olarak bilinmektedir. Yazarın bu çalışması, öğrenme etkisinin birçok farklı alanda uygulanmasına ve yeni öğrenme modellerinin ortaya çıkmasına yol açmıştır. Örneğin, bazı çalışmalarda belirli üretim süreçlerine göre uyarlanmış öğrenme etkisinin birçok özel formu kullanılmış [46], bazılarında üretim ve operasyon kararları optimize edilmiş [47], bazılarında da veriye dayalı analizler yapılarak öğrenmenin çeşitli imalat firmalarında gerçekleştiği gösterilmiştir [48].

Öğrenme etkisinin üretim ve operasyon yönetimi faaliyetlerine uygulanmasıyla elde edilen çok çeşitli avantajlar bulunmaktadır. Bu avantajların yöneticiler tarafından kapsamlı bir şekilde anlaşılması, etkili yönetimsel kararlara ve olumlu sonuçlara fırsat verebilir. İşlem standartlarını daha net belirlemek ve neticesinde gerçekçi üretim çıktılarını gözlemlemek [49], detaylı planlama ve operasyon seviyelerindeki avantajlardan sayılabilir. Yine planlama çerçevesinde, bir proses için uygun çalışma zamanlarını belirlemek [50], üretimin çıktısına yönelik tahminlerde bulunmak [51] ve uygun olmayan standartları güncellemek [52] gibi öğrenme etkisinden kaynaklanan avantajlardan faydalanılmıştır. Öğrenme etkisinin farklı alanlardaki uygulamalarını ve bu uygulamalardan elde edilen olumlu sonuçları incelemek için yakın zamanda yapılan detaylı literatür araştırmasına bakılabilir [53].

3.1.1. Öğrenme Etkisi Modelleri

Öğrenme etkisi, aynı işi tekrarlayan çalışanların performansını ölçmeye yarayan bir fonksiyonla ifade edilmektedir [17]. Öğrenme etkisi genel olarak, logaritmik-doğrusal (Çizelge 3.1), üstel (Çizelge 3.2) ve hiperbolik (Çizelge 3.3) fonksiyonlarla modellenmiştir. Bu kapsamda verilen öğrenme fonksiyonları, yaygın olarak

kullanılan ve çoğu çalışmacı tarafından bilinen modellerdir. Farklı öğrenme modelleri ve çeşitli kullanım alanlarıyla ilgili daha detaylı bilgiler [17] ve [53] da bulunmaktadır. Çizelge 3.1-3.3 ile verilen öğrenme fonksiyonlarında Glock vd.' nin [53] kullandığı notasyona uyulmuştur.

Çizelge 3.1' de görülen y_x, y_1 ve b sırasıyla x. birimi üretmek için gereken ortalama süreyi, ilk birimi üretmek için gereken süreyi ve öğrenme etkisi faktörünü ifade etmektedir. Burada, $b = \log(LR)/\log(2)$ ile belirlenir ve LR öğrenme oranıdır. Denklem 3.3' deki c, gerçek verilerden elde edilen standart zamanı, denklem 3.4 ve 3.6' daki B, önceki tecrübe düzeyini temsil etmektedir. Denklem 3.5 ve 3.6' da görülen $M \in [0,1]$, üretim sistemindeki makinenin öğrenme şiddetine etkisini belirleyen bir parametredir. Eğer $M = 1$ ise, işlemlerin otomatik bir makinede, $M = 0$ ise, manuel bir makinede yapıldığı ifade edilmiş olur.

Çizelge 3.1. Logaritmik-doğrusal öğrenme fonksiyonları

Wright [45]	$y_x = y_1 x^{-b}$	Log-linear model	(3.2)
Baloff [54]	$y_x = (y_1 - c)x^{-b} + c$	Plateau model	(3.3)
Carlson [55]	$y_x = y_1(x + B)^{-b}$	Stanford-B model	(3.4)
Jong [56]	$y_x = y_1(M + (1 - M)x^{-b})$	DeJong model	(3.5)
Carlson [55]	$y_x = y_1(M + (1 - M)(x + B)^{-b})$	S-curve	(3.6)

Çizelge 3.2' deki denklem 3.7' de görülen $P(x)$ üretim oranını, $P_{\max}$ maksimum üretim oranını, $y_1 = 1/P_1$ ve son olarak λ hızı temsil eden pozitif bir parametreyi ifade etmektedir. Denklem 3.8' deki α ve β değişkenleri deneysel olarak belirlenen parametrelerdir. Denklem 3.9' daki y_t, y_c, y_f ve τ ifadeleri sırasıyla t zamanındaki üretim oranını, ilk üretim oranını, maksimum üretim oranıyla ilk üretim oranı arasındaki farkı ve belirli bir eğri için zaman sabitini belirtmektedir.

Çizelge 3.2. Üstel öğrenme fonksiyonları

Levy [57]	$P(x) = P_{\max}\left(1 - e^{-\lambda x}\right) + (x^b / y_1)e^{-\lambda x}$	(3.7)
Pegels [58]	$y_x = \alpha y_1^{x-1} + \beta$	(3.8)
Bevis vd. [59]	$y_t = y_c + y_f\left(1 - e^{-t/\tau}\right)$	(3.9)
Knecht [60]	$y_x = y_1 x^{-b} e^{vx}$	(3.10)
Kvalseth [61]	$y_x = \beta_1 + \beta_2 e^{-m(x-1)}$	(3.11)

Denklem 3.10' daki v pozitif değer alan bir parametreyi ifade etmektedir. Denklem 3.11' deki y_x çevrim süresini, x çevrim miktarını (sayısını), β_1 en yüksek performans seviyesini, β_2 öğrenmeye bağlı elde edilen en yüksek performans gelişimini ve son olarak m verilen bir β_2 değeri için geçerli olan öğrenme oranını ifade etmektedir.

Çizelge 3.3. Hiperbolik öğrenme fonksiyonları

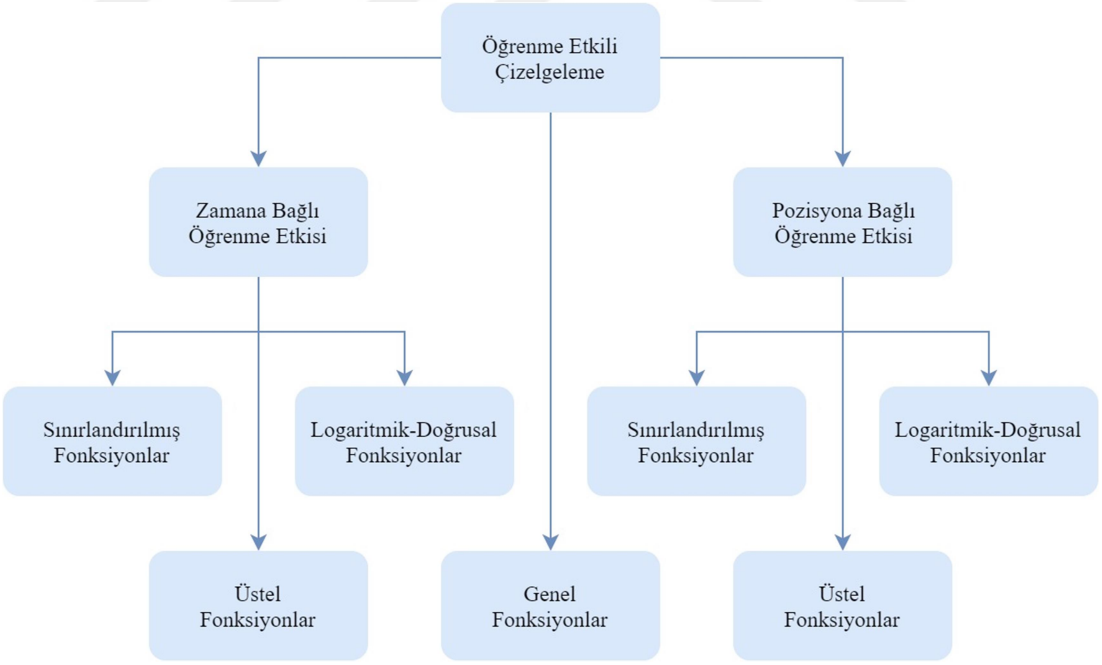
Mazur ve Hastie [62]	$y_t = k(t/(t+R))$	(3.12)
Mazur ve Hastie [62]	$y_t = k((t+p)/(t+p+R))$	(3.13)

Çizelge 3.3' deki y_t öğrenme veya performans değerinini, t iş üzerinde harcanan zamanı, yapılan tekrarı veya iş için alınan eğitim değerini, k öğrenme için asimptotu, R öğrenme şiddetinin k' ya yaklaştığı oran değerini ifade etmektedir.

Denklem 3.12' deki fonksiyon bu yapısıyla iki değişkenli bir model olup öğrenme etkisinin birikimli şiddetini ölçmektedir. Denklem 3.13' deki fonksiyon ile yanlış olan motor becerilerin yerine yenilerinin getirilmesini ifade eden yeni bir model elde edilmiştir. Buradaki p değeri, ilk durumdaki eğitim (kabiliyet, beceri) seviyesini zaman birimi veya tekrar sayısı cinsinden gösteren bir parametredir.

3.1.2. Çizelgelemede Öğrenme Etkisi

Klasik çizelgeleme teorisinde, çoğu çalışma, planlama süreci boyunca işlerin işlem sürelerinin sabit ve değişmez olduğunu varsayar. Ancak, öğrenme etkisine bağlı olarak işlerin işlem sürelerinin değişime tabi olduğu durumlar da vardır. Çizelgeleme çerçevesinde öğrenme etkisi, işlemcinin edindiği deneyimi temsil eden bir parametreye göre azalan fonksiyonlarla ifade edilmektedir. Biskup [21] ve Cheng ve Wang [63], öğrenme etkisini çizelgeleme problemlerinde kullanarak bu alandaki çalışmalara öncülük etmişlerdir. Devamında, öğrenme kavramı klasik çizelgeleme teorisinde yer bulmuş ve çeşitli üretim sistemlerinde öğrenme etkisinin varlığını ifade eden çizelgeleme modelleri yaygınlaşmıştır [16]. Bu kapsamda çeşitliliği artan öğrenme fonksiyonları, pozisyona bağlı ve zamana bağlı öğrenme etkileri olarak iki sınıfta toplanmıştır. Yakın zamanda Azzouz vd.' nin [18] yaptığı literatür araştırmasında, çizelgeleme problemlerinde kullanılan öğrenme fonksiyonları Şekil 3.1' de olduğu gibi sınıflandırılmıştır.



Şekil 3.1. Çizelgelemede kullanılan öğrenme modellerinin sınıflandırılması (Azzouz vd.' nden [18] uyarlanmıştır)

3.2. Unutma Etkisi

Öğrenme sürecinde kazanılan bazı tecrübeler performans artışına yol açarken, öğrenme sürecindeki duraksamalar ve çeşitli nedenler, kazanılan bu verimliliğin bir miktar azalmasına neden olur. Bir nevi öğrenmenin zıttı olan, performans düşüşüne yol açan ve ancak öğrenme etkisi ile beraber gözlemlenebilen bu olgu unutma etkisidir. Badiru [41], unutma etkisinin nedenlerini 3 grupta toplamıştır: (a) üretimdeki planlı duruşlar nedeniyle aralıklı olarak meydana gelen unutma, (b) makine arızaları nedeniyle rasgele veya tahmini olarak meydana gelen unutma, (c) tekdüzelik ve uzun süreli çalışma gibi bazı doğal nedenlere bağlı meydana gelen unutma. Bunun yanında, yapılan işin niteliği, üretim araçları, yerleşim düzeni ve işçilerin kabiliyeti gibi üretim parametrelerindeki değişimin neden olduğu bozulma da unutmaya sebep olan bir diğer etkendir [64].

Aralıklı üretim periyotlarında, yeterli uzunlukta bir duruş olduğu varsayımıyla, yeni bir üretim periyodu başlarken, bir önceki periyottan elde edilen faydaların (öğrenme) kayıpsız bir şekilde devam etmesi mümkün olamamaktadır. Bundan dolayı, üretime yeniden başlarken ölçülen üretim hızı, üretimin durduğu anda ölçülen üretim hızı kadar yüksek olmamaktadır. Belirli bir sırayla farklı işlerin yapıldığı durumlarda, üretimdeki her duruştan sonra, tecrübe ve üretim kabiliyeti açısından başlangıçtaki seviyeye gelindiğini varsaymak da oldukça yanıltıcıdır. Bu nedenle, az da olsa önceki deneyimlerden kalan bir miktar tecrübe ve üretim kabiliyetinin olduğu kabul edilmelidir. Önceki deneyimlerden elde edilen kazanç (tecrübe, üretim kabiliyeti, el alışkanlığı vs.) miktarı duruş uzunluğuna bağlıdır [65]. Dolayısıyla, eğer bir üretim süreci farklı uzunluktaki parçalara ayrılmışsa ve bu parçalar arasında değişik uzunlukta duruşlar söz konusuysa, gerçekleşen işlem süreleri, üretim sürecini oluşturan bu parçaların ve duruşların uzunluğuna bağlıdır.

Önceki çalışmaların bir kısmı, öğrenme etkisinin tersi olan unutma etkisini incelemiş ve analiz etmiştir [38]. Jaber ve Bonney [66], bu çalışmalarının önemli bir kısmında öğrenme ve unutma etkilerinin bir arada olduğu durumları ifade etmişlerdir. Bunun yanında, servis endüstrisinde öğrenme ve unutma etkilerinin bir arada ele alındığı çalışmalar da oldukça yaygınlaşmıştır [67]. Bu tezde, üretim süreçlerindeki çeşitli

duruşlardan kaynaklanan unutma etkisi üzerinde durulmuştur. Aşağıda anlatılan çalışmaların çoğu unutma etkisini, öğrenme etkisiyle kazanılan kabiliyetin sağladığı verimliliğin azalması olarak ele almıştır. Bu azalma genellikle, kesinti uzunluğunun ve kesinti öncesi mevcut öğrenme miktarının bir fonksiyonu olarak tanımlanmıştır.

3.2.1. Duruş Nedeniyle Meydana Gelen Unutma

Badiru [41], öğrenme sürecindeki kesintilerden kaynaklanan çok değişkenli bir unutma etkisi modeli önermiştir. Steedman [65], duruş süresiyle unutma miktarı arasındaki ilişkiyi tanımlamıştır. Elde ettiği bulgulara göre, ardışık üretim süreçleri arasındaki duruşlardan kaynaklanan unutma etkisinin şiddeti, üretim süreçleri boyunca kazanılan kabiliyetler sayesinde azalmaktadır. Sule [43], öğrenme ve unutmanın aynı eğriyle formülüze edildiği bir model önermiştir. Modellenen eğri üzerinde, öğrenme etkisi aşağı yönde, öğrenme sürecindeki duruşlar (unutma etkisi) ise yukarı yönde harekete yol açmaktadır.

Sule [68], çeşitli ürünlerin aynı tesisteki üretim döngüsüne dair planlama kararlarında öğrenme ve unutma etkilerini dikkate alan ekonomik parti büyüklüğü çizelgeleme problemini ele almıştır. Tanımlanan problemde, bir ürünle ilgili üretim kararı ancak o ürünün miktarı mevcut talepten daha az ise verilmektedir. Dolayısıyla, ilgili ürünün tekrar üretilmesine kadar geçen süreçte unutmaya yol açacak boyutta bir duruş olabilmektedir. Smunt [69], tanımladığı parti büyüklüğü çizelgeleme probleminde, ardışık iki üretim periyodu arasındaki duruştan kaynaklanan unutma etkisini, sistemde herhangi bir ayar faliyeti olmamasına rağmen ayar maliyeti olarak dikkate almıştır.

Globerson vd. [38], ardışık işler arasındaki molaların unutmaya sebep olduğu deneysel bir çalışma yapmışlardır. Bu deneylerdeki dikkat çekici bulgu, en belirgin düzeyde performans düşüşünün mola süresinin bitişinden hemen sonraki zamanlarda olduğudur. Böylece, unutma şiddetinin mola uzunluğuna ve moladan önceki üretim süreçlerinde kazanılan tecrübeye bağlı olduğu kanısına varılmıştır. Bir başka deneysel çalışmada Bailey [70], unutmayla prosedürel görevler arasındaki ilişkiyi

açıkça tanımlamayı başardı. Deney sonuçları, sürekli işlerden daha çok aralıklı üretim süreçlerinde belirgin ve yüksek düzeyde ölçülebilir unutmanın oluştuğunu göstermiştir. Wixted ve Ebbesen [71], çeşitli unutma modelleri üzerinde çalışmış ve faydalı bulgularla sonuçlanan deneyler yapmışlardır. Genel olarak deneyler, öğrenme sürecini aksatan faaliyetlerin unutmaya yol açtığını göstermiştir. İlaveten, unutmanın şiddetinin, öğrenmeyi aksatan süreçteki dikkat dağıtıcı öğelerin ve öğrenme yoğunluğunun bir fonksiyonu olduğu belirtilmiştir.

Shtub vd. [22], Bailey' in [70] çalışmasındaki modele dayanan deneysel bir çalışma yapmıştır. Çalışma, otomasyon düzeyi yüksek bir fabrikadaki veri girişi görevlerine ve duruşların bu görevler üzerindeki etkisine odaklanmıştır. Çalışmanın temel katkısı, Bailey' in [70] anahtar bulgularını, manuel bir çalışma ortamından ziyade bilgisayar destekli bir çalışma ortamında test etmiş olmasıdır. Deneylerin sonucu olarak, öğrenme, unutma ve yeniden öğrenme ilişkisini tahmin etmek için bir model geliştirilmiştir. Bir başka deneysel çalışmada Dar-El vd. [72], duruş ve unutma arasındaki ilişkiyi $b^f = b_0(q+1)^{-0.152}$ ile ifade etmişlerdir. Buradaki b^f ile unutmayı da içeren öğrenme oranı, b_0 ile unutmayı içermeyen öğrenme oranı ve q ile duruş periyodu gösterilmiştir.

Jaber ve Bonney [73], unutma etkisini temsil etmesi bakımından duruş süresini de formülüze ettikleri matematiksel bir öğrenme-unutma eğrisi (LFCM) modellemişlerdir. Yazarlar ayrıca, oluşturdukları modeli Salameh vd.' nin [74] ve Globerson vd.' nin [38] çalışmalarında tanımlanan vaka analizlerinde kullanmışlardır. Jaber ve Bonney [75] bir diğer çalışmalarında, Carlson ve Rowe [76] tarafından geliştirilen değişken regresyon değişken unutma (VRVF) modeli, Elmaghraby [77] tarafından geliştirilen ekonomik üretim miktarı (EMQ) modeli ve LFCM modeli arasında deneysel bir kıyaslama yapmışlardır. Bu çalışma neticesinde, LFCM modelinin daha tutarlı sonuçlar verdiğini göstermişlerdir.

Chiu [78], ekonomik parti büyüklüğü problemi için üstel öğrenme-unutma fonksiyonu tanımlamıştır. Unutma oranının (f) duruş süresinden bağımsız olduğu durumlar için $a_j = a_{j-1}[f + (1-f)(Q+1)^{-b}]$ fonksiyonu, ardışık üretim periyotları arasındaki mola süresine (t) bağlı olduğu durumlar için $a_j =$

$a_{j-1}[f(t) + (1 - f(t))(Q + 1)^{-b}]$ fonksiyonu önerilmiştir. McCreery ve Krajewski [79], öğrenme etkisiyle beraber sürekli üretim sürecindeki duruşlardan kaynaklanan unutma etkisini de içeren bir montaj hattı problemi üzerinde çalışmışlardır. Nembhard ve Uzumeri [80], anlık duruşlardan ziyade duruşların kümülatif süresini dikkate alarak unutma etkisini formülüze etmişlerdir. Bundan dolayı, yakın zamanda gerçekleşme indeksi (recency index - RC) olarak adlandırdıkları bir değişken (R_x) kullanmışlardır, $R_x = (\sum_{i=1}^{x}(t_i - t_0))/x(t_x - t_0)$. Bu formüldeki $t_x - t_0$ ifadesi birim x için geçen süreyi temsil etmektedir. Shafer vd. [81], çok fazla çalışan ve iş esnekliğinin olduğu bir montaj hattının verimliliğini belirlemek için RC modelini kullanmışlardır. Nembhard ve Shafer [82], parametre heterojenliğinin RC modeli üzerindeki etkilerini araştırdıkları deneysel bir çalışma yapmışlardır.

Jaber vd. [83], çalışan esnekliğinin olduğu bir sistemde, LFCM modeli ile VRVF modelini kıyaslamışlardır. Yapılan bu kıyaslamayı unutma için belirledikleri yedi özelliğe dayandırmışlardır. Jaber ve Sikström [84], LFCM, RC ve Sikström ve Jaber [85] tarafından geliştirilen güç entegrasyonu difüzyonu (the power integration diffusion – PID) modellerinin kıyaslandığı bir sayısal çalışma yapmışlardır. Yapılan çalışmada, bireysel öğrenme araştırılmıştır ve unutmanın sadece mola uzunluğuna dayandığı varsayılmıştır. Davidovitch vd. [86], bir proje yönetimi simülasyonu ile iki mola dönemi senaryosu arasında önemli bir fark olduğunu göstermişlerdir, öyle ki uzun bir mola unutma etkisini artırmıştır.

Teyarachakul vd. [87], unutmanın duruşa bağlı olduğu bir S-curve öğrenme-unutma fonksiyonu tanımlamıştır. Fonksiyon $F(X, I) = X + [1 - (bI + 1)e^{-bI}][T(1) - X]$ ile ifade edilirken, $F(X, I)$ terimi I üretim periyoduna ait mola süresinden sonraki birim üretim zamanını, X terimi unutmadan bağımsız birim üretim zamanını, $b > 0$ terimi unutma çarpanını ve $T(1) - X$ terimi de kümülatif öğrenme miktarını yani deneyimi temsil etmektedir. Glock ve Jaber [88], LFCM modelini dikkate alarak çok aşamalı bir üretim-stok problemi üzerinde çalışmışlardır. Biel ve Glock [89], unutmanın temel sebebinin değişik üretim süreçleri arasındaki duruşlar olduğu varsayımıyla, çok aşamalı bir üretim süreci için bir simülasyon modeli geliştirmişlerdir. Korytkowski [90], birçok alanda yetenekli çalışanların olduğu bir yeterlilik yönetim probleminde LFCM modelini kullanmışlardır.

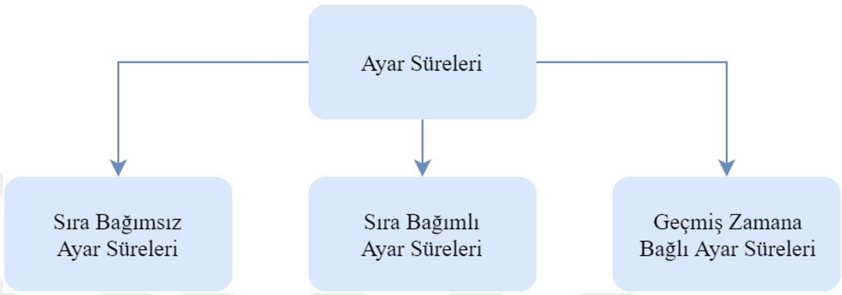
3.2.2. Duruş Dışı Nedenlerle Meydana Gelen Unutma

Yukarıda görüldüğü üzere, öğrenme sürecinin kesintiye uğradığı üretim sistemlerinde unutma fonksiyonları uygulanabilir olmaktadır [41]. Bunun yanında, iş çeşidi, kullanılan araçlar, tesis yerleşimi, çalışanlar vb. gibi üretim sistemini oluşturan bileşenlerdeki değişimden kaynaklanan unutma etkisinden de bahsetmek mümkündür [64]. Bundan dolayı mevcut bazı modeller, unutmanın bir sonucu olarak performansta düşüşe neden olan öğrenme sürecindeki kesinti veya duruşları dikkate almazlar. Unutmayı bu şekilde ele alan çalışmalarda, çok değişkenli modellerden ziyade genellikle tek değişkenli modeller kullanılmaktadır [41]. Kher vd. [91], Carlson ve Rowe [76] tarafından ifade edilen öğrenme/unutma/öğrenme (LFL) konseptini kullanarak bir eş kaynak-kısıtlı (dual resource constrained - DRC) sistem modellemiştir. DRC sistemi, çalışanların görevlerinin değiştirilmesini gerektiren işçi esnekliği içerdiğinden dolayı, tanımlanan bu sistemde unutmanın oluştuğu düşünülmüştür. Grosse vd. [92], depo çalışanının işyerine olan aşinalığını kaybetmesine neden olan bir depo yerleşim problemi ile unutmanın etkisinin gerçekleştiğini göstermiştir.

3.3. Ayar Süresi

Endüstriyel süreçlerde ayar faaliyetleri, işlerden birini tamamlayıp bir sonrakine geçerken, sıradaki işin tamamlanmasına direkt katkı sunmayan işlemlerden oluşur. Tamamlanan son uygun ürünle bir sonraki uygun ürünün başlaması arasında ayar işlemleriyle geçen zamana ayar süresi denilmektedir. Bu nedenle, ayar işlemleri bir katma değer faktörü olarak düşünülmez [9] ve meydana getirdiği değişim nedeniyle üretim ve hizmet süreçlerindeki maliyetli aksaklıkları temsil eder [93]. Örneğin, baskılı devre kartı montajında ayar işlemlerinin neden olduğu kapasite kaybının %20 ile %50 arasında değiştiği belirtilmiştir [94]. Ayar işlemleri, bir kaynağın çeşitli ürünler tarafından ortak kullanımı söz konusu olduğunda yapılmaktadır. Planlama konusunda karar vericiler ayar sürelerini genel olarak ihmal etme eğiliminde olmuşlardır. Bunun neticesinde, ayar işlemleri için harcanan zaman (veya mali unsurlar) göz ardı edilmiş veya üretim faaliyetlerinin bir parçası olarak kabul

edilmiştir. Bu tutum bazı sistemler için makul görünse de, bazı sistemlerde de net olarak belirlenip ayrı olarak değerlendirilmesi gerekmektedir. Ayar sürelerinin ayrı olarak ele alındığı durumlarda, yapısal olarak iki tip ayar süresinden bahsedilmektedir. Bunlardan birinde, ayar süresini sadece yapılacak olan iş belirlerken (sıra bağımsız ayar süresi - BZAS), bir diğerinde, hem tamamlanan hem de yapılacak olan iş belirlemektedir (sıra bağımlı ayar süresi - BLAS) [9].



Şekil 3.2. Ayar sürelerinin sınıflandırılması

Ayar sürelerinin yapısal olarak sınıflandırılması Şekil 3.2' de gösterildiği gibidir. Ayar sürelerinin ayrıştığı bu noktada, ilk kez Koulamas and Kyparisis [95] tarafından tanımlanan geçmiş zamana bağlı ayar süresinden (GZAS) bahsetmek gerekmektedir. İleri teknolojik yöntemlerle üretilen devre kartlarına bir grup elektronik devre elemanının bir arada eklenme süreci, bu ayar süresinin çıkış kaynağı olmuştur. Tanımlanan ayar süresi, mevcut ana kadar geçen birikimli işlem süresinin doğrusal fonksiyonudur [96]. Dolayısıyla, bir işlemden önce yapılması gereken ayar işlemlerinin süresi bir problem girdisi olarak verilmeyen ve problem çözümü için oluşturulan çizelgeye göre değişiklik gösteren bir yapıdadır. Bir işe ait GZAS, daha önce tamamlanan işlem sürelerine bağlıdır ve bu nedenle, BZAS ve BLAS' dan farklıdır. Bu yapıya bağlı olarak, GZAS çeşitli öğrenme etkisi modelleriyle formülüze edilebilmiştir. Son zamanlarda yapılan bazı seri-parti [97, 98] ve paralel-parti [99] üretim çalışmalarında tanımlanan ayar süreleri de GZAS ile aynı yapıdadır. GZAS kapsamında yapılan çalışmaların nerdeyse tamamı öğrenme etkisi içermesine rağmen, yapısal farklılıktan dolayı bu tezin konusu dışında bırakılmıştır.

4. LİTERATÜR TARAMASI

4.1. Öğrenme Etkisi İçeren Çizelgeleme Problemleri

Bu tezde üzerinde durulan öğrenme modelleri, pozisyona bağlı ve zamana bağlı öğrenme etkileridir. Bu kapsamda ortak bir notasyon oluşturmak için pozisyona bağlı öğrenme etkisi LP, zamana bağlı öğrenme etkisi de LT ile ifade edilecektir. Çizelgeleme alanında en sık karşılaşılan öğrenme modelleri, Biskup [21] tarafından ilk kez uygulanan LP (denklem 4.1), Kuo ve Yang [100] trafından ilk kez uygulanan LT (denklem 4.7) ve Koulamas ve Kyparisis [101] tarafından ilk kez uygulanan LT' nin farklı bir versiyonudur (denklem 4.44). Kullanılan diğer modeller genellikle bu 3 modelden türetilmiştir. Öğrenme etkisi kapsamında, literatürde karşılaşılan öğrenme fonksiyonları Çizelge 4.1' de verilmiştir. Öğrenme modellerindeki ortak ifadeler olan p_j^A (p_{ij}^A) ve p_j (p_{ij}) sırasıyla, gerçekleşen işlem süresini ve normal işlem süresini temsil etmektedir.

Öğrenme etkisi kapsamında, literatürde karşılaşılan amaç fonksiyonları Çizelge 4.2' de verilmiştir. Amaç fonksiyonlarında, denklem 4.48' deki p ifadesi tamamlanma zamanının bir kuvvetini, denklem 4.64' deki C ifadesi $AG1$ müşterisine ait işlerin maksimum tamamlanma zamanı için bir üst sınır değerini, denklem 4.65' deki T ifadesi $AG2$ müşterisine ait işlerin maksimum gecikmesi için bir üst sınır değerini temsil etmektedir. Ayrıca, birçok amaç fonksiyonunda kullanılan ortak ifadeler d, q, D, v_{ij}, u_{ij} ve W_j sırasıyla, teslim tarihini, teslim tarihi gevşekliğini, teslim tarihi aralığını, bir işe ayrılan kaynak maliyetini, bir işe ayrılan kaynak miktarını ve bir işin bekleme zamanını temsil etmektedir. İncelenen çizelgeleme problemlerinde kullanılan çözüm yöntemleri Çizelge 4.3' de, dikkate alınan özel durumlar, kısıtlar ve özellikler ise Çizelge 4.4' de verilmiştir.

Çizelge 4.1. Çizelgeleme problemlerinde kullanılan öğrenme fonksiyonları

Sıra No	Öğrenme Fonksiyonu	Denklem No
1	$p_j^A = p_j r^a \qquad a \leq 0$	(4.1)
2	$p_{ij}^A = p_{ij} r^a k^b \qquad a, b < 0$ $i, r \rightarrow$ grup indisi $\quad j, k \rightarrow$ bir grup içindeki sıra indisi	(4.2)
3	$p_{ij}^A = p_{ij} \left(1 - \left(\sum_{k=1}^{r-1} p_{i[k]} / \sum_{k=1}^{n} p_{ik}\right)\right)^a b^{r-1} + \alpha t$ $a \geq 1, 0 < b < 1, \alpha \geq 0 \rightarrow$ yaşlanma oranı, $j \rightarrow$ iş indisi $t \geq 0 \rightarrow j$ işinin başlama zamanı, $i \rightarrow$ makine indisi	(4.3)
4	$p_j^A = p_j(M + (1 - M) r^a) + \alpha t$ $a \leq 0, \alpha \geq 0 \rightarrow$ yaşlanma oranı, $t \geq 0 \rightarrow j$ işinin başlama zamanı $0 \leq M \leq 1 \rightarrow$ makinenin teknoloji düzeyi	(4.4)
5	$p_{ij}^A = p_{ij} \left(c_1 a^{\sum_{k=1}^{r-1} p_{i[k]}} + c_2\right) b^{r-1}$ $a > 0, \quad c_1, c_2 \geq 0, \quad c_1 + c_2 = 1, \quad b \leq 1 \rightarrow LR$ $j \rightarrow$ iş indisi, $\quad i \rightarrow$ makine indisi	(4.5)
6	$p_j^A = (p_j + \alpha t) r^a \qquad \alpha > 0 \rightarrow$ yaşlanma oranı $a \leq 0, \qquad t \geq 0 \rightarrow j$ işinin başlama zamanı	(4.6)
7	$p_j^A = p_j \left(1 + \sum_{k=1}^{r-1} p_{[k]}\right)^a \qquad a \leq 0$	(4.7)
8	$p_j^A = \begin{cases} p_j \left(1 - \int_0^t f(x)dx\right) & \text{eğer } t \leq k_0 \\ p_j \left(1 - \int_0^t f(x)dx + \int_0^{t-k_0} g(x)dx\right) & \text{eğer } t > k_0 \end{cases}$	(4.8)

Çizelge 4.1. Devam

Sıra No	Öğrenme Fonksiyonu	Denklem No
	$t = \sum_{k=1}^{r-1} p_{[k]}$, $k_0 \geq 0$ → unutmanın başladığı eşik değeri $f(x)$ → öğrenme fonksiyonu, $g(x)$ → unutma fonksiyonu	
9	$p_j^A = p_j f\left(\sum_{k=1}^{r-1} \beta_k p_{[k]}, r\right)$ $\forall k$ için $\beta_k > 0$	(4.9)
10	$p_{ij}^A = p_{ij}\left(1 + \sum_{k=1}^{r-1} \beta_k \ln\left(p_{i[k]}\right)\right)^a r^b$ $a, b \leq 0$, $\forall k$ için $\beta_k > 0$, $k < l$ için $\beta_k < \beta_l$ j → iş indisi, i → makine indisi	(4.10)
11	$p_{ij}^A = p_{ij} \prod_{k=0}^{r-1} l_k$ $l_0 = 1$, $0 < l_k \leq 1$ j → iş indisi, i → makine indisi	(4.11)
12	$p_{ij}^A = (1/V_i)\left(\left(p_{ij} + \alpha_j(r-1)\right) r^{a_i}\right)$ α_j → j işinin yaşlanma oranı, j → iş indisi a_i → i makinesinin öğrenme etkisi faktörü V_i → i makinesinin hız faktörü, i → makine indisi	(4.12)
13	$p_j^A = \begin{cases} p_j & \text{tamirattan önce ise} \\ \beta_j p_j & \text{diğer durumda} \end{cases}$ $0 < \beta_j \leq 1$ $T_{MA}(t) = t_0 \pm \alpha t$ → zamana bağlı tamirat süresi $T_{MA}(r) = \tau_r + r$ → pozisyona bağlı tamirat süresi $t_0 > 0$ → standart tamirat süresi, $\alpha > 0$ → yaşlanma oranı $t \geq 0$ → tamiratın başlama zamanı, τ_r → pozisyon sabiti	(4.13)
14	$p_j^A = p_j f\left(\sum_{k=1}^{r-1} \beta_k \left(p_{[k]} \vee p_{[k]}^A\right), r\right)$ $\forall k$ için $\beta_k > 0$	(4.14)

Çizelge 4.1. Devam

Sıra No	Öğrenme Fonksiyonu	Denklem No
15	$p_j^A = p_j\left(1+\left(\sum_{k=1}^{r-1} p_{[k]}/\sum_{k=1}^{n} p_{[k]}\right)\right)^a r^b$ $a, b \geq 0 \rightarrow$ yaşlanma durumunda $a, b < 0 \rightarrow$ öğrenme durumunda	(4.15)
16	$p_{ij}^A = \begin{cases} p_{ij} r^a k^b + \alpha t - \beta_{ij} u_{ij} & \text{doğrusal kaynak kullanımı} \\ \left(\left(p_{ij} r^a k^b\right)/u_{ij}\right)^l + \alpha t & \text{konveks kaynak kullanımı} \end{cases}$ $s_i^A = \begin{cases} s_i r^c - \beta_i u_i & \text{doğrusal ayar süresi} \\ \left((s_i r^c)/u_i\right)^l & \text{konveks ayar süresi} \end{cases}$ $0 \leq u_{ij} \leq \bar{u}_{ij} < \left(p_{ij} m^a (n_m)^b\right)/\beta_{ij}, \quad a, b, c \leq 0$ $0 \leq u_i \leq \bar{u}_i < (s_i m^c)/\beta_i, \quad l > 0 \rightarrow$ verilen bir sabit değer $\alpha \geq 0 \rightarrow$ yaşlanma oranı, $t \geq 0 \rightarrow j$ işinin başlama zamanı $\bar{u}_{ij} \rightarrow i$ grubundaki j işi için maksimum kaynak miktarı $\bar{u}_i \rightarrow i$ grubu için maksimum kaynak miktarı $\beta_{ij} > 0 \rightarrow i$ grubundaki j işi için eksiltme oranı $\beta_i > 0 \rightarrow i$ grubu için eksiltme oranı, $r \rightarrow$ grup pozisyonu $k \rightarrow$ gruptaki iş pozisyonu, $m \rightarrow$ grup sayısı	(4.16)
17	$p_{ij}^A = p_{ij} r^a / s_i \quad a \leq 0 \quad i \rightarrow$ makine indisi $s_i \rightarrow i$ makinesi için ayar süresi $j, r \rightarrow$ iş ve pozisyon indisi	(4.17)
18	$p_{ij}^A = p_{ij}\left(c_1 a^{\sum_{k=1}^{r-1} w_{ik} p_{i[k]}} + c_2\right) b^{r-1}$ $0 < a, b \leq 1, \quad c_1, c_2 \geq 0, \quad c_1 + c_2 = 1$ $j \rightarrow$ iş indisi, $i \rightarrow$ makine indisi, $k < l$ için $w_{ik} < w_{il}$	(4.18)
19	$p_j^A = p_j f(x) \qquad f(x)\begin{cases} z_0 & x < t_1 \\ z_1 & t_1 \leq x < t_2 \\ \vdots & \vdots \\ z_m & t_m \leq x \end{cases}$	(4.19)

Çizelge 4.1. Devam

Sıra No	Öğrenme Fonksiyonu	Denklem No
	$t_1 < t_2 < \cdots < t_m < \sum_{j=1}^{n} p_j$ Eğer $z_0 = 1 > z_1 > z_2 > \cdots > z_m > 0$ ise öğrenme etkisi Eğer $z_0 = 1 < z_1 < z_2 < \cdots < z_m < \infty$ ise yaşlanma etkisi	
20	$p_{ij}^A = p_{ij}\max\{r^a, \rho\}$ $a \leq 0$ $i \rightarrow$ makine indisi $0 \leq \rho \leq 1 \rightarrow$ budama parametresi	(4.20)
21	$p_j^A = p_j f\left(\sum_{k=1}^{r-1} \beta_k p_{[k]}, r\right)$ $k < l$ için $\beta_k \leq \beta_l$ $s_j = b\sum_{k=1}^{r-1} p_{[k]}$ $b \geq 0 \rightarrow$ normalizasyon sabiti	(4.21)
22	$p_j^A = p_j f\left(\sum_{k=1}^{r-1} \beta_k p_{[k]}\right) g(r)$ $k < l$ için $\beta_k \leq \beta_l$ $f(\sum_{k=1}^{r-1} \beta_k p_{[k]}) \rightarrow$ LT $g(r) \rightarrow$ LP	(4.22)
23	i) $p_j^A = p_j \alpha(t)(\beta a^{r-1} + \gamma)$ $0 < a \leq 1$ ii) $p_j^A = p_j(\alpha(t) + \beta a^{r-1} + \gamma)$ $\beta > 0$ $\gamma \geq 0$ $\alpha(t) \rightarrow$ artan fonksiyon $t \rightarrow j$ işinin başlama zamanı	(4.23)
24	$p_j^A = p_j f\left(\sum_{k=1}^{r-1} \beta_k p_{[k]}\right) g(r)$ $k < l$ için $\beta_k \leq \beta_l$ $f(\sum_{k=1}^{r-1} \beta_k p_{[k]}) \rightarrow$ LT $g(r) \rightarrow$ LP $q_j = \gamma \sum_{k=1}^{r-1} p_{[k]}^A$ $C_j = p_j^A + q_j + \sum_{k=1}^{r-1} p_{[k]}^A$	(4.24)
25	$p_j^A = p_j r^a$ $a \leq 0$ $t_0 \rightarrow$ ilk işin başlama zamanı $C_j = p_j^A + q_j + t_0 + \sum_{k=1}^{r-1} p_{[k]}^A$ $q_j = \gamma\left(t_0 + \sum_{k=1}^{r-1} p_{[k]}^A\right)$	(4.25)

Çizelge 4.1. Devam

Sıra No	Öğrenme Fonksiyonu	Denklem No
26	$p_j^A = f(r) \qquad f(r)\begin{cases} z_0 & 1 \le r < t_1 \\ z_1 & t_1 \le r < t_2 \\ \vdots & \vdots \\ z_m & t_m \le r \end{cases}$ $1 \le t_1 < t_2 < \cdots < t_m \le n, \qquad z_0 \ge z_1 \ge \cdots \ge z_m$	(4.26)
27	$p_{ij}^A = \begin{cases} p_{ij} + b_{ij}\max\{v - 1 - g_{ij}, 0\} & \text{yaşlanma} \\ p_{ij} - b_{ij}\min\{v - 1, g_{ij}\} & \text{öğrenme} \end{cases}$ $g_{ij} \rightarrow$ eşik değeri $\qquad v \rightarrow j$ işinin çizelgedeki pozisyonu $i \rightarrow$ makine indisi $\qquad b_{ij} \rightarrow$ yaşlanma/öğrenme çarpanı	(4.27)
28	$p_j^A = \begin{cases} p_j f\left(\sum_{k=1}^{r-1} \beta_k p_{[k]}, r\right) & \text{eğer } r \le m \\ p_j f\left(\sum_{k=1}^{r-1} \beta_k p_{[k]}, r\right) g(r) & \text{eğer } r > m \end{cases}$ $k < l$ için $\beta_k \le \beta_l, \ m \rightarrow$ unutmanın başladığı pozisyon değeri $\forall t \ge 0$ ve $1 \le r_1 \le r_2$ için $f(t, r_1)/f(t, r_2) \ge g(r_2)/g(r_1)$ $f(t, r) \rightarrow$ öğrenme fonksiyonu, $g(r) \rightarrow$ unutma fonksiyonu	(4.28)
29	$p_j^A = p_j\left(1 + \sum_{k=1}^{r-1} p_{[k]}\right)^{a_j} \qquad a_j \le 0$	(4.29)
30	$p_{ij}^A = p_{ij}\left(q_r + \sum_{k=1}^{r-1} \beta_k \ln\left(p_{i[k]}\right)\right)^a \qquad i \rightarrow$ makine indisi $k < l$ için $0 \le \beta_k \le \beta_l$ ve $0 \le q_k \le q_l, \qquad a < 0$	(4.30)
31	$p_j^A = \left(p_j + \alpha t\right) r^a \qquad \alpha \ge 0 \rightarrow$ yaşlanma oranı $a \le 0, \qquad t \ge 0 \rightarrow j$ işinin başlama zamanı $m_i = \mu + \sigma t_i \rightarrow i.$ tamirat işlemi, $\quad \mu > 0, \quad \sigma \ge 0$	(4.31)

Çizelge 4.1. Devam

Sıra No	Öğrenme Fonksiyonu	Denklem No
32	$p_j^A = p_j\left(\left(p_0 + \sum_{k=1}^{r-1} p_{[k]}\right)/\left(p_0 + \sum_{k=1}^{n} p_{[k]}\right)\right)^a r^b$ $a \leq 0$ ve $b \geq 0 \rightarrow$ yaşlanma durumunda $a \geq 0$ ve $b \leq 0 \rightarrow$ öğrenme durumunda	(4.32)
33	$p_j^A = p_j \max\left\{\left(1 - \left(\sum_{k=1}^{r-1} p_{[k]}/\sum_{k=1}^{n} p_{[k]}\right)\right)^a, \beta\right\}$ $a \geq 1 \quad 0 < \beta < 1 \rightarrow$ budama parametresi	(4.33)
34	$p_{ij}^A = p_{ij}\left(q(r) + \sum_{k=1}^{r-1} \beta_k p_{i[k]}\right)^a \quad i \rightarrow$ makine indisi $k < l$ için $0 \leq \beta_k \leq \beta_l, \quad a \leq 0, \quad q(r) \rightarrow$ artan fonksiyon	(4.34)
35	i) $p_j^A = p_j(1+t)^a\left(1 - (t/P)\right)^b r^c \qquad a, c \leq 0 \quad b \geq 1$ ii) $p_j^A = p_j\left(x(1+t)^a + y\left(1 - (t/P)\right)^b + zr^c\right)$ $t = \sum_{k=1}^{r-1} p_{i[k]},\ P = \sum_{k=1}^{n} p_{i[k]},\ x, y, z \geq 0,\ x + y + z = 1$	(4.35)
36	$p_{ikj}^A = p_{ikj} \prod_{l=0}^{r-1} a_{lk}/s_i \quad a_{0k} = 1,\ s_i \rightarrow i$ makinesinin hızı $j \rightarrow$ iş indisi, $\quad i \rightarrow$ makine indisi, $\quad k \rightarrow$ operatör indisi $0 < a_{lk} \leq 1 \rightarrow k.$ operatörün yaptığı $l.$ işin öğrenme oranı	(4.36)
37	i) $p_j^A = \left(p_j + bt\right)r^a \qquad a \leq 0 \quad b, c > 0$ ii) $p_j^A = p_j + bt^c r^a \qquad t \geq 0 \rightarrow j$ işinin başlama zamanı	(4.37)
38	$p_j^A = p_j\left(c_1 a^{\sum_{k=1}^{r-1} w_k p_{[k]}} + c_2\right) b^{r-1},\ c_1, c_2 \geq 0,\ c_1 + c_2 = 1$ $0 < a, b \leq 1, \quad k < l$ için $w_{ik} < w_{il},\ s_j = d\sum_{k=1}^{r-1} p_{[k]}, d \geq 0$	(4.38)

Çizelge 4.1. Devam

Sıra No	Öğrenme Fonksiyonu	Denklem No
39	$p_{ij}^A = p_{ij} f_i\left(\sum_{k=1}^{r-1} p_{i[k]}\right) g_i(r),\ s_i^A = s_i + b_i t,\ i \rightarrow$ grup indisi $b_i \geq 0 \rightarrow$ yaşlanma oranı, $t \rightarrow$ başlama zamanı, $j \rightarrow$ iş indisi	(4.39)
40	$p_{ij}^A = \begin{cases} p_{ij} r^a k^b - \beta_{ij} u_{ij} & \text{doğrusal kaynak kullanımı} \\ \left(\left(p_{ij} r^a k^b\right)/u_{ij}\right)^l & \text{konveks kaynak kullanımı} \end{cases}$ $s_i^A = s_i r^a \quad a, b < 0, \quad l > 0 \rightarrow$ verilen bir sabit değer $0 \leq u_{ij} \leq \bar{u}_{ij} < \left(p_{ij} m^a (n_m)^b\right)/\beta_{ij} \quad m \rightarrow$ grup sayısı $\bar{u}_{ij} \rightarrow i$ grubundaki j işi için maksimum kaynak miktarı $\beta_{ij} > 0 \rightarrow i$ grubundaki j işi için eksiltme oranı $k \rightarrow$ gruptaki iş pozisyonu, $r \rightarrow$ grup pozisyonu	(4.40)
41	$p_j^A = p_j f\left(\sum_{k=1}^{r-1} p_{[k]}\right) g(r),\ s_j = b \sum_{k=1}^{r-1} p_{[k]},\ g(r) \rightarrow$ LP $b \geq 0 \rightarrow j$ işinin ayar süresi çarpanı, $f(\sum_{k=1}^{r-1} p_{[k]}) \rightarrow$ LT	(4.41)
42	$p_j^A = p_j f\left(t_0 + \sum_{k=1}^{r-1} p_{[k]}\right) g(r),\ t_0 \geq 0, f \rightarrow$ LT, $g \rightarrow$ LP	(4.42)
43	$p_{ij}^A = p_{ij}\left(c_1\left(1 + \sum_{k=1}^{r-1} p_{i[k]}\right)^{a_i} + c_2 r^{b_i}\right)$ $s_i^A = s_i\left(c_3\left(1 + \sum_{k=1}^{l-1} s_{[k]}\right)^c + c_4 l^d\right) \qquad a_i, b_i, c, d \leq 0$ $0 \leq c_1, c_2, c_3, c_4 \leq 1 \qquad c_1 + c_2 = 1 \qquad c_3 + c_4 = 1$ $i \rightarrow$ grup indisi, $l \rightarrow$ grup pozisyonu, $r \rightarrow$ gruptaki iş pozisyonu	(4.43)
44	$p_j^A = p_j \left(1 - \left(\sum_{k=1}^{r-1} p_{[k]} / \sum_{k=1}^{n} p_k\right)\right)^b \qquad b \geq 1$	(4.44)

Çizelge 4.2. Çizelgeleme problemlerinde kullanılan amaç fonksiyonları

Sıra No	Amaç Fonksiyonu	Denklem No
1	C_{max}	(4.45)
2	$\sum_j C_j$	(4.46)
3	$\sum_j w_j C_j$	(4.47)
4	$\sum_j C_j^p$	(4.48)
5	$\sum_j w_j(1-e^{-rC_j})$	(4.49)
6	$\sum_i \sum_k \sum_j \lvert C_{ik} - C_{ij} \rvert$	(4.50)
7	$\sum_i C_{max}^i$	(4.51)
8	$k_1 C_{max} + k_2 \sum_j C_j$	(4.52)
9	T_{max}	(4.53)
10	$\sum_j T_j$	(4.54)
11	$\sum_j w_j T_j$	(4.55)
12	L_{max}	(4.56)
13	$\sum_j L_j$	(4.57)
14	$\sum_j E_j$	(4.58)
15	$\sum_j w_j E_j$	(4.59)
16	$\sum_j U_j$	(4.60)
17	$\sum_j w_j U_j$	(4.61)
18	$\sum_j (k_1 E_j + k_2 T_j)$	(4.62)
19	$\sum_{j \in AG2} U_j = 0$ şartıyla $\sum_{j \in AG1} w_j C_j$	(4.63)
20	$C_{max}^{AG1} \leq C$ şartıyla $\sum_{j \in AG1 \cup AG} w_j C_j$	(4.64)
21	$T_{max}^{AG2} \leq T$ şartıyla $\sum_{j \in AG1} C_j$	(4.65)
22	$\sum_i \sum_j T_{ij} + \sum_i \sum_j E_{ij}$ ile C_{max}	(4.66)
23	$\sum_j (k_1 E_j + k_2 T_j) + k_3 d$	(4.67)
24	$\sum_j (k_1 E_j + \varphi_j U_j) + k_2 d$	(4.68)
25	$\sum_j (k_1 E_j + \varphi_j U_j) + k_2 q$	(4.69)
26	$\sum_j (\varphi_j U_j + \gamma_j d_j)$	(4.70)
27	$k_1 C_{max} + \sum_j (k_2 E_j + k_3 T_j + k_4 d_j)$	(4.71)

Çizelge 4.2. Devam

Sıra No	Amaç Fonksiyonu	Denklem No
28	$\sum_j \left(k_1 C_j + k_2 E_j + k_3 T_j + k_4 d_j\right)$	(4.72)
29	$\sum_j \left(k_1 E_j + k_2 T_j + k_3 d_j + k_4 D_j\right)$	(4.73)
30	$k_1 C_{max} + \sum_j \left(k_2 E_j + k_3 T_j + k_4 d_j + k_5 D_j\right)$	(4.74)
31	$\sum_j \left(k_1 C_j + k_2 E_j + k_3 T_j + k_4 d_j + k_5 D_j\right)$	(4.75)
32	$k_1 d + \sum_j \left(k_2 E_j + k_3 f\left(T_j\right)\right)$	(4.76)
33	$k_1 q + \sum_j \left(k_2 E_j + k_3 f\left(T_j\right)\right)$	(4.77)
34	$k_1 C_{max} + k_2 \sum_i \sum_j v_{ij} u_{ij}$	(4.78)
35	$\sum_i \sum_j \left(k_1 C_{ij} + k_2 v_{ij} u_{ij}\right)$	(4.79)
36	$k_1 C_{max} + \sum_j \left(k_2 C_j + k_3 v_j u_j\right) + k_4 \sum_i \sum_j \left\|C_i - C_j\right\|$	(4.80)
37	$k_1 C_{max} + \sum_j \left(k_2 W_j + k_3 v_j u_j\right) + k_4 \sum_i \sum_j \left\|W_i - W_j\right\|$	(4.81)
38	$k_1 C_{max} + k_2 \sum_i v_i u_i + k_3 \sum_i \sum_j v_{ij} u_{ij}$	(4.82)
39	$k_1 \sum_i C_{max}^i + k_2 \sum_i \sum_j C_{ij} + k_3 \sum_i \sum_k \sum_j \left\|C_{ik} - C_{ij}\right\|$	(4.83)
40	$k_1 \sum_i C_{max}^i + k_2 \sum_i \sum_j W_{ij} + k_3 \sum_i \sum_k \sum_j \left\|W_{ik} - W_{ij}\right\|$	(4.84)

Çizelge 4.5-4.7’ deki sütun başlıkları REF., Ö.F., A.F., T.Ç.Y., Y.Ç.Y. ve P.Ö.K. sırasıyla ilgili çalışmanın referans listesindeki sıra numarasını, öğrenme fonksiyonu numarasını (Çizelge 4.1’ de sıra no ile belirtilen değer), amaç fonksiyon numaralarını (Çizelge 4.2’ de sıra no ile belirtilen değer), kesin çözüm yöntemlerinin kısaltmalarını (Çizelge 4.3’ de verilen), yaklaşık çözüm yöntemlerinin kısaltmalarını (Çizelge 4.3’ de verilen) ve problem özellikleri ve kısıtlarının kısaltmalarını (Çizelge 4.4’ de verilen) belirtmektedir.

Çizelge 4.3. Çizelgeleme problemlerinde kullanılan çözüm yöntemleri

PZ	Polinom zamanlı algoritma	BA	Bal arısı algoritması
MM	Matematiksel model	DS	Dal-sınır algoritması
DP	Dinamik programlama	GA	Genetik algoritma
TB	Tavlama benzetimi	TA	Tabu arama
İA	İteratif arama	PSO	Parçacık sürüsü optimizasyonu
KKO	Karınca kolonisi optimizasyonu	DKA	Değişken komşu arama

Çizelge 4.4. Çizelgeleme problemlerindeki çeşitli kısıtlar ve özellikler

YE	Yaşlanma etkisi	ÖDO	Özel durumda optimizasyon
UE	Unutma etkisi	GZAS	Geçmiş zamana bağlı ayar süresi
GÇ	Grup çizelgeleme	TAA	Teslim tarihi aralığı atama
İM	İki farklı müşteri	KİM	Makineler için kesintisiz işlem
KT	Kaynak tahsisi	ÖOM	Özdeş olmayan makineler
FGZ	Farklı geliş zamanları	TTA	Teslim tarihi atama
Kİİ	İşler için kesintisiz işlem	TZ	Tamirat zamanı
ÇMK	Çoklu müsaitlik kısıtı	KI	Karmaşıklık ıspatı
ÇÖ	Çok ölçütlü çizelgeleme	OT	Ortak teslim tarihi
DMH	Değişken makine hızları	FMH	Farklı makine hızları
BİS	Bulanık işlem süreleri	BTT	Bulanık teslim tarihleri
OA	Operatör atama	Öİ	Özdeş işler
GZTT	Geçmiş zamana bağlı teslim tarihi	ÇOİ	Eşit sayıda birden fazla operasyonu olan işler
OAÇ	Orantılı akış tipi çizelgeleme	İR	İş reddetme
BLAS	Sıra bağımlı ayar süresi	HAÇ	Hibrid akış tipi çizelgeleme

Çizelge 4.5. Tek makineli çizelgeleme problemleri

REF.	Ö.F.	A.F.	T.Ç.Y.	Y.Ç.Y.	P.Ö.K.
[10]	1	10	DS	GA, DKA	BLAS
[101]	44	1, 2	PZ	---	---
[102]	30	1-3, 10, 12	PZ	---	ÖDO
[103]	34	1-3, 12	PZ	---	ÖDO
[104]	22	1-4, 12	PZ	---	YE, ÖDO
[105]	3	1-3	PZ	---	YE, ÖDO
[106]	10	1-3	PZ	---	ÖDO
[107]	4	1, 2	PZ	---	---
[108]	40	34, 35	PZ	---	GÇ, KT, ÖDO
[109]	16	36, 37	PZ	---	KT
[110]	14	1-3, 10, 12	PZ	---	YE, ÖDO
[111]	22	3, 5, 10, 12-15	PZ	---	ÖDO
[112]	41	16	PZ	---	GZAS
[113]	32	1, 2, 8	PZ	---	YE
[114]	42	1-5, 9, 10, 12-15	PZ	---	YE, ÖDO
[115]	1	10	DS, MM	BA	FGZ
[116]	1	11	DS	PZ	---
[117]	25	23-26	PZ	---	TTA, GZTT
[118]	16	38	PZ	---	GÇ,YE,KT,ÖDO
[119]	6	30	PZ	---	YE, TAA
[120]	43	1, 2	PZ	---	GÇ, ÖDO
[121]	31	1, 2	PZ	---	YE, TZ, ÖDO
[122]	7	32, 33	PZ, DP	---	TAA
[123]	32	1, 2	PZ	---	YE
[124]	8	1-3, 10, 12	PZ	---	UE, ÖDO
[125]	1	19	DS	PZ	İM
[126]	21	1-5, 10, 12	PZ	---	YE,GZAS, ÖDO
[127]	9	1-4, 9, 10, 12	PZ	---	YE, ÖDO
[128]	6	30	PZ	---	YE, TAA

Çizelge 4.5. Devam

REF.	Ö.F.	A.F.	T.Ç.Y.	Y.Ç.Y.	P.Ö.K.
[129]	41	1-4, 9, 12, 13, 16	PZ	---	GZAS, ÖDO
[130]	23	1-4, 12	PZ	---	YE, ÖDO
[131]	1	2	DS, MM	GA, TB	ÇMK
[132]	24	1-5, 9, 10, 12, 13	PZ	---	YE, GZTT, ÖDO
[133]	19	3	DP	TB, TA	YE
[134]	26	1, 12, 16	DS	TB, PZ	FGZ, KI
[135]	19	3	DP	TA, TB	KI, ÖDO
[136]	6	27, 28, 30, 31	PZ	---	YE, TTA
[137]	13	29	PZ	---	TAA, YE, TZ
[138]	14	1-5, 9-12, 17	PZ	---	ÖDO, ÇÖ
[139]	35	1-4, 12	PZ	---	ÖDO
[140]	15	1, 2	PZ	---	ÖDO
[141]	15	1, 2	PZ	---	ÖDO
[142]	1	20	DS	GA	İM
[143]	9	1-3, 9, 10, 12	PZ	---	YE, ÖDO
[144]	28	1-3, 9, 10, 12	PZ	---	UE, ÖDO
[145]	29	12	---	---	KI
[146]	29	1, 3, 12	PZ	---	KI, ÖDO
[147]	38	1-4, 12	PZ	---	GZAS, ÖDO
[148]	39	1, 2	PZ	---	GÇ, YE, ÖDO
[149]	7	8	MM	KKO	FGZ
[150]	19	3	DS	---	YE, KI
[151]	42	1, 2, 8	PZ	---	YE
[152]	37	10	MM	GA, TA	YE

Çizelge 4.5-4.7’ de verilen bilgiler ışığında elde edilen bazı istatistikler Şekil 4.1-4.5’ de gösterilmiştir. En temel atölye yapısı olduğu için tek makine çizelgeleme problemi nispeten daha fazla çalışılmıştır. Öğrenme fonksiyonları genellikle her çalışmada küçük de olsa farklılık göstermektedir, yine de aynı fonksiyonu kullanan

birden fazla çalışma da mevcuttur. Amaç fonksiyonları problem yapısına göre farklılık göstermektedir. Mesela, teslim tarihi atama problemlerinde ve kaynak kısıtlı çizelgeleme problemlerinde çok kriterli benzer amaç fonksiyonları tercih edilirken, diğerlerinde tamamlanma zamanına bağlı amaç fonksiyonları daha çok tercih edilmiştir. Teslim tarihine dayalı amaç fonksiyonları da çokça kullanılmıştır. Bu çalışmalarda genellikle birden fazla amaç fonksiyonu dikkate alınmış ve özel şartlar altındaki çözüm önerileri geliştirilmiştir. Çözüm yöntemlerine bakıldığında, büyük bir kısmının polinom zamanlı çözüm yöntemiyle çözülebildiği söylenebilir ancak, özellikle teslim tarihine dayalı amaç fonksiyonları için özel bir durum dikkate alınmıştır. NP-zor yapıdaki problemler için çoğunlukla DS, ikinci olarak da MM kullanılmıştır.

Çizelge 4.6. Paralel makineli çizelgeleme problemleri

REF.	Ö.F.	A.F.	T.Ç.Y.	Y.Ç.Y.	P.Ö.K.
[107]	4	1, 2	PZ	---	---
[153]	37	18	PZ	---	YE, OT
[154]	17	1	---	TB, GA	BİS, DMH
[155]	6	39, 40	PZ	---	YE, ÖDO
[156]	1	2	DS	GA	ÇOİ
[157]	12	22	DS, MM	---	YE,BİS,BTT,ÖOM
[158]	25	2, 6, 7	PZ	---	GZTT
[159]	1	1-3, 16, 17	MM	---	FMH, Öİ
[160]	36	1	---	GA, PSO	FMH, DMH, OA
[161]	4	1, 2	PZ	---	YE
[162]	7	1	DP	---	YE
[163]	27	1	DP	PZ	YE
[164]	37	18	MM	İA	YE, OT

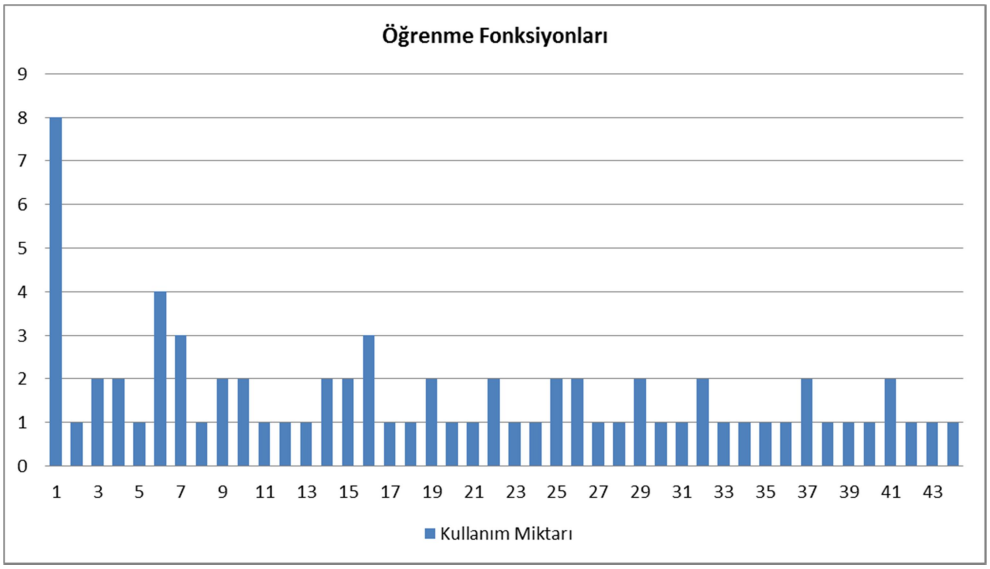
Çizelge 4.7. Akış tipi çizelgeleme problemleri

REF.	Ö.F.	A.F.	T.Ç.Y.	Y.Ç.Y.	P.Ö.K.
[101]	44	1, 2	PZ	---	ÖDO
[102]	30	1, 2	PZ	---	ÖDO
[103]	34	1, 2	PZ	---	ÖDO
[104]	22	1-4, 12	PZ	---	YE, ÖDO
[105]	3	1-3	PZ	---	YE, ÖDO
[106]	10	1-3	PZ	---	ÖDO
[165]	33	2	DS	GA	---
[166]	20	1-5, 12	---	PZ	---
[167]	10	8	DS	PZ	---
[168]	18	1-5	---	PZ	---
[169]	11	21	DS	GA	İM
[170]	27	1	---	PZ	YE, KI, ÖDO
[171]	26	1	---	TA, TB	YE, KI
[172]	2	1-3, 12	---	PZ	GÇ, ÖDO
[173]	3	1-3	PZ	---	KİM, YE, ÖDO
[174]	16	36, 37	PZ	---	Kİİ, KT, ÖDO
[175]	7	1-4, 12	---	PZ	ÖDO
[176]	5	12	DS	İA	ÖDO
[177]	1	1	DS	TB	HAÇ
[178]	1	1, 2, 7	DP	---	OAÇ, İR
[179]	1	8	DS	PZ	---

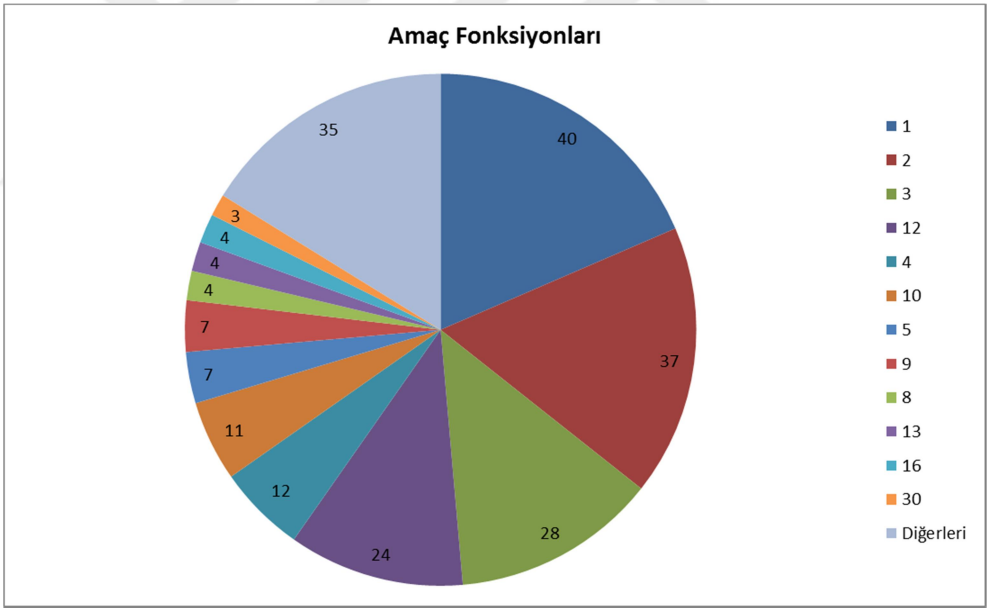
Sezgisel çözüm yöntemleri genellikle NP-zor yapıdaki problemler için geliştirilmiştir. Polinom zamanlı çözüm yöntemi bulunan çalışmalarda sezgisel yöntem düşünülmemesi normal olmakla birlikte, optimum çözümü oluşturan yöntemin kendisi hali hazırda bilinen basit sıralama kuralları olmaktadır. Sezgisel yöntem kullanımına ihtiyaç duyulan çalışmalarda çoğunlukla, basit çözüm kurucular, GA ve TB tercih edilmiştir.

Literatürde, öğrenme etkili çizelgeleme problemi yoğun olarak çalışılmıştır ve çalışılmaya da devam etmektedir. Yukarıdaki analizin dışında tutulan birçok çalışma da mevcuttur. Oldukça geniş bir alanı kapsayan bu konudaki çalışmaların tamamını bu tezde ele almak mümkün olmadığından, yakın zamanda yapılan ve göz önündeki dergilerde yayınlanmış çalışmaların bir kısmı da incelenmiştir. Bu kapsamda, [97–99, 180–184]' deki çalışmalarda öğrenme etkisi içeren tek makine çizelgeleme problemi, [99, 184–189]' daki çalışmalarda öğrenme etkisi içeren paralel makine çizelgeleme problemi ve [190–199]' deki çalışmalarda da öğrenme etkisi içeren akış tipi çizelgeleme problemi ele alınmıştır.

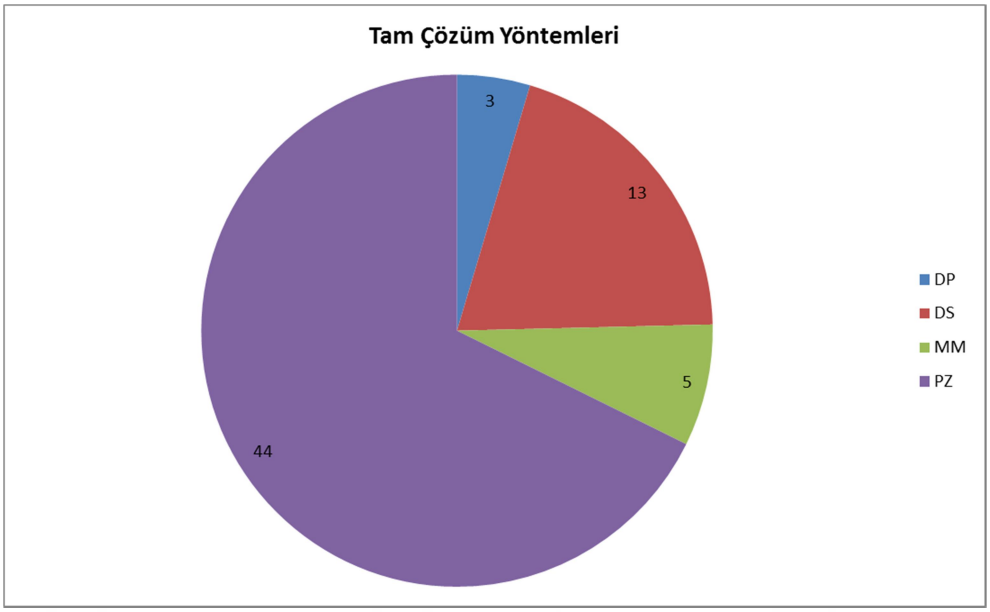
Şekil 4.1. Atölye yapısına göre çizelgeleme problemleri



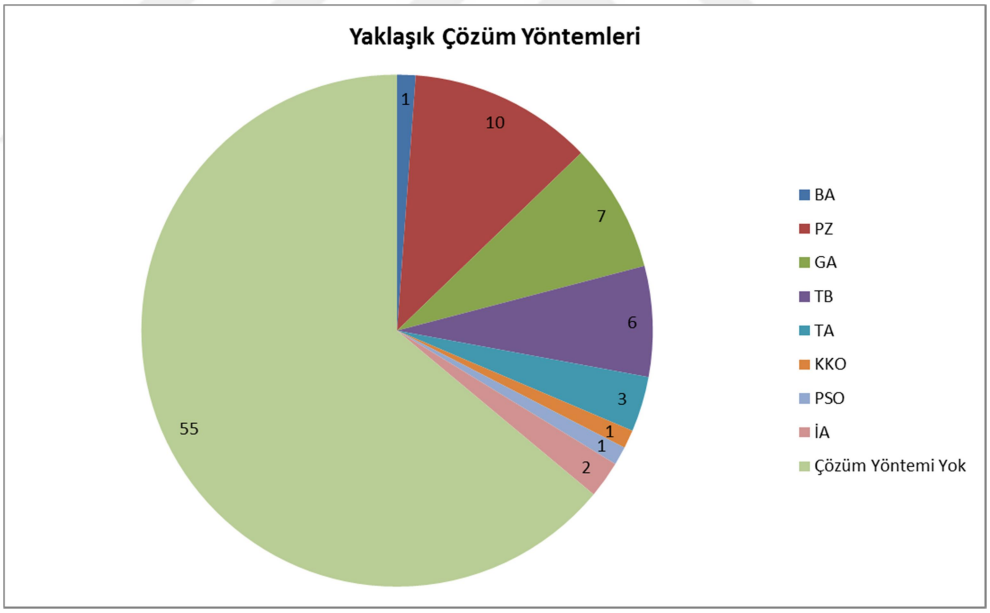
Şekil 4.2. Öğrenme fonksiyonlarının kullanım miktarı



Şekil 4.3. Amaç fonksiyonlarının kullanım miktarı



Şekil 4.4. Kesin çözüm yöntemlerinin kullanım miktarı



Şekil 4.5. Yaklaşık çözüm yöntemlerinin kullanım miktarı

4.2. Unutma Etkisi İçeren Çizelgeleme Problemleri

Yang ve Chand [200], öğrenme ve unutma etkileri altındaki iş ailelerinin toplam tamamlanma zamanını minimize etmek için tek makine çizelgeleme problemi üzerinde çalışmışlardır. Ele alınan problemde, her iş ailesine ait bir ayar süresi vardır ve ayar faaliyeti, ayar süresinden bağımsız olarak unutmaya neden olmaktadır. Pan vd. [201], duruşların unutmaya yol açtığına dayanarak iş ve iş grupları arasında ayar sürelerinin dikkate alınması gerektiğini ileri sürmüşlerdir. Yazarlar, iş gruplarının maksimum tamamlanma zamanını minimize etmek için Chiu [78] tarafından tanımlanan öğrenme-unutma modelini tek makine çizelgeleme probleminde kullanmışlardır. Liao vd. [202], unutmanın ayar süresine dayandığı bir öğrenme-unutma modeli ile iş gruplarının toplam tamamlanma zamanını ve bakım maliyetini minimize etmek için önleyici bakım işleminin olduğu bir tek makine çizelgeleme problemi üzerinde çalışmışlardır.

Li vd. [203], unutmanın ikinci makinedeki bekleme zamanına dayandığı bir öğrenme-unutma modeli ile iki makineli akış tipi çizelgeleme problemi üzerinde çalışmışlardır. İlk makinede işler arasında makine boşta zamanı oluşmadığı için unutma etkisinin gerçekleşmediğini ileri sürmüşlerdir. Li vd. [23], öğrenme ve unutma içeren bir beklemesiz akış tipi çizelgeleme problemi (bir işin sıradaki işleminin başlama zamanıyla mevcut işleminin bitiş zamanının eşit olduğu bir çizelgeleme sistemi) üzerinde çalışmışlardır. Ayar sürelerinin unutmaya yol açtığını ifade etmelerine rağmen önerdikleri öğrenme-unutma modeli ayar sürelerini içermemektedir. Lai ve Lee [144], genel bir öğrenme-unutma modeli içeren tek makine çizelgeleme problemine ait çeşitli amaç fonksiyonlarının optimum değerleri için gerekli şartları ifade etmişlerdir. Tanımlanan problemde kullanılan model, denklem 4.28' de verilmiştir. Wu vd. [124], zamana bağlı genel bir öğrenme-unutma modeli içeren tek makine çizelgeleme problemine ait çeşitli amaç fonksiyonlarının optimum değerleri için gerekli şartları ifade etmişlerdir. Tanımlanan problemde kullanılan model, denklem 4.8' de verilmiştir.

Wu vd. [204], Wu vd. [124] tarafından tanımlanan öğrenme-unutma modelinden türetilmiş zamana ve pozisyona bağlı farklı bir model önermişlerdir. Tanımladıkları

bu yeni modelle, tek makine çizelgeleme problemine ait çeşitli amaç fonksiyonlarının optimum değerleri için gerekli şartları ifade etmişlerdir. Marichelvam vd. [199], gerçek bir sistemde gözlemlenen özelliklere dayanarak, çalışanların kabiliyetlerini ve yaşlarını yansıtan bir öğrenme-unutma modelini hibrid akış tipi çizelgeleme problemine uygulamışlardır. Dikkate alınan tüm etkiler, yaygın olan fonksiyonlardan ziyade sabit birer katsayı olarak belirlenmiştir.

4.3. Ayar Süresi İçeren Çizelgeleme Problemleri

Ayar süreleriyle ilgili yapılan araştırma $1|s_{ij}|\sum_j w_j T_j$ ve $1|s_{ij}|\sum_j T_j$ problemleriyle sınırlı tutulmuştur. Ayar süresi içeren çizelgeleme problemi oldukça fazla olmasına rağmen [9, 205, 206], ilgili gecikme problemlerinin kesin çözümü için yapılan sınırlı sayıda çalışma bulunmaktadır. Picard ve Queyranne [207], $1|s_{ij}|\sum_j w_j T_j$ problemini zamana bağlı gezgin satıcı problemi olarak modellemiş ve bu modelin yumuşatmasıyla elde ettikleri alt sınırı kullanarak etkili bir DS geliştirmişlerdir. Ragatz [208], $1|s_{ij}|\sum_j T_j$ problemi için tespit ettiği baskınlık kuralları ve alt sınırı kullanarak DS geliştirmiştir. Rubin ve Ragatz [13], $1|s_{ij}|\sum_j T_j$ problemi için bir GA geliştirmiş ve [208]' deki DS ile elde edilen optimum sonuçlarla kıyaslamışlardır.

Luo vd. [209], $1|s_{ij}|\sum_j T_j$ problemi için tespit ettiği yeni baskınlık kuralları ve alt sınırı kullanarak DS geliştirmişlerdir. Luo ve Chu [210], zamana bağlı gezgin satıcı probleminden ilham alan yeni alt sınır ile [209]' daki çalışmanın devamı niteliğinde yeni bir DS önermişlerdir. Yine $1|s_{ij}|\sum_j T_j$ problemi için Bigras vd. [211], zamana bağlı gezgin satıcı probleminin tam sayılı matematiksel modelinden elde ettikleri alt sınırı kullanarak bir DS geliştirmişlerdir. Son olarak, $1|s_{ij}|\sum_j w_j T_j$ problemi için DP ve DS içeren en etkili kesin çözüm yöntemi Tanaka ve Araki [11] tarafından geliştirilmiştir. Bahsedilen problemler için sezgisel yöntemler, kesin çözüm yöntemlerine nazaran daha fazla kullanılmıştır. Kim vd. [212], $1|s_{ij}|\sum_j w_j T_j$ problemi için sıralama kurallarını ve sinir ağlarını koordine eden hibrid bir yöntem geliştirmişlerdir. Lee vd. [213], 'görünen gecikme maliyeti' (apparent tardiness cost - ATC) [214, 215] kuralına hazırlık zamanlarının adapte edilmesiyle genişletilen

'hazırlık zamanlarının olduğu görünen gecikme maliyeti' (apparent tardiness cost with setups - ATCS) kuralını geliştirmişlerdir ve bu yeni yöntemin [216]' da önerilenden daha başarılı olduğunu göstermişlerdir.

Bahsi geçen problemler için TB [217], GA [218–220], KKO [221], İA [222, 223], TA [224] ve dağınık arama (scatter search - DA) [225] gibi çeşitli sezgisel yöntemler de kullanılmıştır. Bunların yanında, performansıyla dikkat çeken sezgisel yöntemler; Liao ve Juan [226] tarafından geliştirilen KKO, Lin ve Ying [227] tarafından geliştirilen TB, GA ve TA, Anghinolfi ve Paolucci [228, 229] tarafından geliştirilen KKO ve PSO, Tasgetiren vd. [230] tarafından geliştirilen kesikli kademeli evrim algoritması (discrete differential evolution algorithm – KKE), Ying vd. [231] tarafından geliştirilen İA, Kirlik ve Oguz [232] tarafından geliştirilen DKA, Guo ve Tang [233] tarafından geliştirilen DA, Xu vd. [234] tarafından geliştirilen İA, Xu vd. [235] tarafından geliştirilen KKE, González ve Vela [236] tarafından geliştirilen GA ve son olarak Subramanian ve Farias [237] tarafından geliştirilen İA olmuştur.

4.4. Değerlendirme

Birçok araştırmacı, öğrenme etkisi içeren çizelgeleme problemlerine yoğunlaşmış ve çeşitli çözüm önerileri geliştirmişlerdir [16]. Bu konuyla ilgili yakın zamanda yapılan literatür araştırmasından [18] anlaşıldığı üzere NP-zor yapıdaki $1|L|\sum_j T_j$ ve $1|L|\sum_j w_j T_j$ problemlerini dikkate alan çalışma nispeten daha az miktardadır. Eren ve Güner [238], $1|LP|\sum_j T_j$ problemini ilk kez tanımlamış ve MM, TB, TA gibi çeşitli çözüm yöntemleri geliştirmişlerdir. Yin vd. [116], $1|LP|\sum_j w_j T_j$ problemi için bir takım baskınlık kurallarıyla desteklediği DS geliştirmişlerdir. Yin vd. [115], $1|r_j, LP|\sum_j T_j$ probleminin kesin çözümleri için DS, yaklaşık çözümleri için BA geliştirmişlerdir. Wu vd. [239], $1|LP|\sum_j \min\{T_j, p_j r^a\}$ probleminin kesin çözümleri için DS, yaklaşık çözümleri için GA geliştirmişlerdir. Azadeh vd. [152], zamana bağlı monoton olmayan işlem sürelerinin, öğrenme ve yaşlanma etkilerinin olduğu $1|LT+D|\sum_j T_j$ probleminin kesin çözümleri için MM, yaklaşık çözümleri için GA ve TA' nın olduğu hibrid bir yöntem geliştirmişlerdir. Görüldüğü üzere, öğrenme

etkisinin dikkate alındığı ve NP-zor yapıda olan toplam gecikme problemleri henüz tek makine problemlerinde dahi oldukça sınırlı sayıdadır. Bunun yanında, atölye ortamı ve amaç fonksiyonu bakımından çeşitli çizelgeleme problemlerinde öğrenme etkisinin ve BLAS' nin beraber dikkate alındığı az sayıda çalışma da gözlemlenmiştir.

Toksarı ve Güner [240, 241], $P|LT+D, s_{ij}|\sum_j\left(k_1E_j+k_2T_j\right)$ problemi için bir TDOP önermişlerdir. Yue vd. [242], işlerin gruplara ayrıldığı ve gruplar arasında BLAS olan $1|LT, s_{ij}|C_{max}, \sum_j T_j$ pareto optimal çizelgeleme problemi için KKO geliştirmişlerdir. Shahvari ve Logendran [190], gruplara ayrılmış işlerin partiler halinde işlem gördüğü ve partiler arasında BLAS olan hibrid akış tipi atölyede $F|LP, r_j, s_{ij}|\sum_j w_j\left(k_1C_j+k_2T_j\right)$ problemi için MM, İA, TA ve PSO geliştirmişlerdir. Mousavi vd. [191], işlem süreleri gibi ayar süreleri üzerinde de öğrenme etkisi olan, tekrarlı ve hibrid yapıdaki akış tipi atölyede $F|LP, s_{ij}|C_{max}, \sum_j T_j$ pareto optimal çizelgeleme problemi için GA geliştirmişlerdir. Azadeh vd. [243], işlem süreleri stokastik olan esnek yapıdaki akış tipi atölyede $F|LT+D, s_{ij}|\sum_j T_j$ problemi için GA ve yapay sinir ağlarının entegre edildiği bir simülasyon modeli geliştirmişlerdir. Expósito-Izquierdo vd. [188], sadece ayar süreleri üzerinde öğrenme etkisi olan $P|LP+D, s_{ij}|\sum_j C_j$ problemi için DKA içeren bir simülasyon modeli geliştirmişlerdir. Bunların yanında, beklemesiz $F|LP+F, s_{ij}|\sum_j C_j$ problemi [23] için İA, işlerin gruplara ayrıldığı ve önleyici bakım içeren $1|LP+F, s_{ij}, s_{pq}|C_{max}$ problemi [201] için GA ve yine iş grupları ve önleyici bakım içeren $1|LP+F+D, s_{ij}, s_{pq}|\sum_j C_j$ problemi [202] için GA önerilen, öğrenme etkisi yanında unutma etkisi de içeren çalışmalar bulunmaktadır.

Bu bölümdeki değerlendirme çerçevesinde; öğrenme etkisi ve BLAS içeren çizelgeleme problemleri genellikle çok makineli atölye ortamlarında çalışılmıştır. Anlatılan çalışmalardaki tanımlanan problemler NP-zor yapıdadır ve birçoğunda kesin çözüm yöntemi geliştirilmemiştir. Sadece iki çalışmada MM oluşturulmuştur ancak, önerilen modellere dayalı deneysel çalışmalar oldukça sınırlı boyuttaki problemler için yapılmıştır.

5. PROBLEM TANIMI

Bu bölümde, ele alınan 2 farklı problemin matematiksel tanımı, varsayımları ve simgesel gösterimleri ifade edilmiştir. İlk problem olarak, tek makinede toplam gecikmenin minimizasyonu hedeflenmiştir. Bu problemde, işlem süreleri üzerinde pozisyona bağlı öğrenme etkisi ve işler arasında BLAS bulunmaktadır. Böylece, tanımlanan ilk problem [29]' de belirtilen üçlü gösterime göre $1|LP, s_{ij}|\sum_j T_j$ ile ifade edilmektedir.

İkinci problemde ise, tek makinede maksimum tamamlanma zamanının minimizasyonu hedeflenmiştir. İlk problemden farklı olarak, işlem süreleri üzerinde zamana bağlı öğrenme etkisi ve BZAS' ne bağlı unutma etkisi bulunmaktadır. Böylece, tanımlanan ikinci problem [29]' de belirtilen üçlü gösterime göre $1\left|LT + F, s_j\right|C_{max}$ ile ifade edilmektedir. Ele alınan problemler ortak bir ifadeyle şu şekilde tanımlanabilir:

Belirli miktarda bağımsız ve eş zamanlı hazır olan işler bulunmaktadır. İşler arasında herhangi bir öncelik ilişkisinin olmadığı ve işlere ait işlem sürelerinin bölünemeyeceği varsayılmıştır. Herhangi bir iş için işleme başlamadan önce yapılması gereken bir takım ayar faaliyetleri vardır ve bu faaliyetler tanımlanan ayar süresi kadar işlem görülecek makineyi meşgul etmektedir.

Tüm işler, herhangi bir anda sadece bir işe ait işlemi ve ayar faaliyetlerini yapabilen, bu zamanlar dışında sürekli müsait olan bir makinede işlem görecektir. Tanımlanan bu sistemde, istenen performans ölçütünü optimize edecek şekilde işlerin tümünün yapılabilmesi için bir çizelgeye ihtiyaç duyulmaktadır. Her iki problemde de ortak olan unsurların simgesel gösterimi aşağıda verilmiştir.

N	işler kümesi, $N = \{J_1, J_2, \cdots, J_n\}$.
n	işlerin sayısı.
π	işlerin oluşturduğu bir çizelge $\pi = \{\pi(1), \pi(2), \cdots, \pi(n)\}$.

$[k]$	bir çizelgenin k. pozisyonunda bulunan işi ifade eder. Örneğin, $[k] = \pi(k) = j$.
p_j	bir $j \in N$ işinin normal işlem süresi.
s_j	bir $j \in N$ işinin işlemden önceki ayar süresi.

İlerleyen alt bölümlerde, her iki problem için detaylı ifadeler verilmiştir.

5.1. $1|LP, s_{ij}|\sum_j T_j$ Problemi

Bu problemde, işlem süreleri üzerinde pozisyona bağlı öğrenme etkisi ve işler arasında BLAS vardır. Çizelgeye göre işlem sırası gelen bir iş için gereken ayar faaliyetleri, çizelgede işlemi tamamlanan son işe göre değişmektedir. İlaveten, bir iş çizelgenin ilk pozisyonunda olsa dahi bu işe başlamadan önce bir takım ayar faaliyetleri gerekmektedir. Bu nedenle, bu problemdeki ayar süreleri iki indisle belirtilmiştir.

s_{0j}	bir $j \in N$ işinin, ilk iş olarak işleme alınacağı durumda gereken ayar süresi.
s_{ij}	bir $j \in N$ işinin, $i \in N$ işinden hemen sonra işleme alınacağı durumda gereken ayar süresi.
d_j	bir $j \in N$ işinin teslim tarihi.
C_j^π	bir π çizelgesindeki $j \in N$ işinin tamamlanma zamanı.
T_j^π	bir π çizelgesindeki $j \in N$ işinin gecikmesi.
T^π	bir π çizelgesinin toplam gecikmesi.

Eğer j işi $\pi = \{\pi(1), \pi(2), \cdots, \pi(r), \cdots, \pi(n)\}$ çizelgesinin r. pozisyonunda işlem görüyorsa, bu işin gerçekleşen işlem süresi

$$p_{\pi(r)}^A = p_{[r]}^A = p_j^A = p_j r^a \tag{5.1}$$

olmaktadır. Burada üst olarak belirtilen a öğrenme endeksidir ve $a \leq 0$. İlk

pozisyondaki iş için belirli bir ayar süresi olduğundan dolayı, hesaplamaları kolaylaştırmak ve bütünlük sağlamak için $p_0 = 0$ ve $d_0 = 0$ olan bir sanal iş J_0 tanımlanmıştır. Sanal iş her zaman çizelgenin 0. pozisyonunda bulunmaktadır ve böylece $\pi(0) = [0] = J_0$ olduğu varsayılmıştır. Tanımlanan bu sistemde, toplam gecikmeyi minimize edecek şekilde işlerin tümünün yapılabilmesini sağlayan olurlu bir π çizelgesi aranacaktır. Bu çizelgenin r. pozisyonunda işlem gören bir j işinin tamamlanma zamanı ve gecikmesi sırasıyla denklem 5.2 ve 5.3 ile, çizelgenin toplam gecikmesi ise denklem 5.4 ile hesaplanmaktadır.

$$C_j^{\pi} = \sum_{k=1}^{r} s_{[k-1][k]} + p_{[k]}k^a \tag{5.2}$$

$$T_j^{\pi} = \max\{C_j^{\pi} - d_j, 0\} \tag{5.3}$$

$$T^{\pi} = \sum_{j=1}^{n} T_j^{\pi} \tag{5.4}$$

Çizelge 5.1' de tanımlanan probleme ait bir örneğin ayar süreleri, Çizelge 5.2' de ilgili örneğin işlem süreleri ve teslim tarihleri verilmiştir. Söz konusu örnek, Bölüm 7.1' de belirlenen parametrelere göre türetilmiş bir soru setine aittir. Bu soru seti, iş sayısı $n = 25$, teslim tarihi sıkılık faktörü $\tau = 0.3$, teslim tarihi aralık faktörü $R = 0.25$ ve ayar süresi şiddet faktörü $\eta = 0.25$ parametre değerleriyle elde edilmiştir.

Çizelge 5.3' de de ilgili örneğin 3 farklı yaklaşık çözümü ve optimum çözümü verilmiştir. Söz konusu örneğin çözümleri $L = 0.7$ öğrenme etkisi faktörü için tespit edilmiştir. Yaklaşık çözümlerden ilki Smith kuralı [2] olarak da bilinen işlem sürelerinin artan sıralamasına göre (shortest processing time - SPT), ikincisi Jackson kuralı [244] olarak da bilinen teslim tarihlerinin artan sıralamasına göre (earliest due date - EDD), üçüncüsü ise ATCS sıralamasına göre oluşturulmuştur. Örneğin bir de optimum çözümü verilmiştir. Optimum çözüm Bölüm 6.1.1' de anlatılan DS ile elde edilmiştir.

Çizelge 5.1. $1|LP, s_{ij}|\sum_j T_j$ problem örneğinin ayar süreleri (s_{ij})

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**
0	18	41	5	29	0	18	41	35	3	21	5	46	6	22	10	10	31	50	50	42	13	16	45	33	8
1	0	3	21	29	43	6	30	17	29	12	5	48	25	15	32	23	18	33	40	4	2	23	43	9	24
2	21	0	34	25	34	26	18	24	3	26	31	33	46	15	33	27	49	48	18	48	0	43	34	26	10
3	35	42	0	15	24	9	1	3	26	28	21	12	49	35	13	34	45	16	27	1	0	6	27	49	48
4	17	38	48	0	21	34	41	39	50	9	10	10	35	23	2	32	37	34	15	21	17	42	3	19	40
5	34	8	24	27	0	24	46	27	5	28	11	15	47	24	28	30	1	45	42	47	8	15	11	4	7
6	26	11	34	15	9	0	28	22	39	31	24	6	47	20	48	45	43	41	50	45	28	17	36	46	31
7	47	25	6	32	48	7	0	47	7	44	46	28	20	21	1	4	12	30	37	20	26	29	28	30	25
8	35	0	18	45	14	25	42	0	48	8	13	10	18	14	17	36	8	16	7	28	24	7	10	21	39
9	11	38	6	27	35	45	9	18	0	20	0	45	6	20	15	1	29	24	30	9	28	7	47	31	15
10	20	12	0	33	24	22	4	15	35	0	28	23	16	5	35	33	43	32	32	2	14	35	27	35	7
11	36	26	26	44	36	49	26	27	38	50	0	0	41	50	25	12	27	35	39	14	8	43	28	10	29
12	12	20	0	36	6	50	7	2	26	3	7	0	50	19	0	5	15	19	1	26	41	7	45	20	2

Çizelge 5.1. Devam

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**
13	24	14	31	27	32	42	43	3	32	18	9	38	0	31	26	10	4	4	16	23	38	50	17	0	35
14	41	37	27	43	3	50	27	3	1	7	17	48	10	0	49	39	28	49	32	39	33	39	44	17	10
15	47	17	50	44	25	50	4	7	8	20	38	40	24	42	0	49	8	38	34	27	23	24	13	11	14
16	15	11	5	24	26	31	8	37	27	30	47	37	1	25	22	0	5	29	12	19	15	28	40	3	34
17	11	9	32	12	40	15	17	1	28	15	28	42	7	5	38	41	0	27	17	7	9	2	36	8	28
18	2	39	35	45	3	24	4	37	25	18	34	48	3	47	37	23	16	0	29	22	41	4	9	11	13
19	12	11	20	41	2	38	17	0	41	37	28	1	9	14	33	15	6	29	0	41	39	49	25	45	12
20	26	41	6	40	34	35	40	44	18	5	42	13	44	9	4	20	45	28	10	0	11	20	30	49	6
21	37	1	19	47	36	38	5	3	2	45	25	14	14	5	30	43	41	22	25	24	0	14	1	15	50
22	38	30	37	6	32	49	14	11	40	1	43	45	4	13	50	11	33	8	25	26	1	0	21	31	13
23	1	8	29	37	38	16	19	28	20	41	43	13	15	43	29	45	47	42	7	36	38	21	0	4	22
24	47	24	2	49	5	43	46	19	45	25	48	38	6	48	19	1	10	4	32	20	0	7	31	0	25
25	9	8	32	35	16	8	22	16	18	45	41	21	3	18	0	2	48	22	17	12	26	11	38	20	0

Çizelge 5.2. $1|LP, s_{ij}|\sum_j T_j$ problem örneğinin işlem süreleri (p_j) ve teslim tarihleri (d_j)

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	p_{16}	p_{17}	p_{18}	p_{19}	p_{20}	p_{21}	p_{22}	p_{23}	p_{24}	p_{25}
58	69	71	71	74	76	79	82	83	86	89	92	100	105	108	109	111	115	124	127	129	136	140	146	150
d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}	d_{14}	d_{15}	d_{16}	d_{17}	d_{18}	d_{19}	d_{20}	d_{21}	d_{22}	d_{23}	d_{24}	d_{25}
728	700	645	757	712	752	765	582	722	744	746	642	529	732	737	739	765	731	710	734	656	740	764	548	567

Çizelge 5.3. $1|LP, s_{ij}|\sum_j T_j$ problem örneğinin SPT, EDD, ATCS ve optimum çizelgeleri (π)

$[r]$	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	T^{π}
π_{SPT}	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	5495.5
π_{EDD}	13	24	25	8	12	3	21	2	19	5	9	1	18	14	20	15	16	22	10	11	6	4	23	7	17	2169.2
π_{ATCS}	5	17	8	2	21	9	11	12	3	7	15	24	16	13	18	1	6	4	23	19	14	10	20	25	22	785.2
π_{OPT}	3	8	2	9	11	12	25	16	13	24	18	1	21	14	5	17	22	10	20	15	7	6	4	23	19	406.5

5.2. $1|LT + F, s_j|C_{max}$ Problemi

Bu problemde, işlem süreleri üzerinde hem öğrenme hem de unutma etkisi vardır. Herhangi bir iş için uygulanan öğrenme etkisi, o işten önce yapılan işlerin normal işlem süreleri toplamına göre değişmektedir. Aynı iş için uygulanan unutma etkisi ise, sadece o işin ayar süresine bağlıdır. Bir iş için gereken ayar faaliyetleri, önceki işten ve çizelgedeki pozisyonundan bağımsızdır, dolayısıyla BZAS kullanılmıştır. Bu problemde kullanılan öğrenme ve unutma etkileri, Wu vd.' nin [124] tanımladığı modelin bir uzantısıdır. İlgili çalışmadaki model denklem 5.5' de olduğu gibi formülüze edilmiştir.

$$p_j^A = \begin{cases} p_j[1 - F(x)], & \text{eğer} \quad x = \sum_{k=1}^{r-1} p_{[k]} \leq k_0 \\ p_j[1 - F(x) + G(x - k_0)], & \text{eğer} \quad x = \sum_{k=1}^{r-1} p_{[k]} > k_0 \end{cases} \tag{5.5}$$

Denklem 5.5' deki modelde p_j^A ve p_j ifadeleri, çizelgenin r. pozisyonunda işlem gören j işinin gerçekleşen ve normal işlem süreleridir. Fonksiyon olarak tanımlanan F ve G ifadeleri sırasıyla öğrenme ve unutma fonksiyonlarıdır. Her iki fonksiyonda bağımsız değişken olarak kullanılan x ifadesi, çizelgenin r. pozisyonundan önce işlem gören işlerin normal işlem sürelerinin toplamıdır. Son olarak, k_0 ile unutma etkisinin gözlemlenmeye başladığı eşik değeri ifade edilmiştir. Wu vd.' nin [124] ele aldığı bu modelde, iş değişimlerine bağlı oluşan unutma etkisi dikkate alınmıştır ve tek değişkenli bir modeldir. Değişken sayısına göre analiz edilen öğrenme ve unutma modelleri hakkında detaylı bilgiler için [41]' a bakılabilir.

Wu vd.' nin [124] çalışmasında tanımlanan problemden farklı olarak, $1|LT + F, s_j|C_{max}$ probleminde ayar süreleri ve ayar sürelerine bağlı unutma etkisi bulunmaktadır. Ayar süreleri üretim periyodu içindeki duruşlar olduğu için, kullanılan unutma etkisi bu duruşların uzunluğuna ve duruştan hemen önceki öğrenme şiddetine bağlıdır. Dolayısıyla, ilgili problem için iki değişkenli bir model kullanılmıştır. Değişkenlerden biri öğrenme etkisini belirlerken, diğeri unutma etkisini belirlemektedir. Daha önce tanımlanan ortak unsurlar dışında bu problemde kullanılacak diğer unsurlar aşağıda verilmiştir.

x_0	unutmanın şiddetini düzenlemek için kullanılan normalizasyon değeri, $x_0 \geq 0$.
k_0	unutmanın başladığı zamanı ifade eden eşik değeri, $k_0 \geq 0$.
x	öğrenmenin şiddetini belirleyen toplam işlem zamanı, $x \geq 0$.
y	unutmanın şiddetini belirleyen duruş zamanı, $y \geq 0$.
$f(x)$	öğrenme fonksiyonu, $f: [0,\infty) \rightarrow [0,1)$ aralıklarında tanımlı artan fonksiyon, $f'(x) > 0$. Ayrıca, $f''(x) < 0$ varsayılmıştır.
$\mathrm{g}(y)$	unutma fonksiyonu, $\mathrm{g}: [0,\infty) \rightarrow [0,1)$ aralıklarında tanımlı artan fonksiyon, $\mathrm{g}'(y) > 0$. Ayrıca, $\mathrm{g}''(y) < 0$ varsayılmıştır.
$h(x,y)$	bağıl fonksiyon, $h(x,y) = f(x) - f(x_0)\mathrm{g}(y)$.
C_{max}^{π}	π çizelgesinin maksimum tamamlanma zamanı.

Yukardaki tanımlamalar ışığında, eğer j işi bir çizelgenin r. pozisyonunda işlem görüyorsa ($r = 1,2,\cdots,n$), bu işin gerçekleşen işlem süresi

$$p_j^A = p_{[r]}^A = \begin{cases} p_j[1 - f(x)], & \text{eğer } x \leq k_0 \\ p_j[1 - f(x) + f(x_0)\mathrm{g}(y)], & \text{eğer } x > k_0 \end{cases} \tag{5.6}$$

olmaktadır. Burada, öğrenme ve unutma şiddetlerini belirleyen bağımsız değişkenler sırasıyla $x = \sum_{k=1}^{r-1} p_{[k]}$ ve $y = s_j$ ile belirlenir. Normalizasyon değeri $x_0 \leq \min_{j \in N}\{p_j\}$ olarak kabul edilmiştir. Böylece, bağıl fonksiyon değerinin $h(x,y) \geq 0$ olması ve dolayısıyla gerçekleşen işlem süresinin $p_j^A \leq p_j$ olması sağlanmıştır. Bu durumun matematiksel ifadesi denklem 5.7' de verilmiştir. Bunun neticesinde, bağıl fonksiyon $h: [0,\infty) \times [0,\infty) \rightarrow [0,1)$ aralıklarında tanımlı ve negatif değer üretmeyen bir fonksiyondur.

$$f(x) - f(x_0)\mathrm{g}(y) \geq f(x) - f(x)\mathrm{g}(y) = f(x)[1 - \mathrm{g}(y)] \geq 0 \tag{5.7}$$

Tanımlanan problemde, tüm işleri içeren bir çizelge $\pi = \{j | j \in N\}$, iki alt çizelge $\pi = \{\pi', \pi''\}$ olarak ifade edilebilir. Bunlardan biri $\pi' = \{j | j \in N', N' \neq \emptyset\}$ ve bir diğeri $\pi'' = \{j | j \in N''\}$ alt çizelgeleridir. Burada $N' \cap N'' = \emptyset$ ve $N' \cup N'' = N$ olduğuna dikkat edilmelidir. Bu alt çizelgelerden π'' için unutma etkisi geçerli iken,

π' için geçerli değildir. Alt çizelge $\pi' = \{\hat{\pi}, j\}$ olarak ifade edildiğinde, denklem 5.6' dan yola çıkılarak denklem 5.8 ve 5.9 elde edilir.

$$k_0 \geq \sum_{i \in \hat{\pi}} p_i = \hat{P} \tag{5.8}$$

$$k_0 < \hat{P} + p_j = P' \tag{5.9}$$

Bu problemdeki performans ölçütü, maksimum tamamlanma zamanının (son işin tamamlanma zamanı) minimizasyonudur ve denklem 5.10 ile hesaplanmaktadır.

$$C_{max}^{\pi} = \sum_{j \in \pi} s_j + \sum_{j=1}^{|\pi'|} p_{[j]} \left[1 - f\left(\sum_{i=1}^{j-1} p_{[i]}\right)\right] + \sum_{j=|\pi'|+1}^{n} p_{[j]} \left[1 - h\left(\sum_{i=1}^{j-1} p_{[i]}, s_{[j]}\right)\right] \tag{5.10}$$

Çizelge 5.4' de tanımlanan probleme ait bir örneğin işlem süreleri ve ayar süreleri verilmiştir. Söz konusu örnek, Bölüm 7.2' de belirlenen parametrelere göre türetilmiş bir soru setine aittir. Bu soru seti, iş sayısı $n = 25$, olası en büyük işlem süresi $\tilde{p}_{\max} = 100$ ve eşik değeri oranı $\sigma = 0.25$ parametre değerleriyle elde edilmiştir. Bu eşik değeri oranına göre $k_0 = 277$ olmaktadır. Öğrenme fonksiyonu $f(x) = 1 - (1 + x)^a$, unutma fonksiyonu $g(y) = 1 - (1 + y)^a$ ve normalizasyon değeri $x_0 = \min_{j \in N}\{p_j\}$ olarak belirlenmiştir ($a = -0.152$).

Çizelge 5.5' de de ilgili örneğin SPT sıralaması ve optimum çizelgesi verilmiştir. Örneğin optimum çözümü hem Bölüm 6.2.3' de anlatılan TDOP ile hem de Bölüm 6.2.4' de anlatılan DP ile elde edilmiştir.

Çizelge 5.4. $1|LT + F, s_j|C_{max}$ problem örneğinin işlem süreleri (p_j) ve ayar süreleri (s_j)

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	p_{16}	p_{17}	p_{18}	p_{19}	p_{20}	p_{21}	p_{22}	p_{23}	p_{24}	p_{25}
64	74	2	35	43	74	67	76	68	27	1	1	6	20	43	32	30	11	88	42	92	78	44	37	53
s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}	s_{11}	s_{12}	s_{13}	s_{14}	s_{15}	s_{16}	s_{17}	s_{18}	s_{19}	s_{20}	s_{21}	s_{22}	s_{23}	s_{24}	s_{25}
51	28	10	66	3	78	7	14	56	12	30	46	93	63	85	64	43	8	65	84	70	33	93	87	59

Çizelge 5.5. $1|LT + F, s_j|C_{max}$ problem örneğinin SPT ve optimum çizelgeleri (π)

$[r]$	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	C_{max}^{π}
π_{SPT}	12	11	3	13	18	14	10	17	16	4	24	20	15	5	23	25	1	7	9	6	2	8	22	19	21	1738.2
π_{OPT}	12	11	3	13	18	14	16	4	24	20	15	23	21	10	17	5	25	1	7	9	6	2	8	22	19	1734.3

6. ÇÖZÜM YÖNTEMLERİ

Bir çizelgeleme problemi çözüm özelliği açısından genellikle bir optimizasyon problemidir. Optimizasyon, belirlenen performans ölçütünün çeşitli kısıtlar altında minimizasyonu veya maksimizasyonudur [245] ve bu yapıda olan tüm problemler optimizasyon problemidir. Optimizasyon problemi için geliştirilen çözüm yöntemi kendine özgüdür ve problemin yapısına göre değişiklik gösterir. Problemin yapısı ise, performans ölçütüne ve kısıtlara göre belirlenir [246]. Bir minimizasyon problemi, denklem 6.1 ile tanımlanabilir. Eğer çözüm uzayı S sınırlı ancak oldukça büyük bir küme ise, bir kombinatoryal optimizasyon problemi, eğer $S = \mathbb{R}^n$ ise, bir sürekli optimizasyon problemi tanımlanmıştır.

$$\min\{\gamma(\pi) \mid \pi \in D \subseteq S\} \tag{6.1}$$

Çizelgeleme problemlerinin yapısal özelliklerinden biri de, kombinatoryal optimizasyon problemi olmalarıdır. Kombinatoryal optimizasyon, çözüm uzayının kesikli çözümlerle (örn: tam sayı) sınırlandırılmış alt uzayındaki optimum çözümü bulmakla ilgilidir [247]. Kombinatoryal problemlerde olurlu bir çözümün matematiksel gösterimi sade ve özlü olmasına rağmen, olurlu çözümlerin sayısı oldukça büyük olmaktadır ve genellikle matematiksel gösterimin boyutuna göre üstel olarak artmaktadır. Örneğin, şebeke olarak ifade edilen problemin boyutu (örn: düğüm sayısı) n ise, çözüm uzayının boyutu (çözüm sayısı) 2^n olabilmektedir veya tam sayılardan oluşan tek boyutlu bir dizi olarak ifade edilen bir problemin boyutu n ise, çözüm uzayının boyutu $n!$ olabilmektedir. Bu nedenle, böyle bir problemin kesin çözümünü bulmak için uzaydaki tüm çözümleri tek tek denemek uygulanabilir bir yöntem değildir ve daha etkili çözüm yöntemleri geliştirmek gerekmektedir. Etkili çözüm yöntemi, kavramsal olarak ilk Edmonds [248] tarafından ifade edilmiştir ve günümüzde de kabul görmektedir. Bu kavrama göre, bir yönteme (kesin çözüm yöntemi) etkili denilebilmesi için, çözüm süresinin üst sınırı problem boyutun polinomuyla ifade edilebilmelidir. Başka bir deyişle, polinom zamanlı çözüm yöntemleri etkili yöntemlerdir. Bu nedenle, çözüm yöntemine karar vermeden önce ilgili problemin karmaşıklığı belirlenmelidir.

6.1. $1|LP, s_{ij}|\sum_j T_j$ Problemine Ait Çözüm Yöntemleri

Bilindiği üzere, $1||\sum_j T_j$ probleminin NP-zor (normal şiddette) olduğu ıspatlanmıştır [249]. Dolayısıyla, işlem süreleri üzerinde pozisyona bağlı öğrenme etkisi olan $1|LP|\sum_j T_j$ [238] ve BLAS olan $1|s_{ij}|\sum_j T_j$ [13] problemleri de ıspata ihtiyaç duyulmadan NP-zor (şiddeti belirtilmemiş) olarak kabul edilmiştir. Çünkü öğrenme etkisi ve BLAS' nin her ikisi de problemin zorluk derecesini azaltıcı bir etki göstermemiştir. Ayrıca, $1|s_{ij}|\sum_j T_j$ problemi zamana bağlı gezgin satıcı problemi olarak da ifade edilebildiği için karmaşıklığı tartışma konusu edilmemiştir [207]. Bu tezde ele alınan $1|LP, s_{ij}|\sum_j T_j$ problemi, işlem süreleri üzerinde pozisyona bağlı öğrenme etkisiyle birlikte BLAS içerdiğinden dolayı NP-zor problemdir ve kesin çözümü tespit etmek için PZ uygulamak mümkün değildir. Bu nedenle, $1|LP, s_{ij}|\sum_j T_j$ problemi için kesin çözüm yöntemi olarak DS, yaklaşık çözüm yöntemi olarak da bazı operatörler bakımından farklılık gösteren iki GA ve iki DKA geliştirilmiştir.

6.1.1. Dal-Sınır Algoritması

DS' nın çoğunlukla kullanılan yapısı, işlerin birer birer dallanma yapan düğümlere sistematik bir şekilde atanmasıdır. Bölünme, sınırlama, dallanma ve budama işlemleri DS' nın dört temel prosedürünü oluşturur [116, 208–210]. Bu tezde de aynı yapı kullanılmıştır. Öncelikle, standart bir düğüm yapısı tanımlanmıştır. Bir düğüm, algoritmanın problemsiz bir şekilde çalışması için gerekli olan bilgiyi saklayacak yapıda olmalıdır. Tanımlanan probleme ait herhangi bir örneğin çözümünde, ihtiyaç duyulacak düğüm sayısı önceden bilinmediği için, düğümleri saklamak üzere standart diziler veya benzeri yapılar DS' da kullanılamamaktadır. Bu nedenle, ihtiyaç duyuldukça kullanıma alınacak ve gerektiğinde serbest bırakılacak dinamik hafıza yapıları tercih edilmelidir.

Bu tezde geliştirilen DS' da, düğümler için tanımlanan yapısal değişkenleri saklamak ve etkili şekilde kullanmak üzere bağlı liste (linked list) olarak adlandırılan zincir

yapısı kullanılmıştır. Zincir yapısında, bir düğümden bir başka düğüme hareket tek yönlüdür. Tüm zincirler içerdiği düğümler bakımından atanan iş sayısına göre oluşturulmuştur. Herhangi bir zincire ait olan düğümler 'aktif' olarak nitelendirilir. Kullanılan zincir sayısı $n-1$' dir ve erişim için her bir zincirin ilk düğümünün adresini saklayan $n-1$ boyutlu bir işaretçi dizisi tanımlanmıştır. Örneğin işaretçi dizisinin list$[k]$ $(k = 1,2,\cdots,n-1)$ olduğu varsayılırsa, list$[5]$, çizelgesinin 5. pozisyonuna iş ataması bekleyen düğümlerin oluşturduğu zincirin ilk düğümüne ait adresi saklar. Algoritmanın çalışma süresince oluşacak tüm düğümler aşağıdaki değişkenleri içeren bir yapıda tanımlanmıştır.

$\circledast$	sonraki komşu düğümün adresini tanımlayan işaretçi.
$\ddot{\pi}$	düğümde saklanan n boyutlu çizelge.
k	$\ddot{\pi}$ çizelgesindeki iş ataması bekleyen pozisyon.
$C^{\ddot{\pi}}_{[k-1]}$	$\ddot{\pi}$ çizelgesindeki atanan son işin tamamlanma zamanı.
$\sum_{l=1}^{k-1} T^{\ddot{\pi}}_{[l]}$	$\ddot{\pi}$ çizelgesindeki atanan işlerin toplam gecikmesi.
$T^{\ddot{\pi}}$	$\ddot{\pi}$ çizelgesinin toplam gecikmesi için alt sınır değeri.

Bu tanımlamalara göre, aktif bir düğümde $\ddot{\pi} = \{J_1, J_2, \cdots, J_n\}$ ve $k \geq 1$ dolayısıyla, $C^{\ddot{\pi}}_{[k-1]} \geq 0$, $\sum_{l=1}^{k-1} T^{\ddot{\pi}}_{[l]} \geq 0$ ve $T^{\ddot{\pi}} \geq 0$ olduğu anlaşılmaktadır. DS kök düğümle çalışmaya başlar ve sonlanana kadar bölünme, sınırlama, dallanma ve budama işlemlerini tekrarlar. Kök düğümde $\ddot{\pi} = \{J_1, J_2, \cdots, J_n\}$ ve $k = 1$ dolayısıyla, $C^{\ddot{\pi}}_{[0]} = 0$, $\sum_{l=1}^{0} T^{\ddot{\pi}}_{[l]} = 0$ ve $T^{\ddot{\pi}} = T^{\pi}$ (denklem 6.47 ve 6.48) olur.

Tanımlanan $1|LP, s_{ij}|\sum_j T_j$ probleminde olurlu bir çözüm, yani çizelge işlerin permütasyonlu gösterimiyle ifade edilmiştir. Permütasyonlu gösterim çizelgeleme problemlerinde yaygın olarak kullanılmaktadır. Bu gösterimle elde edilebilecek olurlu çözümlerin sayısı $n!$ olduğundan dolayı, özellikle problem girdisinin alacağı büyük değerler için optimum çözümü tespit etmek oldukça zorlaşmaktadır. Bundan dolayı DS' da kullanılmak üzere, optimum çözümü diğer çözümlerden ayıran birtakım baskınlık kuralları ve polinom zamanda hesaplanabilen bir alt sınır geliştirilmiştir.

6.1.1.1. Baskınlık Kuralları

Tanımlanan probleme ait $\pi_1 = \{\pi_s, a_l, i, j, b_f, \pi_e\}$ ve $\pi_2 = \{\pi_s, a_l, j, i, b_f, \pi_e\}$ çizelgeleri arasındaki tek fark i ve j komşu işlerin pozisyonlarıdır. Söz konusu i ve j işlerinin π_1 çizelgesindeki pozisyonları r_1 ve r_2 iken, π_2 çizelgesindeki pozisyonları bunun tersidir. Her iki çizelgenin aynı pozisyonlarında bulunan a_l işi r_1 pozisyonundan önce işlem gören alt çizelgenin $\{\pi_s, a_l\}$ son işini temsil ederken, b_f işi r_2 pozisyonundan sonra işlem gören alt çizelgenin $\{b_f, \pi_e\}$ ilk işini temsil etmektedir. Her iki çizelgede de $\{\pi_s, a_l\}$ alt çizelgesinin sıralaması ve başlangıç zamanı aynı olduğundan $C_{a_l}^{\pi_1} = C_{a_l}^{\pi_2}$ ve dolayısıyla $\sum_{k\in\{\pi_s,a_l\}} T_k^{\pi_1} = \sum_{k\in\{\pi_s,a_l\}} T_k^{\pi_2}$ olur. Bu durumda, π_1 çizelgesinin π_2 çizelgesine baskın olduğunu söylemek için $C_{b_f}^{\pi_1} \leq C_{b_f}^{\pi_2}$ ve $T_i^{\pi_1} + T_j^{\pi_1} \leq T_i^{\pi_2} + T_j^{\pi_2}$ eşitsizliklerinin sağlandığını göstermek yeterlidir.

Önerme 6.1. Denklem 6.2-6.5 ile belirtilen şartlar sağlanıyorsa, π_1 çizelgesi π_2 çizelgesine baskındır.

$$s_{a_l i} - s_{a_l j} + s_{ij} - s_{ji} + s_{jb_f} - s_{ib_f} + (p_i - p_j)({r_1}^a - {r_2}^a) \leq 0 \tag{6.2}$$

$$s_{a_l i} - s_{a_l j} + s_{ij} - s_{ji} + d_i - d_j + (p_i - p_j)({r_1}^a - {r_2}^a) \leq 0 \tag{6.3}$$

$$s_{a_l i} - s_{a_l j} - s_{ji} + p_i({r_1}^a - {r_2}^a) - p_j {r_1}^a \leq 0 \tag{6.4}$$

$$2s_{a_l i} - 2s_{a_l j} + s_{ij} - s_{ji} + (p_i - p_j)(2{r_1}^a - {r_2}^a) \leq 0 \tag{6.5}$$

Ispat. Baskınlığın şartlarından biri olan $C_{b_f}^{\pi_1} \leq C_{b_f}^{\pi_2}$ eşitsizliğinde ifade edilen tamamlanma zamanları denklem 6.6 ve 6.7 ile verilmiştir.

$$C_{b_f}^{\pi_1} = C_{a_l}^{\pi_1} + s_{a_l i} + p_i {r_1}^a + s_{ij} + p_j {r_2}^a + s_{jb_f} + p_{b_f}(r_2 + 1)^a \tag{6.6}$$

$$C_{b_f}^{\pi_2} = C_{a_l}^{\pi_2} + s_{a_l j} + p_j {r_1}^a + s_{ji} + p_i {r_2}^a + s_{ib_f} + p_{b_f}(r_2 + 1)^a \tag{6.7}$$

Bu tamamlanma zamanlarına göre $C_{b_f}^{\pi_1} - C_{b_f}^{\pi_2} \leq 0$ olur ve dolayısıyla denklem 6.2 ile belirtilen şart sağlanmış olur. Baskınlığın bir diğer şartı olan $T_i^{\pi_1} + T_j^{\pi_1} \leq T_i^{\pi_2} + T_j^{\pi_2}$ eşitsizliğinin sağlanması aşağıdaki 4 farklı durumla açıklanmıştır.

Durum 1: $T_i^{\pi_1} = T_j^{\pi_1} = 0$.

Gecikme negatif değer alamayacağı için $0 \leq T_i^{\pi_2} + T_j^{\pi_2}$ olur.

Durum 2: $T_i^{\pi_1} = 0$ ve $T_j^{\pi_1} > 0$.

Bu durumda $T_j^{\pi_1} \leq T_i^{\pi_2} + T_j^{\pi_2}$ eşitsizliği sağlanmalıdır ve $T_j^{\pi_2} = 0$ varsayımıyla bu eşitsizliğin en hassas şartı dikkate alınmış olur [208]. Böylece, geriye $T_j^{\pi_1} - T_i^{\pi_2} \leq 0$ eşitsizliği kalır.

$$T_j^{\pi_1} = C_{a_l}^{\pi_1} + s_{a_l i} + p_i {r_1}^a + s_{ij} + p_j {r_2}^a - d_j > 0 \tag{6.8}$$

$$T_i^{\pi_2} = C_{a_l}^{\pi_2} + s_{a_l j} + p_j {r_1}^a + s_{ji} + p_i {r_2}^a - d_i > 0 \tag{6.9}$$

Denklem 6.8 ve 6.9' dan yola çıkarak $T_j^{\pi_1} - T_i^{\pi_2} \leq 0$ eşitsizliği çözüldüğünde, denklem 6.3 ile belirtilen şart sağlanmış olur.

Durum 3: $T_i^{\pi_1} > 0$ ve $T_j^{\pi_1} = 0$.

Durum 2' ye benzer şekilde $T_j^{\pi_2} = 0$ varsayımıyla $T_i^{\pi_1} \leq T_i^{\pi_2} + T_j^{\pi_2}$ eşitsizliğinin en hassas şartı dikkate alınmış olur. Böylece, geriye $T_i^{\pi_1} \leq T_i^{\pi_2}$ dolayısıyla $C_i^{\pi_1} - C_i^{\pi_2} \leq 0$ eşitsizliği kalır.

$$C_i^{\pi_1} = C_{a_l}^{\pi_1} + s_{a_l i} + p_i {r_1}^a \tag{6.10}$$

$$C_i^{\pi_2} = C_{a_l}^{\pi_2} + s_{a_l j} + p_j {r_1}^a + s_{ji} + p_i {r_2}^a \tag{6.11}$$

Denklem 6.10 ve 6.11' den yola çıkarak $C_i^{\pi_1} - C_i^{\pi_2} \leq 0$ eşitsizliği çözüldüğünde, denklem 6.4 ile belirtilen şart sağlanmış olur.

Durum 4: $T_i^{\pi_1} > 0$ ve $T_j^{\pi_1} > 0$.

Bu durumda $T_i^{\pi_1} + T_j^{\pi_1} \leq T_i^{\pi_2} + T_j^{\pi_2}$ dolayısıyla $C_i^{\pi_1} + C_j^{\pi_1} - C_i^{\pi_2} - C_j^{\pi_2} \leq 0$ eşitsizliğinin sağlanması yeterlidir.

$$C_j^{\pi_1} = C_{a_l}^{\pi_1} + s_{a_l i} + p_i {r_1}^a + s_{ij} + p_j {r_2}^a \tag{6.12}$$

$$C_j^{\pi_2} = C_{a_l}^{\pi_2} + s_{a_l j} + p_j {r_1}^a \tag{6.13}$$

Denklem 6.10-6.13' den yola çıkarak $C_i^{\pi_1} + C_j^{\pi_1} - C_i^{\pi_2} - C_j^{\pi_2} \leq 0$ eşitsizliği çözüldüğünde, denklem 6.5 ile belirtilen şart sağlanmış olur. Sonuç olarak, önerme 6.1' de denklem 6.2-6.5 ile belirtilen şartların tümü sağlandığında π_1 çizelgesi π_2 çizelgesine baskındır ve böylece ıspat tamamlanmıştır. ■

Bir diğer baskınlık kuralında komşu olmayan işlerin yer değişimi ele alınmıştır. Yani, yer değişimi yapan işler arasında en az bir iş daha bulunmaktadır. Tanımlanan probleme ait $\pi_1 = \{\pi_s, a_l, i, b_f, \pi_m, b_l, j, c_f, \pi_e\}$ çizelgesinde i ve j işlerinin pozisyonları r_1 ve r_2 iken, $\pi_2 = \{\pi_s, a_l, j, b_f, \pi_m, b_l, i, c_f, \pi_e\}$ çizelgesinde bunun tersidir. Her iki çizelgenin aynı pozisyonlarında bulunan a_l işi r_1 pozisyonundan önce işlem gören alt çizelgenin $\{\pi_s, a_l\}$ son işini, b_f ve b_l işleri r_1 ve r_2 pozisyonları arasında işlem gören alt çizelgenin $\{b_f, \pi_m, b_l\}$ ilk ve son işlerini, c_f işi ise r_2 pozisyonundan sonra işlem gören alt çizelgenin $\{c_f, \pi_e\}$ ilk işini temsil etmektedir. Her iki çizelgede $\{\pi_s, a_l\}$ alt çizelgesinin sıralaması ve başlangıç zamanı aynı olduğundan $C_{a_l}^{\pi_1} = C_{a_l}^{\pi_2}$ ve dolayısıyla $\sum_{k\in\{\pi_s,a_l\}} T_k^{\pi_1} = \sum_{k\in\{\pi_s,a_l\}} T_k^{\pi_2}$ olur. Bu durumda, π_1 çizelgesinin π_2 çizelgesine baskın olduğunu söylemek için $C_{b_f}^{\pi_1} \leq C_{b_f}^{\pi_2}$, $C_{c_f}^{\pi_1} \leq C_{c_f}^{\pi_2}$ ve $T_i^{\pi_1} + T_j^{\pi_1} \leq T_i^{\pi_2} + T_j^{\pi_2}$ eşitsizliklerinin sağlandığını göstermek yeterlidir.

Önerme 6.2. Denklem 6.14-6.17 ile belirtilen şartlar sağlanıyorsa, π_1 çizelgesi π_2 çizelgesine baskındır.

$$s_{a_l i} - s_{a_l j} + s_{ib_f} - s_{jb_f} + (p_i - p_j){r_1}^a \leq 0 \tag{6.14}$$

$$s_{a_l i} - s_{a_l j} + s_{ib_f} - s_{jb_f} + s_{b_l j} - s_{b_l i} + s_{jc_f} - s_{ic_f} + (p_i - p_j)({r_1}^a - {r_2}^a) \leq 0 \tag{6.15}$$

$$s_{a_l i} - s_{a_l j} + s_{ib_f} - s_{jb_f} + s_{b_l j} - s_{b_l i} + \big(p_i - p_j\big)({r_1}^a - {r_2}^a) + d_i - d_j \leq 0 \tag{6.16}$$

$$2s_{a_l i} - 2s_{a_l j} + s_{ib_f} - s_{jb_f} + s_{b_l j} - s_{b_l i} + (p_i - p_j)(2{r_1}^a - {r_2}^a) \leq 0 \tag{6.17}$$

Ispat. Baskınlığın şartlarından biri olan $C_{b_f}^{\pi_1} \leq C_{b_f}^{\pi_2}$ eşitsizliğinde ifade edilen tamamlanma zamanları denklem 6.18 ve 6.19 ile verilmiştir.

$$C_{b_f}^{\pi_1} = C_{a_l}^{\pi_1} + s_{a_l i} + p_i {r_1}^a + s_{ib_f} + p_{b_f}(r_1 + 1)^a \tag{6.18}$$

$$C_{b_f}^{\pi_2} = C_{a_l}^{\pi_2} + s_{a_l j} + p_j {r_1}^a + s_{jb_f} + p_{b_f}(r_1 + 1)^a \tag{6.19}$$

Bu tamamlanma zamanlarına göre $C_{b_f}^{\pi_1} - C_{b_f}^{\pi_2} \leq 0$ olur ve dolayısıyla denklem 6.14 ile belirtilen şart sağlanmış olur. Yine, baskınlığın şartlarından biri olan $C_{c_f}^{\pi_1} \leq C_{c_f}^{\pi_2}$ eşitsizliğindeki tamamlanma zamanları denklem 6.20 ve 6.21 ile verilmiştir.

$$\begin{aligned} C_{c_f}^{\pi_1} \quad = \quad & C_{a_l}^{\pi_1} + s_{a_l i} + p_i {r_1}^a + s_{ib_f} + p_{b_f}(r_1 + 1)^a + \theta + s_{b_l j} \\ & + p_j {r_2}^a + s_{jc_f} + p_{c_f}(r_2 + 1)^a \end{aligned} \tag{6.20}$$

$$\begin{aligned} C_{c_f}^{\pi_2} \quad = \quad & C_{a_l}^{\pi_2} + s_{a_l j} + p_j {r_1}^a + s_{jb_f} + p_{b_f}(r_1 + 1)^a + \theta + s_{b_l i} \\ & + p_i {r_2}^a + s_{ic_f} + p_{c_f}(r_2 + 1)^a \end{aligned} \tag{6.21}$$

$$\theta = s_{b_f\pi_m(1)} + p_{\pi_m(1)}(r_1+2)^a + s_{\pi_m(1)\pi_m(2)} + p_{\pi_m(2)}(r_1+3)^a + \cdots + p_{b_l}(r_2-1)^a \quad (6.22)$$

Bu tamamlanma zamanlarına göre $C_{c_f}^{\pi_1} - C_{c_f}^{\pi_2} \leq 0$ olur ve dolayısıyla denklem 6.15 ile belirtilen şart sağlanmış olur. Baskınlığın bir diğer şartı olan $T_i^{\pi_1} + T_j^{\pi_1} \leq T_i^{\pi_2} + T_j^{\pi_2}$ eşitsizliğinin sağlanması önerme 6.1' deki durumlarla açıklanmıştır.

Durum 1: $T_i^{\pi_1} = T_j^{\pi_1} = 0$ (Bkz. Önerme 6.1).

Durum 2: $T_i^{\pi_1} = 0$ ve $T_j^{\pi_1} > 0$.

Bu durumda $T_j^{\pi_1} - T_i^{\pi_2} \leq 0$ eşitsizliği değerlendirilmelidir (Bkz. Önerme 6.1).

$$T_j^{\pi_1} = C_{a_l}^{\pi_1} + s_{a_l i} + p_i {r_1}^a + s_{ib_f} + p_{b_f}(r_1+1)^a + \theta + s_{b_l j} + p_j {r_2}^a - d_j \quad (6.23)$$

$$T_i^{\pi_2} = C_{a_l}^{\pi_2} + s_{a_l j} + p_j {r_1}^a + s_{jb_f} + p_{b_f}(r_1+1)^a + \theta + s_{b_l i} + p_i {r_2}^a - d_i \quad (6.24)$$

Denklem 6.23 ve 6.24' den yola çıkarak $T_j^{\pi_1} - T_i^{\pi_2} \leq 0$ eşitsizliği çözüldüğünde, denklem 6.16 ile belirtilen şart sağlanmış olur.

Durum 3: $T_i^{\pi_1} > 0$ ve $T_j^{\pi_1} = 0$.

Bu durumda $C_i^{\pi_1} - C_i^{\pi_2} \leq 0$ eşitsizliği değerlendirilmelidir (Bkz. Önerme 6.1). Denklem 6.14 ile belirtilen şart sağlandığında $C_{b_f}^{\pi_1} \leq C_{b_f}^{\pi_2}$ olduğu için $C_i^{\pi_1} < C_{b_f}^{\pi_1} \leq C_{b_f}^{\pi_2} < C_i^{\pi_2}$ ve dolayısıyla $C_i^{\pi_1} - C_i^{\pi_2} \leq 0$ olur.

Durum 4: $T_i^{\pi_1} > 0$ ve $T_j^{\pi_1} > 0$.

Bu durumda $C_i^{\pi_1} + C_j^{\pi_1} - C_i^{\pi_2} - C_j^{\pi_2} \leq 0$ değerlendirilmelidir (Bkz. Önerme 6.1).

$$C_i^{\pi_1} = C_{a_l}^{\pi_1} + s_{a_l i} + p_i {r_1}^a \tag{6.25}$$

$$C_i^{\pi_2} = C_{a_l}^{\pi_2} + s_{a_l j} + p_j {r_1}^a + s_{jb_f} + p_{b_f}(r_1 + 1)^a + \theta + s_{b_l i} + p_i {r_2}^a \tag{6.26}$$

$$C_j^{\pi_1} = C_{a_l}^{\pi_1} + s_{a_l i} + p_i {r_1}^a + s_{ib_f} + p_{b_f}(r_1 + 1)^a + \theta + s_{b_l j} + p_j {r_2}^a \tag{6.27}$$

$$C_j^{\pi_2} = C_{a_l}^{\pi_2} + s_{a_l j} + p_j {r_1}^a \tag{6.28}$$

Denklem 6.25-6.28' den yola çıkarak $C_i^{\pi_1} + C_j^{\pi_1} - C_i^{\pi_2} - C_j^{\pi_2} \leq 0$ eşitsizliği çözüldüğünde, denklem 6.17 ile belirtilen şart sağlanmış olur. Sonuç olarak, önerme 6.2' de denklem 6.14-6.17 ile belirtilen şartların tümü sağlandığında π_1 çizelgesi π_2 çizelgesine baskındır ve böylece ıspat tamamlanmıştır. ■

6.1.1.2. Alt Sınır

Geliştirilen alt sınır, tanımlanan problem girdilerinin özel bir durumuna ait çözümü temsil etmektedir. Problem girdileri (p_j, d_j ve s_{ij}) belirtilen bir takım şartları sağladığında, bu girdilerin çeşitli sıralamasıyla oluşan bir çizelge ile optimum sonuç kolayca bulunabilmektedir. BLAS yumuşatılarak BZAS' ne dönüştürüldüğünde, tanımlanan $1|LP, s_{ij}|\sum_j T_j$ problemi $1|LP, s_j|\sum_j T_j$ problemi olur. Bahsi geçen dönüşümle (denklem 6.29) bir işe başlamadan önceki ayar sürelerinin en küçüğü tespit edilip, o işin standart ayar süresi olarak belirlenir.

$$s_j = \min\left\{\min_{i \in N, i \neq j}\{s_{ij}\}, s_{0j}\right\} \qquad \forall j \in N \tag{6.29}$$

Daha sonra, tüm s_j, d_j ve p_j değerlerinin artan sıralamalarını temsil eden N_s, N_d ve

N_p iş setleri kullanılarak, denklem 6.30' da görüldüğü gibi π çizelgesi oluşturulur.

$$s_{\pi(l)} = s_{N_s(l)},\ d_{\pi(l)} = d_{N_d(l)} \text{ ve } p_{\pi(l)} = p_{N_p(l)} \qquad \forall l \in \{1,2,\cdots,n\} \qquad (6.30)$$

Bunun neticesinde, π çizelgesinde $1 \leq k \leq l \leq n$ için $s_{\pi(k)} \leq s_{\pi(l)}$, $d_{\pi(k)} \leq d_{\pi(l)}$ ve $p_{\pi(k)} \leq p_{\pi(l)}$ ilişkisi söz konusudur.

Önerme 6.3' den yola çıkarak, elde edilen bu çizelgenin toplam gecikmesi optimaldir ve tanımlanan $1|LP, s_{ij}|\sum_j T_j$ probleminin alt sınırıdır [208].

Önerme 6.3. Tüm i ve j iş çiftleri için $d_i \leq d_j$, $s_i \leq s_j$ ve $p_i \leq p_j$ eşitsizlikleri sağlanıyorsa, $1|LP, s_j|\sum_j T_j$ problemi için EDD sıralaması optimumdur.

Ispat. Önermenin ıspatı için, komşu iki işin yer değişimi incelenmiştir. Tanımlanan probleme ait $\pi_1 = \{\pi_s, i, j, \pi_e\}$ çizelgesinde i ve j işlerinin pozisyonları r_1 ve r_2 iken, $\pi_2 = \{\pi_s, j, i, \pi_e\}$ çizelgesinde bunun tersidir. Bu çizelgelerden π_1 EDD kuralına uyarken, π_2 bu kuralı ihlal etmektedir. Her iki çizelgede π_s alt çizelgesinin sıralaması ve başlangıç zamanı aynı olduğundan $C^{\pi_1}_{[r_1-1]} = C^{\pi_2}_{[r_1-1]}$ ve dolayısıyla $\sum_{k\in\pi_s} T_k^{\pi_1} = \sum_{k\in\pi_s} T_k^{\pi_2}$ olur. Bu durumda, π_1 çizelgesinin optimalliğini doğrulamak için $C_j^{\pi_1} \leq C_i^{\pi_2}$ ve $T_i^{\pi_1} + T_j^{\pi_1} \leq T_i^{\pi_2} + T_j^{\pi_2}$ olduğunu göstermek yeterlidir.

$$C_j^{\pi_1} = C^{\pi_1}_{[r_1-1]} + s_i + p_i {r_1}^a + s_j + p_j {r_2}^a \qquad (6.31)$$

$$C_i^{\pi_2} = C^{\pi_2}_{[r_1-1]} + s_j + p_j {r_1}^a + s_i + p_i {r_2}^a \qquad (6.32)$$

İşlem süreleri $p_i \leq p_j$ ve öğrenme endeksi $a \leq 0$ olduğu için, denklem 6.31 ve 6.32' den yola çıkarak $C_j^{\pi_1} - C_i^{\pi_2} = ({r_1}^a - {r_2}^a)(p_i - p_j) \leq 0$ elde edilir. Böylece, optimalliğin şartlarından $C_j^{\pi_1} \leq C_i^{\pi_2}$ doğrulanmış olur. Optimalliğin diğer şartı olan $T_i^{\pi_1} + T_j^{\pi_1} \leq T_i^{\pi_2} + T_j^{\pi_2}$ eşitsizliğinin sağlanması aşağıdaki 4 farklı durumla açıklanmıştır ve ilgili gecikmeler denklem 6.33-6.36 ile verilmiştir.

$$T_i^{\pi_1} = \max\left\{C_{[r_1-1]}^{\pi_1} + s_i + p_i {r_1}^a - d_i, 0\right\} \geq 0 \tag{6.33}$$

$$T_j^{\pi_1} = \max\left\{C_{[r_1-1]}^{\pi_1} + s_i + p_i {r_1}^a + s_j + p_j {r_2}^a - d_j, 0\right\} \geq 0 \tag{6.34}$$

$$T_i^{\pi_2} = \max\left\{C_{[r_1-1]}^{\pi_2} + s_j + p_j {r_1}^a + s_i + p_i {r_2}^a - d_i, 0\right\} \geq 0 \tag{6.35}$$

$$T_j^{\pi_2} = \max\left\{C_{[r_1-1]}^{\pi_2} + s_j + p_j {r_1}^a - d_j, 0\right\} \geq 0 \tag{6.36}$$

Durum 1: $T_i^{\pi_1} = T_j^{\pi_1} = 0$ (Bkz. Önerme 6.1).

Durum 2: $T_i^{\pi_1} = 0$ ve $T_j^{\pi_1} > 0$.

Bu gecikmelere göre $C_i^{\pi_1} \leq d_i \leq d_j < C_j^{\pi_1} \leq C_i^{\pi_2}$ eşitsizliği söz konusudur. Dolayısıyla, $T_i^{\pi_2} > 0$ ve $T_j^{\pi_2} = \max\{C_j^{\pi_2} - d_j, 0\}$ olmalıdır. Bu durumda $d_i \leq d_j$, $p_i \leq p_j$ ve $a \leq 0$ olduğu için, $T_j^{\pi_1} - T_i^{\pi_2} - T_j^{\pi_2} = (p_i - p_j)({r_1}^a - {r_2}^a) + d_i - d_j - \max\{C_j^{\pi_2} - d_j, 0\} \leq 0$ doğrulanmış olur.

Durum 3: $T_i^{\pi_1} > 0$ ve $T_j^{\pi_1} = 0$.

Bu gecikmelere göre $d_i < C_i^{\pi_1} < C_j^{\pi_1} \leq d_j$ ve $d_i < C_i^{\pi_1} < C_j^{\pi_1} \leq C_i^{\pi_2}$ eşitsizlikleri söz konusudur. Dolayısıyla, $T_i^{\pi_2} > 0$ ve $T_j^{\pi_2} = \max\{C_j^{\pi_2} - d_j, 0\}$ olmalıdır. Bu durumda $C_i^{\pi_1} < C_i^{\pi_2}$ olduğu için, $T_i^{\pi_1} - T_i^{\pi_2} - T_j^{\pi_2} = C_i^{\pi_1} - C_i^{\pi_2} - \max\{C_j^{\pi_2} - d_j, 0\} \leq 0$ doğrulanmış olur.

Durum 4: $T_i^{\pi_1} > 0$ ve $T_j^{\pi_1} > 0$.

Bu gecikmelere göre $d_i < C_i^{\pi_1} < C_j^{\pi_1} \leq C_i^{\pi_2}$ ve $d_i \leq d_j < C_j^{\pi_1} \leq C_i^{\pi_2}$ eşitsizlikleri söz konusudur. Dolayısıyla, $T_i^{\pi_2} > 0$ ve $T_j^{\pi_2} = \max\{C_j^{\pi_2} - d_j, 0\}$ olmalıdır. Bu durumda $C_j^{\pi_2} \leq d_j$ ve $C_j^{\pi_2} > d_j$ ihtimalleri ayrı olarak ele alınmalıdır.

Eğer $C_j^{\pi_2} \leq d_j$ ise $T_j^{\pi_2} = 0$ ve $d_i < C_i^{\pi_1} \leq C_j^{\pi_2} \leq d_j < C_j^{\pi_1} \leq C_i^{\pi_2}$ olduğu için, $T_i^{\pi_1} + T_j^{\pi_1} - T_i^{\pi_2} = C_i^{\pi_1} - d_j - \left(C_i^{\pi_2} - C_j^{\pi_1}\right) \leq 0$ doğrulanmış olur. Aksi halde, $C_j^{\pi_2} > d_j$ ise $T_j^{\pi_2} = C_j^{\pi_2} - d_j > 0$ ve $C_i^{\pi_1} \leq C_j^{\pi_2} < C_j^{\pi_1} \leq C_i^{\pi_2}$ olduğu için, $T_i^{\pi_1} + T_j^{\pi_1} - T_i^{\pi_2} - T_j^{\pi_2} = C_i^{\pi_1} - C_j^{\pi_2} + C_j^{\pi_1} - C_i^{\pi_2} \leq 0$ doğrulanmış olur.

Sonuç olarak, belirtilen tüm şartlar altında π_1 çizelgesinin optimalliği doğrulanmış ve böylece ıspat tamamlanmıştır. ■

6.1.1.3. Bölünme

Bu aşamada, başlangıç dışında dallanma ile belirlenen aktif bir düğümden yeni düğümler türetilir. Bölünme için seçilen $k \leq n-1$ seviyesindeki aktif bir düğümden, en fazla $n-k+1$ adet yeni düğüm meydana gelebilir ve bu yeni düğümlere halef düğüm denir. İlgili aktif düğümdeki $\ddot{\pi}$ çizelgesinin $(k, k+1, \cdots, n)$. pozisyonundaki işler, sırayla k. pozisyona atanarak yeni bir halef düğüm oluşturulur. İş atamasına bağlı olarak, halef düğüme ait k değeri ve $\ddot{\pi}$ çizelgesi belirlenmiş olur. Halef düğüme ait bu bilgiler, baskınlık testlerinin yapılması için yeterlidir. Her halef düğüme, zincire eklenmeden önce, önerme 6.1 ve 6.2' nin her biri için iki kez baskınlık testi yapılır. Her testte ele alınan işler farklı olduğu için ilk testin uygulaması daha kolay ve hızlı yapılabilmektedir. İlk olarak, önerme 6.1' e göre baskınlık testinin ilki yapılır. Bu testte dikkate alınan işler aşağıda belirtildiği gibidir.

$$a_l = \ddot{\pi}(k-4), j = \ddot{\pi}(k-3), i = \ddot{\pi}(k-2) \text{ ve } b_f = \ddot{\pi}(k-1) \tag{6.37}$$

Görüldüğü üzere, bu testin yapılabilmesi için $4 \leq k < n$ olmalıdır ve önerme 6.1' deki simgelerle ifade edilirse, denklem 6.37' deki iş sıralaması π_2 çizelgesini göstermektedir. Eğer belirlenen işler için önerme 6.1 sağlanırsa, bu halef düğüme baskın olan bir başka düğüm olduğu anlaşılır (baskın olan düğümdeki iş sıralaması $i = \ddot{\pi}(k-3)$ ve $j = \ddot{\pi}(k-2)$ dışında denklem 6.37' de belirtilenle aynıdır) ve bu halef düğüm budanarak bir sonraki halef düğümü oluşturma işlemleri başlar. Aksi halde, önerme 6.1' e göre baskınlık testinin ikincisi yapılır. Bu testte dikkate alınan

işler aşağıda belirtildiği gibidir.

$$\forall b_f \in \{\ddot{\pi}(l) \mid l = k, k+1, \cdots, n\} \text{ için}$$
$$\{a_l = \ddot{\pi}(k-3),\ j = \ddot{\pi}(k-2),\ i = \ddot{\pi}(k-1)\} \tag{6.38}$$

Görüldüğü üzere, bu testin yapılabilmesi için $3 \leq k < n$ olmalıdır ve ilk testin işlem karmaşıklığı $O(1)$ iken bu testin işlem karmaşıklığı $O(n-k+1)$' dir. Eğer denklem 6.38' de belirlenen işler için önerme 6.1 sağlanırsa, bu halef düğüme baskın olan bir başka düğüm olduğu anlaşılır (baskın olan düğümdeki iş sıralaması $i = \ddot{\pi}(k-2)$ ve $j = \ddot{\pi}(k-1)$ dışında denklem 6.38' de belirtilenle aynıdır) ve bu halef düğüm budanarak bir sonraki halef düğümü oluşturma işlemleri başlar. Aksi halde, sınırlama aşamasına geçilir.

Bu tezde sınırlama aşaması, bölünme aşaması kapsamında çalışan bir alt prosedür olarak tasarlanmıştır. Dolayısıyla, bir düğüm için sınırlama işlemleri tamamlandığında, bölünme işlemleri devam eder. Bu aşamada, $k = n$ olan bir halef düğüm için $T^{\ddot{\pi}}$ değeri üst sınırdan (olurlu bir çizelgeye ait bilinen en küçük toplam gecikme değeri) küçük ise bu düğümün $T^{\ddot{\pi}}$ değeri üst sınır olarak güncellenir. Aksi halde, mevcut üst sınır değiştirilmez. Böyle bir düğümden elde edilebilecek halef düğüm kalmadığından dolayı, bu halef düğüm budanarak bir sonraki halef düğümü oluşturma işlemleri başlar.

Halef düğüm için $k < n$ durumunda ise, denklem 6.46' da belirtilen alt sınır değeri mevcut üst sınır değeriyle kıyaslanır. Eğer $T^{\ddot{\pi}}$ değeri üst sınırdan büyük veya üst sınıra eşit ise, bu halef düğüm budanarak bir sonraki halef düğümü oluşturma işlemleri başlar. Aksi halde, algoritma bölünme aşamasında bir sonraki işlemle devam eder. Sınırlama aşamasından geçen halef düğüm için önerme 6.2' ye göre baskınlık testinin ilki yapılır. Bu testte dikkate alınan işler aşağıda belirtildiği gibidir.

$$\exists r\colon 0 \leq r < k-4 \text{ için}$$
$$\{a_l = \ddot{\pi}(r),\ j = \ddot{\pi}(r+1),\ b_f = \ddot{\pi}(r+2),$$
$$i = \ddot{\pi}(k-2),\ c_f = \ddot{\pi}(k-1)\} \tag{6.39}$$

Görüldüğü üzere, bu testin yapılabilmesi için $5 \leq k < n$ olmalıdır ve önerme 6.2' deki simgelerle ifade edilirse, denklem 6.39' daki iş sıralaması π_2 çizelgesini göstermektedir. Eğer belirlenen işler için önerme 6.2 sağlanırsa, bu halef düğüme baskın olan bir başka düğüm olduğu anlaşılır (baskın olan düğümdeki iş sıralaması $i = \ddot{\pi}(r+1)$ ve $j = \ddot{\pi}(k-2)$ dışında denklem 6.39' da belirtilenle aynıdır) ve bu halef düğüm budanarak bir sonraki halef düğümü oluşturma işlemleri başlar. Aksi halde, önerme 6.2' ye göre baskınlık testinin ikincisi yapılır. Bu testte dikkate alınan işler aşağıda belirtildiği gibidir.

$$\begin{aligned}
&\exists r\colon 0 \leq r < k-3 \text{ için} \\
&\quad \{\forall c_f \in \{\ddot{\pi}(l) \mid l = k, k+1, \cdots, n\} \text{ için} \\
&\qquad \{a_l = \ddot{\pi}(r), j = \ddot{\pi}(r+1), b_f = \ddot{\pi}(r+2), i = \ddot{\pi}(k-1)\}\}
\end{aligned} \tag{6.40}$$

Görüldüğü üzere, bu testin yapılabilmesi için $4 \leq k < n$ olmalıdır ve ilk testin işlem karmaşıklığı $O(k-5)$ iken bu testin işlem karmaşıklığı $O\big((k-4)(n-k+1)\big)$' dir. Eğer denklem 6.40' da belirlenen işler için önerme 6.2 sağlanırsa, bu halef düğüme baskın olan bir başka düğüm olduğu anlaşılır (baskın olan düğümdeki iş sıralaması $i = \ddot{\pi}(r+1)$ ve $j = \ddot{\pi}(k-1)$ dışında denklem 6.40' da belirtilenle aynıdır) ve bu halef düğüm budanarak bir sonraki halef düğümü oluşturma işlemleri başlar.

Aksi halde, bu halef düğüm k değerine uygun bir zincire alt sınır değerine göre eklenir. Tüm zincirler, k değeri açısından uygun düğümlerin alt sınırlarının artan sıralamasına göre oluşturulur. Bir zincire yeni bir aktif düğüm ekleneceği zaman bu sıralama gözetilir.

Alt sınır değeri üst sınır değerinden küçük olan ve baskınlık kurallarıyla elenmeyen bir halef düğümün ilgili zincire eklenmesinden sonra, eğer bir başka halef düğümün oluşumu mümkün ise, bölünme aşaması yeni bir halef düğüm ile devam eder. Aksi halde, aktif düğümün tüm halefleri oluşturulduğundan dolayı, bu aktif düğüm budanır. Böylelikle, algoritma dallanma aşamasına geçer.

6.1.1.4. Sınırlama

Sınırlama aşamasındaki bir düğümde $k \leq n-1$ ise, hesaplanan $T^{\ddot{\pi}}$ değeri bu düğümdeki çizelge için bir alt sınırdır. Aksi halde ($k = n$), atanacak tek iş kaldığı için hesaplanan $T^{\ddot{\pi}}$ değeri bu düğümdeki çizelgenin toplam gecikmesidir. Düğümdeki $\ddot{\pi}$ çizelgesinin $(k-1)$. pozisyonundaki işin $C^{\ddot{\pi}}_{[k-1]}$ ve $\sum_{l=1}^{k-1} T^{\ddot{\pi}}_{[l]}$ değerleri sırasıyla denklem 6.41 ve 6.42' ye göre hesaplanır.

$$C^{\ddot{\pi}}_{[k-1]} = C^{\ddot{\pi}}_{[k-2]} + s_{[k-2][k-1]} + p_{[k-1]}(k-1)^a \tag{6.41}$$

$$\sum_{l=1}^{k-1} T^{\ddot{\pi}}_{[l]} = \sum_{l=1}^{k-2} T^{\ddot{\pi}}_{[l]} + \max\left\{C^{\ddot{\pi}}_{[k-1]} - d_{[k-1]}, 0\right\} \tag{6.42}$$

Eğer $k = n$ ise, n. pozisyondaki işin $C^{\ddot{\pi}}_{[k]}$ ve $\sum_{l=1}^{k} T^{\ddot{\pi}}_{[l]}$ değerleri de denklem 6.41 ve 6.42 ile hesaplanır ve $k = n$ olduğu için $T^{\ddot{\pi}} = \sum_{l=1}^{k} T^{\ddot{\pi}}_{[l]}$ alt sınırdan ziyade toplam gecikmenin gerçek değeri olur. Eğer $k < n$ ise, henüz atanmamış $j = k, k+1, \cdots, n$ pozisyonlarındaki işler için denklem 6.29 ve 6.30' da belirtilen düzenlemeler ile düğümdeki çizelgeye ait toplam gecikmenin alt sınırı hesaplanır. Bu düzenlemeler (denklem 6.29 ve 6.30), aslında $k = 1$ seviyesindeki kök düğümü ifade etmektedir ve $k > 1$ seviyesindeki düğümler için aşağıda gösterildiği gibi genelleştirilebilir.

$$s_{[j]} = \min\left\{\min_{i\in\{k,k+1,\cdots,n\}, i\neq j}\left\{s_{[i][j]}\right\}, s_{[k-1][j]}\right\}, \forall j \in \{k, k+1, \cdots, n\} \tag{6.43}$$

$$s_{\pi(l)} = s_{N_s(l)}, d_{\pi(l)} = d_{N_d(l)}, p_{\pi(l)} = p_{N_p(l)}, \forall l \in \{1,2,\cdots, n-k+1\} \tag{6.44}$$

Denklem 6.43 ve 6.44' den yola çıkarak oluşturulan π çizelgesinde, $l = 1,2,\cdots, n-k+1$ pozisyonları için $C^{\pi}_{[l]}$ ve $T^{\pi}_{[l]}$ değerleri denklem 6.45 ve 6.46' da verilmiştir. Önerme 6.3' e göre, T^{π} değeri $\ddot{\pi}$ çizelgesinin $k, k+1, \cdots, n$ pozisyonlarındaki işler için toplam gecikmenin alt sınırıdır ve denklem 6.47' de gösterildiği gibi hesaplanır. Tüm bunlar neticesinde, ilgili düğümdeki $\ddot{\pi}$ çizelgesinin toplam gecikmesine ait alt sınır değeri denklem 6.48' de belirtildiği gibidir.

$$C_{[l]}^{\pi} = C_{[k-1]}^{\ddot{\pi}} + \sum_{t=1}^{l} s_{\pi(t)} + p_{\pi(t)} t^{a} \qquad \forall l \in \{1,2,\cdots,n-k+1\} \qquad (6.45)$$

$$T_{[l]}^{\pi} = \max\{C_{[l]}^{\pi} - d_{\pi(l)}, 0\} \qquad \forall l \in \{1,2,\cdots,n-k+1\} \qquad (6.46)$$

$$T^{\pi} = \sum_{l=1}^{n-k+1} T_{[l]}^{\pi} \qquad (6.47)$$

$$T^{\ddot{\pi}} = \sum_{l=1}^{k-1} T_{[l]}^{\ddot{\pi}} + T^{\pi} \qquad (6.48)$$

6.1.1.5. Dallanma

Birçok DS' da, dallanma prosedürü olarak yatay ve dikey arama yöntemleri kullanılmaktadır. Yatay arama yöntemi öncelikle en düşük seviyeli düğümleri seçerken, dikey arama yöntemi en büyük seviyeli düğümleri seçer. Genellikle, kullanılan alt sınır yeterince kuvvetli (optimum çözüme olabildikçe yakın olan) olmadığında, dikey arama yöntemi tercih edilmektedir. Bu tezde her iki yöntem beraber kullanılmasına rağmen, çoğunlukla dikey arama yöntemi tercih edilmiştir.

Algoritma ilk olarak dikey arama yöntemiyle başlamaktadır ve üst sınır değeri 3 kez iyileştiğinde yatay arama yöntemine geçmektedir. Yatay arama yöntemi, $k \leq 3$ olan aktif düğüm kalmayıncaya kadar devam etmektedir. Yatay arama tamamlanınca algoritma tekrar dikey aramaya geçip sonlanana kadar bu yöntemle devam etmektedir. Tüm zincir yapıları alt sınır değerinin artan sıralamasına uygun olduğundan dolayı, her iki arama yönteminde de ilgili zincirin ilk düğümü seçilmektedir. Böylece, her seferinde üst sınırı iyileştirebilecek potansiyeldeki düğüme ulaşmak için harcanan zamanda tasarruf sağlanmış olmaktadır. Ayrıca, bir zincirdeki ilk düğümün alt sınır değerinin üst sınır değerinden küçük olmaması durumunda, zincirdeki tüm düğümlerin budanması için ikinci bir kontrol gerekmemektedir. Dallanma aşamasında eğer uygun bir aktif düğüm bulunamaz ise, algoritma sonlanır. Aksi halde, algoritma bulunan aktif düğümün haleflerini oluşturmak üzere bölünme aşamasına geçer.

6.1.1.6. Budama

Geliştirilen DS' da bir düğüm üç durumda budanır. Bir düğümün bölünme aşamasında tüm haleflerinin oluşturulmasından sonra budanması, bu durumlardan biridir. Diğer bir durum da, bir düğüme baskın bir başka düğüm olduğunun baskınlık kuralları ışığında kanıtlanmasıdır. Son olarak, bir düğüm alt sınır değeri üst sınır değerinden küçük değilse budanır.

YA ($\pi_{\text{üst sınır}}$) → ($\pi_{\text{üst sınır}}$):	
Adım 1:	$\pi \leftarrow \pi_{\text{üst sınır}}$ ve $k \leftarrow 0$.
Adım 2:	$\pi \leftarrow$ b-insert+ (π).
Adım 3:	Eğer $T^{\pi} < T^{\pi_{\text{üst sınır}}}$ ise $\pi_{\text{üst sınır}} \leftarrow \pi$ ve Adım 2' ye git.
Adım 4:	$\pi \leftarrow$ n-insert+ (π).
Adım 5:	Eğer $T^{\pi} < T^{\pi_{\text{üst sınır}}}$ ise $\pi_{\text{üst sınır}} \leftarrow \pi$, $k \leftarrow 1$ ve Adım 4' e git.
Adım 6:	$\pi \leftarrow$ f-insert+ (π).
Adım 7:	Eğer $T^{\pi} < T^{\pi_{\text{üst sınır}}}$ ise $\pi_{\text{üst sınır}} \leftarrow \pi$, $k \leftarrow 1$ ve Adım 6' ya git.
Adım 8:	Eğer $k = 1$ ise Adım 1' e git.

Şekil 6.1. Yerel arama algoritması

Budanacak düğümlerin belirlenmesinde referans değer olarak kullanıldığından dolayı üst sınır da algoritmanın performansını etkileyen önemli bir faktördür. Bu nedenle, DS' da yatay arama metoduna geçmeden hemen önce, mevcut üst sınır değerine sahip olan çizelgeye yerel arama algoritması (YA) uygulanmıştır. YA, tek bir çözümle başlayan ve mevcut çözümün tanımlanmış bir komşuluğundaki lokal optimum çözümü bulan iteratif bir sezgisel yöntemdir. Tanımlanmış komşuluktaki lokal optimum çözümü bulmak için uyarlanan yöntemler, en iyi gelişmeyi arama stratejisi (best improvement search strategy) olarak adlandırılırken, mevcut çözümden daha iyi olan herhangi bir çözümü bulmak için uyarlanan yöntemler, ilk gelişmeyi arama stratejisi (first improvement search strategy) olarak adlandırılmıştır.

YA' da tek bir komşuluk yapısı olabileceği gibi birden fazla farklı komşuluk yapıları da bir arada kullanılabilmektedir. Kullanılan YA Şekil 6.1' de gösterilmiştir. YA' da kullanılan komşuluk yapıları; ileriye kaydırma (b-insert, Bkz. Şekil 6.4d), ikili kaydırma (n-insert, Bkz. Şekil 6.4e) ve geriye kaydırmadır (f-insert, Bkz. Şekil 6.4c). Bu komşuluklardaki en iyi gelişmeyi aramak için, ileriye kaydırma (b-insert+, Bkz. Şekil 6.11), ikili kaydırma (n-insert+, Bkz. Şekil 6.13) ve geriye kaydırma (f-insert+, Bkz. Şekil 6.9) operatörleri kullanılmıştır. Bahsedilen komşuluk yapıları ve bu komşuluklara ait çözümlerde arama yapan operatörler Bölüm 6.1.4.3' de detaylı şekilde anlatılmıştır.

Şekil 6.1' de görüldüğü üzere, YA mevcut üst sınır değerine sahip olan çizelgeyi ($\pi_{\text{üst sınır}}$) başlangıç çözümü olarak kullanmaktadır (Adım 1). Öncelikle, mevcut çözüm (π) ve ileriye kaydırma komşuluk yapısına (b-insert) göre olan tüm komşu çözümler arasındaki en iyi çözüm belirlenip (b-insert+) mevcut çözüme atanmaktadır (Adım 2). Yeni mevcut çözümün toplam gecikmesi (T^{π}) üst sınır değerine sahip olan çizelgenin toplam gecikmesinden ($T^{\pi_{\text{üst sınır}}}$) küçük ise, üst sınır değeri bu yeni çözümle güncellenir ve aynı komşuluktaki arama tekrarlanır (Adım 3). İleriye kaydırma komşuluk yapısına göre gelişme olmaz ise, mevcut çözüm (π) ve ikili kaydırma komşuluk yapısına (n-insert) göre olan tüm komşu çözümler arasındaki en iyi çözüm belirlenip (n-insert+) mevcut çözüme atanmaktadır (Adım 4). Bu atama sonucu oluşan yeni mevcut çözümün toplam gecikmesi (T^{π}) üst sınır değerine sahip olan çizelgenin toplam gecikmesinden ($T^{\pi_{\text{üst sınır}}}$) küçük ise, üst sınır değeri bu yeni çözümle güncellenir, kontrol parametresine (k) 1 değeri atanır ve aynı komşuluktaki arama tekrarlanır (Adım 5). İkili kaydırma komşuluk yapısına göre gelişme olmaz ise, mevcut çözüm (π) ve geriye kaydırma komşuluk yapısına (f-insert) göre olan tüm komşu çözümler arasındaki en iyi çözüm belirlenip (f-insert+) mevcut çözüme atanmaktadır (Adım 6). Bu atama sonucu oluşan yeni mevcut çözümün toplam gecikmesi (T^{π}) üst sınır değerine sahip olan çizelgenin toplam gecikmesinden ($T^{\pi_{\text{üst sınır}}}$) küçük ise, üst sınır değeri bu yeni çözümle güncellenir, kontrol parametresine (k) 1 değeri atanır ve aynı komşuluktaki arama tekrarlanır (Adım 7). Geriye kaydırma komşuluk yapısına göre gelişme olmaz ise, YA son adıma geçer (Adım 8). Bu adımda; kontrol parametresi yardımıyla Adım 4 veya 6' da en az 1 gelişme olup olmadığına bakılır. Eğer gelişme olmuş ise, Adım 1' den itibaren

algoritma yeni bir iterasyona geçer (Adım 8). Aksi halde, YA sonlanır ve çıktı olarak $\pi_{\text{üst sınır}}$ çizelgesini verir.

6.1.2. Genetik Algoritma-1

Kalıtsal gelişimi taklit eden çözüm yöntemleri için 'genetik algoritma' ifadesi ilk olarak Holland [250] tarafından kullanılmıştır. GA' lar, kendi oluşturdukları çok çeşitli bir popülasyonun en uygun bireylerinden, genetik operatörler vasıtasıyla yeni bir nesil üreten ve bu üretimi döngüsel bir yapıda yineleyen yapay evrimsel süreçlerdir. Üretilen her yeni nesil bir sonraki üretilecek neslin kaynağını teşkil eder ve genellikle bir önceki nesilden daha verimlidir.

Bu verimliliğin ölçüsü, popülasyonu oluşturan bireylerin bazı özellikler bakımından arzu edilen düzeye yakınlığıyla belirlenir ve genellikle uygunluk değeri olarak adlandırılır. Uygunluk değeri iyi olan bireylerden oluşan bir popülasyon elde etmek, ancak doğru çalışan genetik operatörlerle mümkün olur. GA' nın temel operatörleri çaprazlama, mutasyon ve yeniden üretimdir. Çaprazlama ve mutasyon operatörleri, sistematik bir şekilde mevcut nesildeki bireylerin kalıtımsal özelliklerini barındıran yeni bireyler üretirler. Yeniden üretim operatörü ise, çaprazlama ve mutasyon sonucu meydana gelen yeni bireyler ile mevcut popülasyondaki bireyler arasından sistematik bir seçimle belirlediği bireylerden yeni bir nesil oluşturur. GA' larla ilgili daha geniş bilgi için [251] bakılabilir. Bu tezde geliştirilen GA1' in operatörleriyle birlikte yapısal tüm özellikleri aşağıda detaylı olarak anlatılmıştır.

6.1.2.1. Birey Yapısı

GA' larda probleme ait bir çözümü temsil eden en temel öğe bireydir. Bu nedenle, popülasyondaki herhangi bir birey, çözüm uzayındaki herhangi bir olurlu çözümü temsil edebilecek yapıda olmalıdır. Ayrıca, bireyin yapısı genetik operatörlerin sağlıklı şekilde çalışmasına da olanak vermelidir. Bölüm 6.1.1' de anlatıldığı üzere, tanımlanan probleme ait olurlu bir çizelge (çözüm) işlerin permütasyonlu

gösterimiyle ifade edilmiştir. Permütasyonlu gösterim işlerin ardışık tamsayılarla ifade edildiği tek boyutlu bir diziyle temsil edilir ve bu yapı özellikle tek makine çizelgeleme problemlerinde yaygın olarak tercih edilmiştir [131, 252–256]. Dolayısıyla, GA1' de de birey yapısı olarak uygulama pratikliği açısından permütasyonlu gösterim kullanılmıştır.

6.1.2.2. Başlangıç Popülasyonu

Başlangıç çözümü, sonuç kalitesi açısından sezgisel yöntemlerin tümünde oldukça önemli ve etkilidir. GA popülasyon tabanlı bir yöntem olduğundan, başlangıç çözümü çözüm uzayında bulunan belirli miktardaki alternatiflerden meydana gelir. Makine çizelgeleme problemlerinde, başlangıç çözümü olarak basit sıralama kurallarına sıkça başvurulmuştur. Tamamlanma zamanına dayanan performans ölçütleri için SPT sıralama kuralı, teslim tarihine dayanan performans ölçütleri için EDD sıralama kuralı ile oluşturulan çizelgeler tercih edilmektedir [28]. Bunun yanında, ATCS kuralı da toplam (ağırlıklı) gecikme için kullanılan bir yöntemdir. Buradan yola çıkarak, GA1 için tanımlanan problemin yapısına uygun hızlı ve etkili yöntemlerin meydana getirdiği alternatif çözümler tercih edilmiştir. Böylece, her biri başlangıç popülasyonunun %5 lik kısmını oluşturacak şekilde SPT, EDD ve ATCS kurallarından elde edilen çizelgeler ve bu çizelgelerdeki bazı komşu işlerin kendi aralarında yerleri değiştirilerek farklılaştırılan çizelgeler kullanılmıştır. Başlangıç popülasyonunun kalan %85 lik kısmı ise, işlerin rasgele dizilimiyle elde edilen çizelgelerden oluşturulmuştur.

6.1.2.3. Uygunluk Fonksiyonu

GA' larda bireylerin kalitesi uygunluk değeriyle belirlenmektedir. Uygunluk değeri iyi olan bireyler, tüm genetik operatörler için uygunluk değeri oranında öncelik kazanırlar ve bundan dolayı, algoritmanın performansını büyük ölçüde etkilerler. Bu sebeple, bir popülasyondaki bireylerin uygunluk değerlendirilmesi sistematik ve standart bir yönteme bağlanmalıdır.

Geliştirilen GA1' deki uygunluk değeri, TB' ye benzer şekilde en iyi çözümün iyileşmesine bağlı olarak güncellenen bir sıcaklık parametresiyle düzenlenmiştir. Sıcaklık parametresi, uygunluk değeri zayıf olan bireylerin de genetik operatörlerce dikkate alınmasını sağlayarak, popülasyon çeşitliliğinin artması için kullanılmıştır. Anlatımda kolaylık için bazı ifadelerin simgesel gösterimleri aşağıda verilmiştir.

℃	sıcaklık parametresi.
Γ	mevcut popülasyon.
i	mevcut popülasyondaki bir birey $i \in \Gamma$.
π_i	i bireyinin temsil ettiği çizelge.
π^*	mevcut en iyi bireyin temsil ettiği çizelge.
ΔT_i	i bireyinin mevcut en iyi çözümden sapma oranı.
$\mathcal{F}$	uygunluk fonksiyonu $\mathcal{F} = 1/\exp(\Delta T/℃)$.
$\mathcal{F}(i)$	i bireyinin uygunluk değeri.

Bu gösterimden yola çıkarak, bir i bireyinin ΔT_i ve $\mathcal{F}(i)$ değerleri denklem 6.49 ve 6.50' de verildiği gibi hesaplanmaktadır. Bunun yanında, eğer $\Delta T/℃ > 5$ ise, $\Delta T/℃ = 5$ kabul edilmiştir, çünkü $1/\exp(5) \cong 0.007$ olmaktadır ve bu değer uygunluk fonksiyonunun (0,1] aralığında alabileceği diğer değerlere göre anlamsızdır.

$$\Delta T_i = 100\left(T^{\pi_i} - T^{\pi^*}\right)/T^{\pi^*} \tag{6.49}$$

$$\mathcal{F}(i) = 1/\exp(\Delta T_i/℃) \tag{6.50}$$

6.1.2.4. Çaprazlama ve Mutasyon Operatörleri

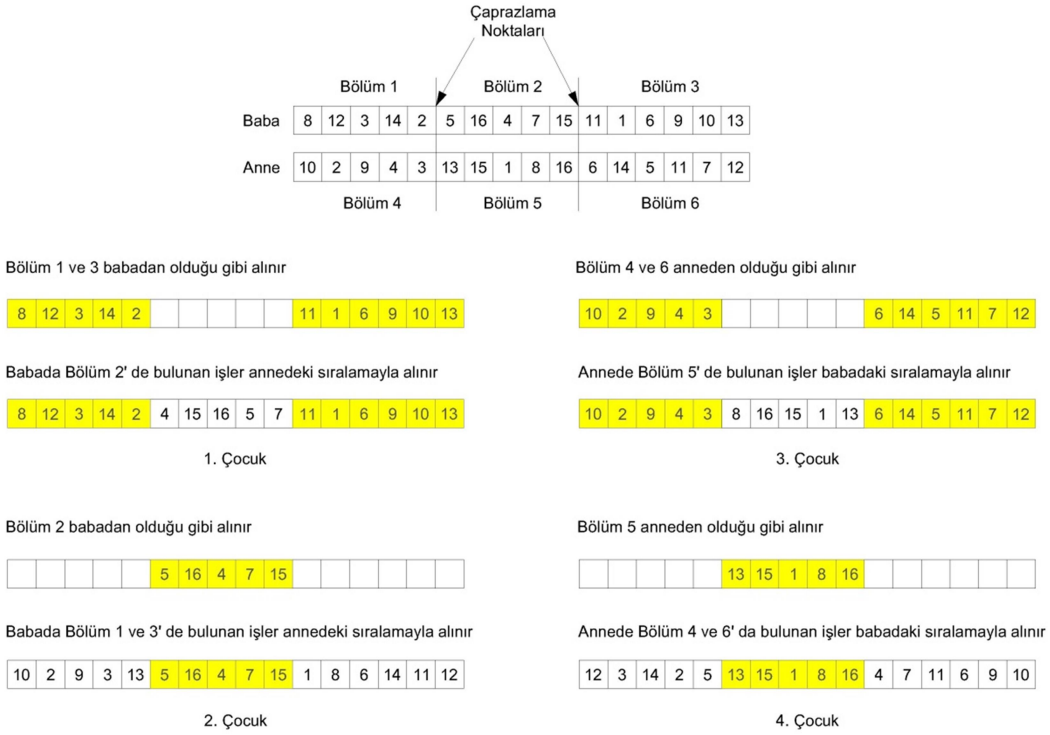
GA' nın en iyi çözüme ulaşmasında işlevsel özelliğini belirleyen, çaprazlama ve mutasyon operatörlerinin yapısıdır. Çaprazlama operatörü, genellikle iki farklı bireyin kalıtımsal özelliklerinin karışımıyla, uygunluk değeri daha iyi olan bireyler üretmek amacıyla tasarlanır. Mutasyon operatörü ise, uygulandığı bireyin yapısını çok değiştirmeden potansiyel iyileşmeleri yakalamak üzere tasarlanır. Bu çerçeveden

bakınca, her iki operatör de üretilecek yeni nesildeki bireylerin daha iyi uygunluk değerine sahip olmaları için kullanılmaktadır.

GA yapısındaki evrimsel gelişim süreci, her zaman daha iyi bireylerin genetik işlemlerden geçerek yeni bireyler oluşturmasına bağlıdır. Bu iyileşme sürecinde, iyi bireyler elde etmenin yanında, çeşitliliğin de önemi büyüktür. Çeşitlilik, iyileşmeye yol açacak bazı genetik yapıların kaybolması ihtimalini düşürürken, daha iyi bireylerin oluşmasını da kolaylaştırır. Bu nedenle, genetik operatörlerin uygulanacağı bireyler sistematik bir seçimle belirlenmelidir. Seçim metodu, her bireye en azından uygunluk değeri oranında seçilme şansı vermelidir.

Geliştirilen GA1' de çaprazlamanın uygulanacağı bireyler rulet çarkı yöntemiyle [257] belirlenmiştir. Her bir aday birey, uygunluk değerine göre seçilme olasılığına sahiptir ve uygunluk değeri iyi olan bireylerin seçilme olasılığı yüksek olmaktadır. Bunun yanında, seçilen adayların çaprazlama işlemine girip giremeyeceği çaprazlama oranına göre belirlenmektedir. Rulet çarkı ile belirlenen her aday için [0,1) aralığında rasgele bir sayı türetilerek çaprazlama oranıyla kıyaslanır. Rasgele türetilen sayı çaprazlama oranından büyük değil ise, seçilen bireye çaprazlama işlemi uygulanır.

Kellegöz vd. [258], özellikle bireylerin permütasyonlu gösterimle ifade edildiği durumlarda kullanılan çeşitli çaprazlama operatörlerini $1||\sum_j w_j T_j$ problemi için geliştirdikleri GA aracılığıyla değerlendirmişlerdir. Bu değerlendirmenin sonuçlarından esinlenerek [259]' de uygulamalarıyla birlikte ilk kez tarif edilen ve daha sonra da [14]' de kullanılan iki noktalı çaprazlama metodu tercih edilmiştir (Bkz. Şekil 6.2). Bu çaprazlama operatöründen dört farklı yeni birey elde edilmektedir, ancak bu 4 bireyin arasından en iyi olanı bir sonraki nesle aktarılmıştır. Görüldüğü üzere, tercih edilen çaprazlama operatörü belirtilen birey gösterimini ihlal etmeyecek şekilde çalışmaktadır. Dolayısıyla, olurlu olmayan yapıda birey oluşumu engellenmekte ve herhangi bir düzeltmeye ihtiyaç duyulmamaktadır.



Şekil 6.2. Çaprazlama yöntemi

Mutasyon operatörü olarak, [260]' deki GA' da önerilen, geriye kaydırma komşuluk yapısındaki (f-insert, Bkz. Şekil 6.4c) tüm komşu çözümler arasından birini eşit olasılıkla rasgele belirleyen yöntem (f-insert-, Bkz. Şekil 6.10) kullanılmıştır. Bu operatörde, eşit olasılıkla rasgele belirlenen bir gen, yine eşit olasılıkla rasgele belirlenen ve kendi konumundan daha ilerde bulunan başka bir konuma yerleştirilirken, diğer genler geriye doğru kaydırılmaktadır. Mutasyona uğrayacak birey uygunluk değerinden bağımsız olarak eşit olasılıkla rasgele belirlenmektedir ve çaprazlamada olduğu gibi mutasyon oranına göre, mutasyon işleminin yapılıp yapılmayacağına karar verilmektedir.

6.1.2.5. Yeniden Üretim

Yeni bir nesil oluşturmak için kullanılan birçok yöntem vardır. Bunlardan bazıları, rasgele seçim, elitizm, rulet çarkı yöntemi, turnuva yöntemi ve rank metodudur

[261]. Bu tezde önerilen GA1' de, popülasyon miktarı sabit kalacak şekilde karışık bir yöntem uygulanmıştır. İlk olarak, mevcut en iyi birey YA ile (Bkz. Şekil 6.1) iyileştirilerek, yeni nesle aktarılmıştır. Popülasyonun yarısı, çaprazlama sonucu oluşan ve kabul edilmiş yeni bireylerden, üçte biri mutasyon sonucu oluşan yeni bireylerden ve geriye kalanı mevcut popülasyondaki bireyler arasında yapılan turnuva metoduyla belirlenmiştir. Uygulanan turnuva metodunda, eşit olasılıkla rasgele belirlenen iki farklı birey arasından uygunluk değeri büyük olan seçilmiştir. Ek olarak, çaprazlama ve mutasyon ile meydana gelen ve yeni nesle aktarılmak üzere seçilen bireyler arasındaki en iyi bireylere de YA uygulanmıştır.

6.1.2.6. Algoritmayı Sonlandırma

Algoritmayı yönlendirmek için iki farklı sayaç kullanılmıştır. Sayaçlardan biri sıcaklık parametresini kontrol ederken, bir diğeri algoritmayı sonlandırma işlemine karar vermektedir. Her iki sayaç da iyileşme olmayan ardışık nesil sayısını takip etmektedir. İyileşmeden kasıt, mevcut en iyi bireyin değişimidir. İyileşme olduğunda her iki sayaç da sıfırlanmaktadır. Sıcaklık sayacı belirlenen bir değere ulaştığında, mevcut sıcaklık değeri atamasından bu yana en az bir iyileşme olup olmadığına bakar ve eğer var ise, sıcaklık değerini artırır, aksi halde sıcaklık değerini azaltır. Sonlandırma sayacı ise, belirlenen değere ulaştığında algoritmayı sonlandırır.

6.1.3. Genetik Algoritma-2

Bölüm 6.1.2' de anlatılan GA1' den farklı olarak, GA2' de uygunluk fonksiyonundaki sıcaklık parametresi kullanılmamıştır. Böylece, uygunluk fonksiyonu $\mathcal{F} = 1/\Delta T$ olarak formülüze edilmiştir. Burada kullanılan uygunluk değeri sınırlaması ise şu şekilde yapılmıştır: eğer $\Delta T > 300$ ise, $\Delta T = 300$, aksi halde eğer $\Delta T < 1$ ise $\Delta T = 1$.

6.1.4. Değişken Komşu Arama Algoritması-1

DKA, çeşitli komşuluk yapılarını kullanarak, kombinatoryal optimizasyon problemlerinin çözüm uzayını sistematik bir biçimde araştıran iki aşamalı metasezgisel bir yöntemdir. İyileşme aşamasında lokal optimum çözüm elde etmek, karıştırma aşamasında ise, bu lokal alandan kurtulmak için mevcut çözümün komşuluklarında sistematik bir arama yapar. DKA, ilk önerildiğinden bu yana [262], uygulama alanları ve algoritma yapıları bakımından oldukça yaygınlaşmıştır [263]. Kirlik ve Oğuz [232], $1|s_{ij}|\sum_j w_j T_j$ problemi için oldukça başarılı bir DKA geliştirmişlerdir. Yazarların bu çalışmasındaki DKA, ele alınan problem için yeniden uyarlanmıştır ve kullanılan bazı simgeler aşağıda tanımlanmıştır.

$\mathcal{N}$	komşuluk yapılarının kümesi, $\mathcal{N} = \{\mathcal{N}_1, \mathcal{N}_2, \cdots, \mathcal{N}_\omega\}$.
ω	komşuluk yapılarının sayısı.
$\mathcal{N}_k(\pi)$	bir $\pi \in D$ (denklem 6.1) çözümünün k. komşuluk yapısına uygun bir komşu çözüm, $k = 1,2,\cdots,\omega$ için $\forall \mathcal{N}_k(\pi) \in B \subseteq D$.
$\mathcal{N}_{k+}(\pi)$	bir $\pi \in D$ çözümünün k. komşuluk yapısına uygun komşu çözümlerin en iyisi, $k = 1,2,\cdots,\omega$, $\mathcal{N}_{k+}(\pi) \in B \subseteq D$ ve $\forall \mathcal{N}_k(\pi) \in B \subseteq D$ için $T^{\mathcal{N}_{k+}(\pi)} \leq T^{\mathcal{N}_k(\pi)}$.
$\mathcal{N}_{k-}(\pi)$	bir $\pi \in D$ çözümünün k. komşuluk yapısına uygun komşu çözümlerin arasından eşit olasılıkla rasgele belirlenen bir komşu çözüm, $\mathcal{N}_{k-}(\pi) \in B \subseteq D$.

DKA1, temel olarak karıştırma (shaking - SHK-1) ve değişken komşu iniş (DKİ) algoritmalarıyla ifade edilen iki operatörden meydana gelmektedir. Tek bir çözümle $\pi \in D$ başlayan DKA1, SHK-1 operatöründen elde ettiği komşu çözümle $\mathcal{N}_{k-}(\pi) \in B \subseteq D$ farklı lokal alanlara geçişi sağlayarak, DKİ operatörüyle bu alandaki en iyi çözüme $\mathcal{N}_{k+}(\pi) \in B \subseteq D$ ulaşmayı hedefler.

DKA1 (π_{ATCS}, ω, t, i_{max}, $\mathcal{N}$) → (π^*):

Adım 1: $\pi^* \leftarrow \pi_{ATCS}$.

Adım 2: $i \leftarrow 1$.

Adım 3: $k \leftarrow 1$.

Adım 4: $\pi \leftarrow \pi^*$.

Adım 5: SHK-1 (π, $\mathcal{N}_k$, t) → (π).

Adım 6: DKİ (π, ω, $\mathcal{N}$) → (π).

Adım 7: Eğer $T^{\pi} < T^{\pi^*}$ ise $\pi^* \leftarrow \pi$ ve Adım 2' e git. Aksi halde $k \leftarrow k + 1$.

Adım 8: Eğer $k \leq \omega$ ise Adım 4' e git. Aksi halde $i = i + 1$.

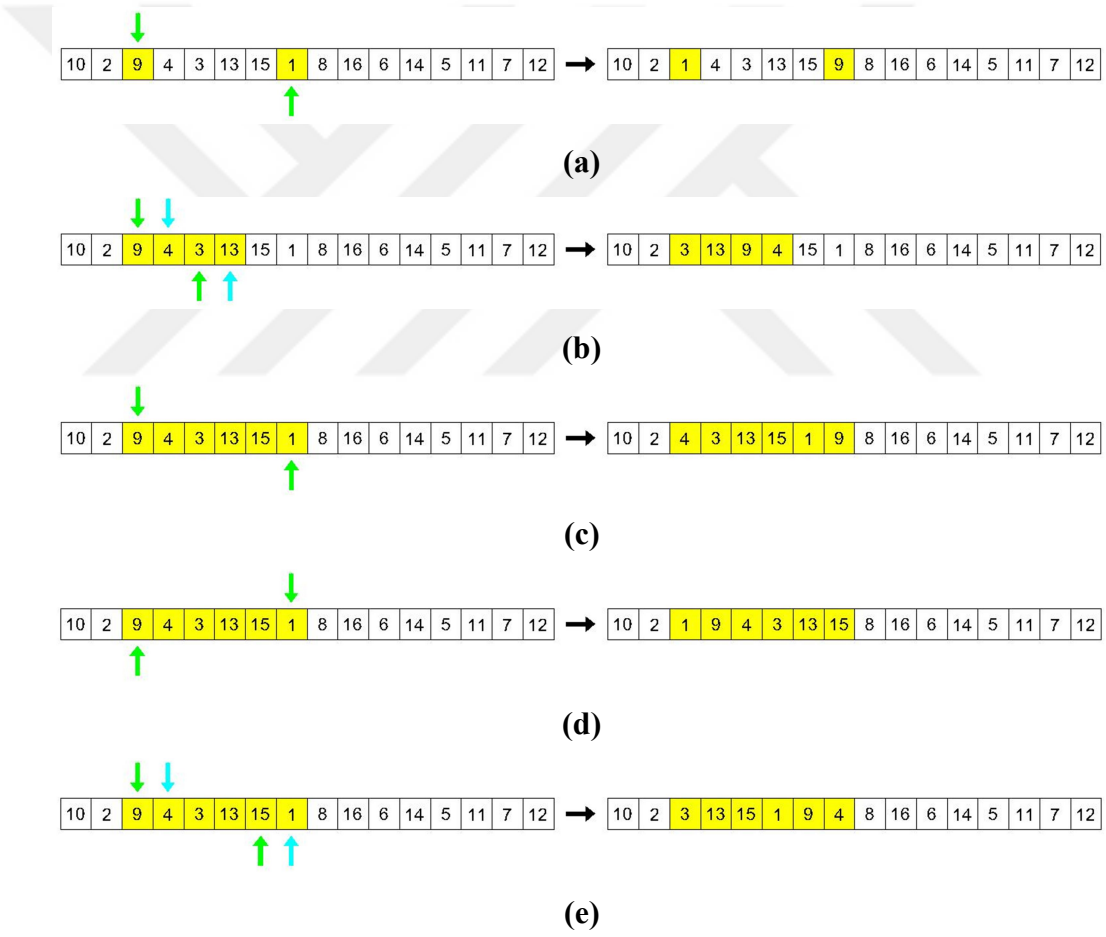
Adım 9: Eğer $i \leq i_{\max}$ ise Adım 3' e git.

Şekil 6.3. Değişken komşu arama algoritması-1

Şekil 6.3' de görüldüğü üzere, DKA1' de ATCS kuralından elde edilen çizelge (π_{ATCS}) başlangıç çözümü olarak kullanılmıştır ve bilinen en iyi çözüm (π^*) olarak belirlenmiştir (Adım 1). Öncelikle, iterasyon sayacına (i) ve belirlenen sırayla oluşturulmuş komşuluk kümesinin ($\mathcal{N}$) sıra sayacına (k) 1 değeri atanmaktadır (Adım 2 ve 3). Daha sonra mevcut çözüm (π) bilinen en iyi çözümle (π^*) güncellenmektedir (Adım 4). Mevcut çözüm (π), mevcut komşuluk yapısı ($\mathcal{N}_k$) ve karıştırma kontrol sayacı (t) bilgilerini kullanan karıştırma operatörü SHK-1, mevcut çözümün ilgili komşuluk yapısına ait komşu çözümlerinden birini belirleyerek çıktı olarak verir (Adım 5).

SHK-1 operatörünün çalışma prensibi ve kullandığı komşuluk operatörleri ilerleyen kısımlarda detaylı bir şekilde anlatılmıştır (Bkz. Şekil 6.15). Karıştırma aşamasının çıktısı olan mevcut çözüm (π), geçiş sağlanan yeni alandaki lokal optimum çözümü aramak üzere iyileşme aşamasına geçer. Bu aşamada; mevcut çözüm (π), komşuluk kümesi ($\mathcal{N}$) ve komşuluk sayısı (ω) bilgilerini kullanan iyileşme operatörü DKİ, sırasıyla mevcut çözümün tüm komşuluk yapılarına ait komşu çözümlerinden en iyisini belirleyerek çıktı olarak verir (Adım 6).

DKİ operatörünün çalışma prensibi ve kullandığı komşuluk operatörleri ilerleyen kısımlarda detaylı bir şekilde anlatılmıştır (Bkz. Şekil 6.16). İyileşme aşamasından elde edilen mevcut çözümün toplam gecikmesi (T^{π}) bilinen en iyi çözümün toplam gecikmesinden (T^{π^*}) küçük ise, bilinen en iyi çözüm mevcut çözümle güncellenir ve mevcut iterasyon tekrarlanır (Adım 7). Aksi halde, bir sonraki komşuluk yapısına geçilir. Yeni komşuluk yapısıyla SHK-1 ve daha sonra DKİ operatörleri tekrarlanır. Belirlenen son komşuluk yapısında da gelişme olmaz ise, bir sonraki iterasyona geçilir (Adım 8). Belirlenen iterasyon sayısına (i_{max}) ulaşıldığında DKA1 sonlanır (Adım 9).



Şekil 6.4. Komşuluk yapıları: (a) iki işin yer değişimi, (b) ikili yer değişimi, (c) geriye kaydırma, (d) ileriye kaydırma, (e) ikili kaydırma

6.1.4.1. Çözüm Yapısı

Çözümlerin temsiliyeti, kurgulanan komşuluk yapılarının sağlıklı uygulanmasını sağlayacak şekilde olmalıdır. Mevcut bir çözümden sistematik bir şekilde olurlu komşu çözümler üretmeye müsait bir yapı tercih edilmelidir. Geliştirilen DKA1' de çözümler, tanımlanan problem yapısına da uygun olduğundan dolayı permütasyonlu gösterimle ifade edilmiştir.

6.1.4.2. Başlangıç Çözümü

Diğer tüm sezgisel yöntemlerde olduğu gibi, DKA1 de bir başlangıç çözümüyle devreye girer. Karıştırma ve iniş aşamalarının işlevsel özelliklerinden dolayı, DKA1 tek çözüm tabanlı bir metasezgiseldir ve tüm operatörler tek bir çözüm üzerinde çalışacak şekilde tasarlanmıştır. Başlangıç çözümünün kalitesi algoritmanın performansını etkilediğinden dolayı, mümkün olan en iyi çözüm başlangıç çözümü olarak tercih edilmelidir. Bu nedenle, ATCS kuralından elde edilen çizelge başlangıç çözümü olarak kullanılmıştır.

6.1.4.3. Komşuluk Yapıları

DKA' dan elde edilen bir çözümün kalitesi temel olarak komşuluk yapılarına bağlıdır. Değişik komşulukların kullanılması bir anlamda DKA' nın karakteristik özelliğidir. Tanımlanan problemi tüm yönleriyle doğru yansıtan komşuluk yapıları belirlemek bu açıdan önemlidir. Hansen vd. [263] değişik komşuluk yapılarının kullanılmasına gerekçe olan bazı gözlemlerini üç maddede toplamıştır.

i. Bir komşuluk yapısına göre lokal optimum olan bir sonuç, başka bir komşuluk yapısına göre lokal optimum olmayabilir.
ii. Global optimum olan bir sonuç tüm komşuluk yapıları için lokal optimumdur.
iii. Birçok farklı problemde gözlenen deney sonuçlarına göre, lokal optimum çözümlerin büyük çoğunluğu (tamamı) birbirlerine nispeten yakındır [264].

Bu tezde önerilen DKA1 için, literatürde sıklıkla karşılaşılan 5 farklı komşuluk yapısı dikkate alınmıştır. Bu komşuluk yapıları, iki işin yer değişimi (swap), ikili yer değişimi (n-swap), geriye kaydırma (f-insert), ileriye kaydırma (b-insert) ve ikili kaydırmadır (n-insert).

swap+ $(\pi) \rightarrow (\mathcal{N}_{k+}(\pi)|k \in \{1,2,\cdots,\omega\})$:

Adım 1: $T \leftarrow T^{\pi}$ ve $i \leftarrow 1$.

Adım 2: $j \leftarrow i+1$ ve $p \leftarrow \pi(i)$.

Adım 3: $\pi(i) \leftarrow \pi(j)$ ve $\pi(j) \leftarrow p$.

Adım 4: Eğer $T^{\pi} < T$ ise $T \leftarrow T^{\pi}$, $i^{*} \leftarrow i$ ve $j^{*} \leftarrow j$.

Adım 5: $\pi(j) \leftarrow \pi(i)$ ve $j \leftarrow j+1$.

Adım 6: Eğer $j \leq n$ ise Adım 3' e git. Aksi halde $\pi(i) \leftarrow p$ ve $i \leftarrow i+1$.

Adım 7: Eğer $i < n$ ise Adım 2' ye git.

Adım 8: Eğer $T < T^{\pi}$ ise $p \leftarrow \pi(i^{*})$, $\pi(i^{*}) \leftarrow \pi(j^{*})$ ve $\pi(j^{*}) \leftarrow p$.

Şekil 6.5. İki işin yer değişimi komşuluğundaki en iyi çözüm operatörü

Swap komşuluğunda (Şekil 6.4a), belirlenen 2 farklı pozisyondaki işler birbirleri arasında yer değiştirir. Bu komşuluk yapısına uygun çözümlerin oluşturduğu bir komşuluk kümesinde lokal optimum çözümü elde etmek için swap+ operatörü (Şekil 6.5) kullanılmaktadır. Bu operatörde, i. pozisyondaki işle $[i+1, n]$ aralığındaki pozisyonlardan birinde (j) bulunan iş yer değiştirilir (Adım 2 ve 3) ve bu değişimle oluşan yeni komşu çözümde gelişme olup olmadığı kontrol edilir (Adım 4). Eğer gelişme var ise, bilinen en iyi sonucu saklayan değişkene (T) komşu çözümün sonucu (T^{π}) atanır ve bu gelişmeyi sağlayan pozisyonları saklayan değişkenler (i^{*} ve j^{*}) güncellenir (Adım 4). Daha sonra, j. pozisyonda yapılan değişiklik geri alınır (Adım 5) ve $(j+1)$. pozisyona geçilerek aynı işlemler n. pozisyon da dahil olmak üzere sırayla tüm pozisyonlar için tekrarlanır (Adım 6). Komşu çözümler i. pozisyon için tamamlandığında i. pozisyonda yapılan değişiklik geri alınır (Adım 6) ve $(i+1)$. pozisyona geçilerek aynı işlemler $(n-1)$. pozisyon da dahil olmak üzere sırayla

tüm pozisyonlar için tekrarlanır (Adım 7). Belirlenen sistematikle oluşabilecek tüm komşu çözümler arasındaki en iyi çözüm mevcut çözümden daha iyi ise, bu gelişmeyi sağlayan pozisyonlardaki işler arasında yer değişimi yapılır (Adım 8).

swap- $(\pi) \rightarrow (\mathcal{N}_{k-}(\pi) \mid k \in \{1,2,\cdots,\omega\})$:	
Adım 1:	$r \in \mathbb{R} \leftarrow [0,1)$ aralığında rasgele bir sayı ve $i \leftarrow \lfloor r(n-1) \rfloor + 1$.
Adım 2:	$r \in \mathbb{R} \leftarrow [0,1)$ aralığında rasgele bir sayı ve $j \leftarrow \lfloor r(n-i) \rfloor + i + 1$.
Adım 3:	$p \leftarrow \pi(i)$, $\pi(i) \leftarrow \pi(j)$ ve $\pi(j) \leftarrow p$.

Şekil 6.6. İki işin yer değişimi komşuluğundaki rasgele çözüm operatörü

n-swap+ $(\pi) \rightarrow (\mathcal{N}_{k+}(\pi) \mid k \in \{1,2,\cdots,\omega\})$:	
Adım 1:	$T \leftarrow T^{\pi}$ ve $i \leftarrow 1$.
Adım 2:	$j \leftarrow i+2$, $p \leftarrow \pi(i)$ ve $r \leftarrow \pi(i+1)$.
Adım 3:	$\pi(i) \leftarrow \pi(j)$, $\pi(i+1) \leftarrow \pi(j+1)$, $\pi(j) \leftarrow p$ ve $\pi(j+1) \leftarrow r$.
Adım 4:	Eğer $T^{\pi} < T$ ise $T \leftarrow T^{\pi}$, $i^* \leftarrow i$ ve $j^* \leftarrow j$.
Adım 5:	$\pi(j) \leftarrow \pi(i)$, $\pi(j+1) \leftarrow \pi(i+1)$ ve $j \leftarrow j+1$.
Adım 6:	Eğer $j < n$ ise Adım 3 git. Aksi halde $\pi(i) \leftarrow p$ ve $\pi(i+1) \leftarrow r$.
Adım 7:	$i \leftarrow i+1$. Eğer $i < n-2$ ise Adım 2' ye git.
Adım 8:	Eğer $T < T^{\pi}$ ise $p \leftarrow \pi(i^*)$, $r \leftarrow \pi(i^*+1)$, $\pi(i^*) \leftarrow \pi(j^*)$, $\pi(i^*+1) \leftarrow \pi(j^*+1)$, $\pi(j^*) \leftarrow p$ ve $\pi(j^*+1) \leftarrow r$.

Şekil 6.7. İkili yer değişimi komşuluğundaki en iyi çözüm operatörü

Swap komşuluk kümesinden eşit olasılıkla rasgele bir çözüm belirlemek için swap- operatörü (Şekil 6.6) kullanılmaktadır. Bu operatörde, $[1, n)$ aralığındaki pozisyonlardan eşit olasılıkla rasgele belirlenen i. pozisyondaki (Adım 1) iş ile $(i, n]$ aralığındaki pozisyonlardan eşit olasılıkla rasgele belirlenen j. pozisyondaki (Adım 2) işin yer değişimi yapılmaktadır (Adım 3).

n-swap- $(\pi) \rightarrow (\mathcal{N}_{k-}(\pi)|k \in \{1,2,\cdots,\omega\})$:

Adım 1: $r \in \mathbb{R} \leftarrow [0,1)$ aralığında rasgele bir sayı ve $i \leftarrow \lfloor r(n-3) \rfloor + 1$.

Adım 2: $r \in \mathbb{R} \leftarrow [0,1)$ aralığında rasgele bir sayı ve $j \leftarrow \lfloor r(n-i-2) \rfloor + i + 2$.

Adım 3: $p \leftarrow \pi(i)$, $l \leftarrow \pi(i+1)$, $\pi(i) \leftarrow \pi(j)$, $\pi(i+1) \leftarrow \pi(j+1)$, $\pi(j) \leftarrow p$ ve $\pi(j+1) \leftarrow l$.

Şekil 6.8. İkili yer değişimi komşuluğundaki rasgele çözüm operatörü

n-Swap komşuluğunda (Şekil 6.4b), belirlenen 2 farklı pozisyondaki işler ve bu işlerden sonraki komşu işler birbirleri arasında yer değiştirir. Bu komşuluk yapısına uygun çözümlerin oluşturduğu bir komşuluk kümesinde lokal optimum çözümü elde etmek için n-swap+ operatörü (Şekil 6.7) kullanılmaktadır. Bu operatörde, mevcut çözümün (π) $i.$ ve $(i+1).$ pozisyonlarındaki işler ile $(i+1, n]$ aralığındaki ardışık iki pozisyonda (j ve $j+1$) bulunan işlerin yerleri değiştirilir (Adım 2 ve 3). Bu değişimle oluşan yeni komşu çözümde gelişme olup olmadığı kontrol edilir (Adım 4). Eğer gelişme var ise, bilinen en iyi sonucu saklayan değişkene (T) komşu çözümün sonucu (T^{π}) atanır ve bu gelişmeyi sağlayan pozisyonları saklayan değişkenler (i^* ve j^*) güncellenir (Adım 4). Daha sonra, $j.$ ve $(j+1).$ pozisyonlarında yapılan değişiklikler geri alınır (Adım 5) ve $(j+1).$ pozisyona geçilerek aynı işlemler $n.$ pozisyona kadar sırayla tüm pozisyonlar için tekrarlanır (Adım 6). Komşu çözümler $i.$ pozisyon için tamamlandığında $i.$ ve $(i+1).$ pozisyonlarında yapılan değişiklikler geri alınır (Adım 6) ve $(i+1).$ pozisyona geçilerek aynı işlemler $(n-2).$ pozisyona kadar sırayla tüm pozisyonlar için tekrarlanır (Adım 7). Belirlenen sistematikle oluşabilecek tüm komşu çözümler arasındaki en iyi çözüm mevcut çözümden daha iyi ise, bu gelişmeyi sağlayan pozisyonlardaki işler ve bu işlerden sonraki komşu işler yer değiştirilir (Adım 8).

n-Swap komşuluk yapısına uygun çözümlerin oluşturduğu bir komşuluk kümesinden eşit olasılıkla rasgele bir çözüm belirlemek için n-swap- operatörü (Şekil 6.8) kullanılmaktadır. Bu operatörde, $[1, n-2)$ aralığındaki pozisyonlardan eşit olasılıkla

rasgele belirlenen $i.$ ve $(i+1).$ pozisyonlarındaki (Adım 1) işler ile $(i+1,\ n)$ aralığındaki pozisyonlardan eşit olasılıkla rasgele belirlenen $j.$ ve $(j+1).$ pozisyonlarındaki (Adım 2) işlerin yer değişimi yapılmaktadır (Adım 3).

f-insert + $(\pi) \rightarrow (\mathcal{N}_{k+}(\pi)|k \in \{1,2,\cdots,\omega\})$:

Adım 1: $T \leftarrow T^{\pi}$ ve $i \leftarrow 1$.

Adım 2: $j \leftarrow i+1$ ve $p \leftarrow \pi(i)$.

Adım 3: $\pi(j-1) \leftarrow \pi(j)$ ve $\pi(j) \leftarrow p$.

Adım 4: Eğer $T^{\pi} < T$ ise $T \leftarrow T^{\pi}$, $i^{*} \leftarrow i$ ve $j^{*} \leftarrow j$.

Adım 5: $j \leftarrow j+1$.

Adım 6: Eğer $j \leq n$ ise Adım 3' e git. Aksi halde $j \leftarrow n$.

Adım 7: $\pi(j) \leftarrow \pi(j-1)$ ve $j \leftarrow j-1$.

Adım 8: Eğer $j > i$ ise Adım 7' ye git. Aksi halde $\pi(i) \leftarrow p$.

Adım 9: $i \leftarrow i+1$. Eğer $i < n$ ise Adım 2' ye git.

Adım 10: Eğer $T < T^{\pi}$ ise $j \leftarrow i^{*}$ ve $p \leftarrow \pi(i^{*})$. Aksi halde sonlandır.

Adım 11: $\pi(j) \leftarrow \pi(j+1)$ ve $j \leftarrow j+1$.

Adım 12: Eğer $j < j^{*}$ ise Adım 11' e git. Aksi halde $\pi(j^{*}) \leftarrow p$.

Şekil 6.9. Geriye kaydırma komşuluğundaki en iyi çözüm operatörü

f-Insert komşuluğunda (Şekil 6.4c), $[1,\ n)$ aralığında belirlenen bir pozisyondaki (i) iş, $(i,\ n]$ aralığındaki başka bir pozisyona (j) yerleştirilirken, $(i,\ j]$ aralığındaki pozisyonlarda bulunan işler aynı sırayla $[i,\ j)$ aralığındaki pozisyonlara kaydırılır. Bu komşuluk yapısına uygun çözümlerin oluşturduğu bir komşuluk kümesinde lokal optimum çözümü elde etmek için f-insert+ operatörü (Şekil 6.9) kullanılmaktadır.

Bu operatörde, $[1,\ n)$ aralığında bulunan $i.$ pozisyondaki iş ile $(i+1).$ pozisyonda bulunan iş yer değiştirilir (Adım 2 ve 3) ve bu değişimle oluşan yeni komşu çözümde gelişme olup olmadığı kontrol edilir (Adım 4). Eğer gelişme var ise, bilinen en iyi sonucu saklayan değişkene (T) komşu çözümün sonucu (T^{π}) atanır ve bu gelişmeyi

sağlayan pozisyonları saklayan değişkenler (i^* ve j^*) güncellenir (Adım 4). Bu değişim i. pozisyondaki iş n. pozisyona gelene kadar tekrarlanır (Adım 5 ve 6). Bu aşamadaki mevcut çözümde, i. pozisyondaki iş n. pozisyondadır ve $(i,\ n]$ aralığındaki pozisyonlarda bulunan işler $[i,\ n)$ aralığındaki pozisyonlara aynı sırayla kaydırılmıştır. Mevcut çözümdeki bu değişimi geri almak için $[i,\ n)$ aralığındaki pozisyonlarda bulunan işler $(i,\ n]$ aralığındaki pozisyonlara aynı sırayla kaydırılır ve yedekleme değişkeninde (p) saklanan değer i. pozisyondaki iş olarak atanır (Adım 6-8). Tüm bu işlemler $[1,\ n)$ aralığındaki pozisyonlarda bulunan işler için sırasıyla tekrarlanır (Adım 9). Belirlenen sistematikle oluşabilecek tüm komşu çözümler arasındaki en iyi çözüm mevcut çözümden daha iyi ise, i^*. pozisyondaki iş j^*. pozisyona atanır ve $(i^*,\ j^*]$ aralığındaki pozisyonlarda bulunan işler $[i^*,\ j^*)$ aralığındaki pozisyonlara sırasıyla kaydırılır (Adım 10-12).

f-insert- $(\pi) \rightarrow (\mathcal{N}_{k-}(\pi) \mid k \in \{1,2,\cdots,\omega\})$:	
Adım	1: $r \in \mathbb{R} \leftarrow [0,1)$ aralığında rasgele bir sayı ve $i \leftarrow \lfloor r(n-1) \rfloor + 1$.
Adım	2: $r \in \mathbb{R} \leftarrow [0,1)$ aralığında rasgele bir sayı ve $j \leftarrow \lfloor r(n-i) \rfloor + i + 1$.
Adım	3: $p \leftarrow \pi(i)$ ve $l \leftarrow i$.
Adım	4: $\pi(l) \leftarrow \pi(l+1)$ ve $l \leftarrow l + 1$.
Adım	5: Eğer $l < j$ ise Adım 4' e git. Aksi halde $\pi(j) \leftarrow p$.

Şekil 6.10. Geriye kaydırma komşuluğundaki rasgele çözüm operatörü

f-Insert komşuluk kümesinden eşit olasılıkla rasgele bir çözüm belirlemek için f-insert- operatörü (Şekil 6.10) kullanılmaktadır. Bu operatörde, $[1, n)$ aralığındaki pozisyonlardan eşit olasılıkla rasgele belirlenen i. pozisyondaki (Adım 1) iş $(i,\ n]$ aralığındaki pozisyonlardan eşit olasılıkla rasgele belirlenen j. pozisyona (Adım 2) yerleştirilirken, $(i, j]$ aralığındaki pozisyonlarda bulunan işler $[i, j)$ aralığındaki pozisyonlara kaydırılır (Adım 3-5).

b-insert + $(\pi) \rightarrow (\mathcal{N}_{k+}(\pi)|k \in \{1,2,\cdots,\omega\})$:

Adım 1: $T \leftarrow T^{\pi}$ ve $i \leftarrow n$.

Adım 2: $j \leftarrow i-1$ ve $p \leftarrow \pi(i)$.

Adım 3: $\pi(j+1) \leftarrow \pi(j)$ ve $\pi(j) \leftarrow p$.

Adım 4: Eğer $T^{\pi} < T$ ise $T \leftarrow T^{\pi}$, $i^{*} \leftarrow i$ ve $j^{*} \leftarrow j$.

Adım 5: $j \leftarrow j-1$.

Adım 6: Eğer $j > 0$ ise Adım 3' e git. Aksi halde $j \leftarrow 1$.

Adım 7: $\pi(j) \leftarrow \pi(j+1)$ ve $j \leftarrow j+1$.

Adım 8: Eğer $j < i$ ise Adım 7' ye git. Aksi halde $\pi(i) \leftarrow p$.

Adım 9: $i \leftarrow i-1$. Eğer $i > 1$ ise Adım 2' ye git.

Adım 10: Eğer $T < T^{\pi}$ ise $j \leftarrow i^{*}$ ve $p \leftarrow \pi(i^{*})$. Aksi halde sonlandır.

Adım 11: $\pi(j) \leftarrow \pi(j-1)$ ve $j \leftarrow j-1$.

Adım 12: Eğer $j > j^{*}$ ise Adım 11' e git. Aksi halde $\pi(j^{*}) \leftarrow p$.

Şekil 6.11. İleriye kaydırma komşuluğundaki en iyi çözüm operatörü

b-insert- $(\pi) \rightarrow (\mathcal{N}_{k-}(\pi)|k \in \{1,2,\cdots,\omega\})$:

Adım 1: $r \in \mathbb{R} \leftarrow [0,1)$ aralığında rasgele bir sayı ve $i \leftarrow \lfloor r(n-1) \rfloor + 2$.

Adım 2: $r \in \mathbb{R} \leftarrow [0,1)$ aralığında rasgele bir sayı ve $j \leftarrow \lfloor r(i-1) \rfloor + 1$.

Adım 3: $p \leftarrow \pi(i)$ ve $l \leftarrow i$.

Adım 4: $\pi(l) \leftarrow \pi(l-1)$ ve $l \leftarrow l-1$.

Adım 5: Eğer $l > j$ ise Adım 4' e git. Aksi halde $\pi(j) \leftarrow p$.

Şekil 6.12. İleriye kaydırma komşuluğundaki rasgele çözüm operatörü

b-Insert komşuluğunda (Şekil 6.4d), $(1, n]$ aralığında belirlenen bir pozisyondaki (i) iş, $[1, i)$ aralığındaki başka bir pozisyona (j) yerleştirilirken, $[j, i)$ aralığındaki pozisyonlarda bulunan işler aynı sırayla $(j, i]$ aralığındaki pozisyonlara kaydırılır. Bu komşuluk yapısına uygun çözümlerin oluşturduğu bir komşuluk kümesinde lokal optimum çözümü elde etmek için b-insert+ operatörü (Şekil 6.11) kullanılmaktadır.

Bu operatörde, $(1,\ n]$ aralığında bulunan i. pozisyondaki iş ile $(i-1)$. pozisyonda bulunan iş yer değiştirilir (Adım 2 ve 3) ve bu değişimle oluşan yeni komşu çözümde gelişme olup olmadığı kontrol edilir (Adım 4). Eğer gelişme var ise, bilinen en iyi sonucu saklayan değişkene (T) komşu çözümün sonucu (T^{π}) atanır ve bu gelişmeyi sağlayan pozisyonları saklayan değişkenler $(i^{*}$ ve $j^{*})$ güncellenir (Adım 4). Bu değişim i. pozisyondaki iş 1. pozisyona gelene kadar tekrarlanır (Adım 5 ve 6). Bu aşamadaki mevcut çözümde, i. pozisyondaki iş 1. pozisyondadır ve $[1,\ i)$ aralığındaki pozisyonlarda bulunan işler $(1,\ i]$ aralığındaki pozisyonlara aynı sırayla kaydırılmıştır. Mevcut çözümdeki bu değişimi geri almak için $(1,\ i]$ aralığındaki pozisyonlarda bulunan işler $[1,\ i)$ aralığındaki pozisyonlara aynı sırayla kaydırılır ve yedekleme değişkeninde (p) saklanan değer i. pozisyondaki iş olarak atanır (Adım 6-8). Tüm bu işlemler $(1,\ n]$ aralığındaki pozisyonlarda bulunan işler için sırasıyla tekrarlanır (Adım 9). Belirlenen sistematikle oluşabilecek tüm komşu çözümler arasındaki en iyi çözüm mevcut çözümden daha iyi ise, i^{*}. pozisyondaki iş j^{*}. pozisyona atanır ve $[j^{*},\ i^{*})$ aralığındaki pozisyonlarda bulunan işler $(j^{*},\ i^{*}]$ aralığındaki pozisyonlara sırasıyla kaydırılır (Adım 10-12).

b-Insert komşuluk kümesinden eşit olasılıkla rasgele bir çözüm belirlemek için b-insert- operatörü (Şekil 6.12) kullanılmaktadır. Bu operatörde, $(1, n]$ aralığındaki pozisyonlardan eşit olasılıkla rasgele belirlenen i. pozisyondaki (Adım 1) iş $[1, i)$ aralığındaki pozisyonlardan eşit olasılıkla rasgele belirlenen j. pozisyona (Adım 2) yerleştirilirken, $[j, i)$ aralığındaki pozisyonlarda bulunan işler $(j, i]$ aralığındaki pozisyonlara kaydırılır (Adım 3-5).

n-Insert komşuluğunda (Şekil 6.4e), $[1,\ n-1)$ aralığında belirlenen i. ve $(i+1)$. pozisyonlarındaki işler $(i,\ n]$ aralığındaki komşu iki pozisyona (j ve $j+1$) yerleştirilirken, $(i+1,\ j]$ aralığındaki pozisyonlarda bulunan işler aynı sırayla $[i,\ j-1)$ aralığındaki pozisyonlara kaydırılır. Bu komşuluk yapısına uygun çözümlerin oluşturduğu bir komşuluk kümesinde lokal optimum çözümü elde etmek için n-insert+ operatörü (Şekil 6.13) kullanılmaktadır. Bu operatörde, $[1,\ n-1)$ aralığında bulunan i. pozisyondaki iş ve $(i+1)$. pozisyondaki komşu iş sırasıyla $(i+1)$. ve $(i+2)$. pozisyonlara, $(i+2)$. pozisyonda bulunan iş de i. pozisyona atanır (Adım 2 ve 3). Eğer bu değişim sonucunda gelişme olur ise, bilinen en iyi sonucu saklayan

değişkene (T) komşu çözümün sonucu (T^{π}) atanır ve bu gelişmeyi sağlayan pozisyonları saklayan değişkenler (i^* ve j^*) güncellenir (Adım 4). Bu değişim i. pozisyondaki iş $(n-1)$. pozisyona gelene kadar tekrarlanır (Adım 5 ve 6). Bu aşamadaki mevcut çözümde, i. ve $(i+1)$. pozisyonlarındaki işler sırasıyla $(n-1)$. ve n. pozisyonlardadır ve $(i+1,\ n]$ aralığındaki pozisyonlarda bulunan işler $[i,\ n-1)$ aralığındaki pozisyonlara aynı sırayla kaydırılmıştır.

n-insert + $(\pi) \rightarrow (\mathcal{N}_{k+}(\pi)|k \in \{1,2,\cdots,\omega\})$:

Adım 1: $T \leftarrow T^{\pi}$ ve $i \leftarrow 1$.

Adım 2: $j \leftarrow i+2$, $p \leftarrow \pi(i)$ ve $r \leftarrow \pi(i+1)$.

Adım 3: $\pi(j-2) \leftarrow \pi(j)$, $\pi(j-1) \leftarrow p$ ve $\pi(j) \leftarrow r$.

Adım 4: Eğer $T^{\pi} < T$ ise $T \leftarrow T^{\pi}$, $i^* \leftarrow i$ ve $j^* \leftarrow j$.

Adım 5: $j \leftarrow j+1$.

Adım 6: Eğer $j \leq n$ ise Adım 3' e git. Aksi halde $j \leftarrow n$.

Adım 7: $\pi(j) \leftarrow \pi(j-2)$ ve $j \leftarrow j-1$.

Adım 8: Eğer $j > i+1$ ise Adım 7' ye git. Aksi halde $\pi(i) \leftarrow p$ ve $\pi(i+1) \leftarrow r$.

Adım 9: $i \leftarrow i+1$. Eğer $i < n-1$ ise Adım 2' ye git.

Adım 10: Eğer $T < T^{\pi}$ ise $j \leftarrow i^*+2$, $p \leftarrow \pi(i^*)$ ve $r \leftarrow \pi(i^*+1)$. Aksi halde sonlandır.

Adım 11: $\pi(j-2) \leftarrow \pi(j)$ ve $j \leftarrow j+1$.

Adım 12: Eğer $j \leq j^*$ ise Adım 11' e git. Aksi halde $\pi(j^*-1) \leftarrow p$ ve $\pi(j^*) \leftarrow r$.

Şekil 6.13. İkili kaydırma komşuluğundaki en iyi çözüm operatörü

Mevcut çözümdeki bu değişimi geri almak için $[i,\ n-1)$ aralığındaki pozisyonlarda bulunan işler $(i+1,\ n]$ aralığındaki pozisyonlara aynı sırayla kaydırılır ve yedekleme değişkenlerinde (p ve r) saklanan değerler sırasıyla i. ve $(i+1)$. pozisyonlarındaki işler olarak atanır (Adım 6-8). Tüm bu işlemler $[1,\ n-1)$

aralığındaki her i. pozisyon için sırasıyla tekrarlanır (Adım 9). Belirlenen sistematikle oluşabilecek tüm komşu çözümler arasındaki en iyi çözüm mevcut çözümden daha iyi ise, i^*. ve $(i^* + 1)$. pozisyonlarındaki işler sırasıyla $(j^* - 1)$. ve j^*. pozisyonlara atanır. Beraberinde, $(i^* + 1, j^*]$ aralığındaki pozisyonlarda bulunan işler $[i^*, j^* - 1)$ aralığındaki pozisyonlara sırasıyla kaydırılır (Adım 10-12).

n-insert- $(\pi) \rightarrow (\mathcal{N}_{k-}(\pi) \mid k \in \{1,2,\cdots,\omega\})$:	
Adım 1:	$r \in \mathbb{R} \leftarrow [0,1)$ aralığında rasgele bir sayı ve $i \leftarrow \lfloor r(n-2) \rfloor + 1$.
Adım 2:	$r \in \mathbb{R} \leftarrow [0,1)$ aralığında rasgele bir sayı ve $j \leftarrow \lfloor r(n-i-1) \rfloor + i + 2$.
Adım 3:	$p \leftarrow \pi(i)$, $q \leftarrow \pi(i+1)$ ve $l \leftarrow i + 2$.
Adım 4:	$\pi(l-2) \leftarrow \pi(l)$ ve $l \leftarrow l + 1$.
Adım 5:	Eğer $l \leq j$ ise Adım 4 git. Aksi halde $\pi(j-1) \leftarrow p$ ve $\pi(j) \leftarrow q$.

Şekil 6.14. İkili kaydırma komşuluğundaki rasgele çözüm operatörü

n-Insert komşuluk kümesinden eşit olasılıkla rasgele bir çözüm belirlemek için n-insert- operatörü (Şekil 6.14) kullanılmaktadır. Bu operatörde ilk olarak, $[1, n-1)$ aralığındaki pozisyonlardan eşit olasılıkla rasgele i. pozisyon (Adım 1) ve $(i+1, n]$ aralığındaki pozisyonlardan eşit olasılıkla rasgele j. pozisyon (Adım 2) belirlenir. Devamında, i. ve $(i+1)$. pozisyonlarındaki işler iki farklı değişkene (p ve q) yedeklenip (Adım 3), $(i+1, j]$ aralığındaki pozisyonlarda bulunan işler $[i, j-1)$ aralığındaki pozisyonlara kaydırılır (Adım 4). Daha sonra, değişkenlerde saklanan (p ve q) değerler $(j-1)$. ve j. pozisyonlarına atanır (Adım 5).

6.1.4.4. Karıştırma

Karıştırma işlemi, mevcut çözümü bir alandan başka bir alana taşıyarak DKA' nın arama işlemini çözüm uzayının farklı bölgelerine yayar. Bu sayede algoritmanın

erken yakınsaması (lokal optimum çözüme takılma) engellenmiş olur. Karıştırma işleminin ne ölçüde yapılacağı, algoritmanın performansı açısından önemlidir. Eğer karıştırma işlemi çok fazla olursa, çözüm kalitesi düşebilir ve algoritmanın koşum zamanı artabilir. Tersi olduğu durumda da erken yakınsama olabilir ve olası iyi çözümler gözden kaçabilir.

Bu nedenle karıştırma işleminin iterasyon sayısına, Bölüm 7' de anlatılan bir takım deneyler sonucunda karar verilmiştir. DKA1' de, karıştırma işlemi SHK-1 operatörüyle yapılmıştır. SHK-1 operatörü, belli bir komşuluk yapısına göre sistematik bir yöntemle çözüm uzayının farklı alanlarına geçiş yapar. SHK-1 operatörü, Şekil 6.4' de verilen komşuluk yapılarına ait çözümlere ulaşmak için iki işin yer değişimi (Şekil 6.6), ikili yer değişimi (Şekil 6.8), geriye kaydırma (Şekil 6.10), ileriye kaydırma (Şekil 6.12) ve ikili kaydırma (Şekil 6.14) komşuluklarındaki rasgele çözüm operatörlerini kullanmaktadır.

SHK-1 $(\pi, \mathcal{N}_k, t) \rightarrow (\pi)$:

Adım 1: $l \leftarrow 1$.

Adım 2: $\pi \leftarrow \mathcal{N}_{k-}(\pi)$.

Adım 3: $l \leftarrow l + 1$.

Adım 4: Eğer $l \leq t$ ise Adım 2' ye git.

Şekil 6.15. Karıştırma operatörü-1

Şekil 6.15' de görüldüğü üzere, mevcut çözüm (π), kullanılacak komşuluk yapısı $(\mathcal{N}_k)$ ve iterasyon sayısı (t) SHK-1 algoritmasının bir girdisi olarak verilmiştir. Öncelikle, iterasyon sayacına (l) 1 değeri atanmaktadır (Adım 1). Mevcut çözümün $\mathcal{N}_k$ komşuluk yapısına uygun komşu çözüm kümesinden eşit olasılıkla rasgele bir çözüm $(\mathcal{N}_{k-}(\pi))$ mevcut çözüme (π) atanır (Adım 2). Bu atama işlemi iterasyon sayısı kadar tekrarlanır (Adım 3 ve 4).

6.1.4.5. Değişken Komşu İniş

İyileşme aşamasında yaygın olarak kullanılan DKİ operatörü, herhangi bir çözümü çeşitli komşuluk yapılarına ait çözüm kümeleriyle kıyasladığından dolayı global optimum çözüme ulaşma ihtimalini artırmaktadır. DKİ operatöründe, tanımlanmış komşuluktaki lokal optimum çözümü bulmak için uyarlanan yöntemler, YA' da olduğu gibi, en iyi gelişmeyi arama veya ilk gelişmeyi arama stratejilerini izlerler. Tanımlanan komşuluk yapılarında en iyi çözümü arayan operatörlerin işlem karmaşıklığı en fazla $O(n^2)$ olduğu için DKA1' de kullanılan DKİ operatöründe en iyi gelişmeyi arama stratejisi dikkate alınmıştır. Bunun dışında, kullanılacak komşuluk yapılarına ve bu komşulukların sırasına, Bölüm 7' de anlatılan bir takım deneyler sonucunda karar verilmiştir.

Şekil 6.16' da görüldüğü üzere, mevcut çözüm (π), komşuluk yapılarını belirli bir sırayla içeren küme ($\mathcal{N}$) ve komşuluk sayısı (ω) DKİ algoritmasının bir girdisi olarak verilmiştir. Öncelikle, komşuluk sayacına (k) 1 değeri atanmaktadır (Adım 1). Mevcut çözümün $\mathcal{N}_k$ komşuluk yapısına uygun komşu çözüm kümesindeki en iyi çözüm ($\mathcal{N}_{k+}(\pi)$) bir dizi değişkenine ($\ddot{\pi}$) atanır (Adım 2). Eğer dizi değişkenindeki çözüm mevcut çözümden daha iyi ise, mevcut çözüm dizi değişkenindeki çözümle güncellenir ve 1. komşuluk yapısından başlayarak aynı işlemler tekrarlanır. Aksi halde, bir sonraki komşuluk yapısına geçilerek aynı işlemler tekrarlanır (Adım 3). Son komşuluk yapısına ($\mathcal{N}_\omega$) uygun komşu çözümlerde de gelişme olmaz ise algoritma sonlanır (Adım 4).

DKİ (π, ω, $\mathcal{N}$) → (π):	
Adım	1: $k \leftarrow 1$.
Adım	2: $\ddot{\pi} \leftarrow \mathcal{N}_{k+}(\pi)$.
Adım	3: Eğer $T^{\ddot{\pi}} < T^{\pi}$ ise $\pi \leftarrow \ddot{\pi}$ ve Adım 1' e git. Aksi halde $k \leftarrow k + 1$.
Adım	4: Eğer $k \leq \omega$ ise Adım 2' ye git.

Şekil 6.16. Değişken komşu iniş operatörü

6.1.4.6. Algoritmayı Sonlandırma

Algoritmayı sonlandırma işlemi, iyileşme olmayan ardışık iterasyon miktarına göre ayarlanmıştır. Bunun için bir sayaç kullanılmıştır. Herhangi bir iterasyonda, iyileşme olduğu zaman sayaç sıfırlanır, aksi halde, belirlenen değere ulaştığında, algoritma sonlandırılır.

6.1.5. Değişken Komşu Arama Algoritması-2

DKA2' de karıştırma işlemi, DKA1' den farklı yapılmıştır. DKA1' deki karıştırma işlemi, algoritmanın sadece erken yakınsamasını önlemektedir. Buna ilaveten DKA2' deki karıştırma operatöründe, baskınlık kuralları kullanılarak herhangi bir iyileşme olup olmadığına da bakılmıştır. Bunun dışında, DKA1' in tüm yapısı aynı şekilde DKA2' de de kullanılmıştır.

SHK-2 $(\pi, \mathcal{N}_k, t) \rightarrow (\pi)$:

Adım 1: $l \leftarrow n$.

Adım 2: $i \leftarrow \pi(l)$, $p \leftarrow l - 1$ ve $j \leftarrow \pi(p)$.

Adım 3: Eğer önerme 6.1 i ve j için sağlanıyor ise Adım 6' ya git.

Adım 4: $p \leftarrow p - 1$. Eğer $p > 0$ ise $j \leftarrow \pi(p)$. Aksi halde Adım 7' ye git.

Adım 5: Eğer önerme 6.2 i ve j için sağlanmıyor ise Adım 4' e git.

Adım 6: $\pi(l) \leftarrow j$, $\pi(p) \leftarrow i$ ve Adım 2' ye git.

Adım 7: $l \leftarrow l - 1$. Eğer $l > 1$ ise Adım 2' ye git.

Adım 8: SHK-1 $(\pi, \mathcal{N}_k, t) \rightarrow (\pi)$.

Şekil 6.17. Karıştırma operatörü-2

Şekil 6.17' de görüldüğü üzere, $(1, n]$ aralığındaki pozisyonlarda bulunan her işin (i) $[1, i)$ pozisyonlarındaki tüm işler (j) ile baskınlık durumu incelenmiştir. İlk olarak komşu iki iş arasındaki baskınlık önerme 6.1' e göre değerlendirilmiştir (Adım 1-3).

Bu değerlendirme sonucunda baskınlık tespit edilmez ise, mevcut çizelge $\pi = \{\pi_s, j, i, \pi_e\}$ değişmeden kalır (Adım 3) ve j işi bir önceki pozisyonda bulunan iş ile güncellenir (Adım 4). Aksi halde, baskınlık tespit edilir ise, mevcut çizelge $\pi = \{\pi_s, i, j, \pi_e\}$ olarak güncellenir ve aynı işlemler bu yeni çizelge için tekrarlanır (Adım 6). Adım 4' de güncellenen j ve mevcut i işleri artık komşu işler değildir. Dolayısıyla, baskınlık önerme 6.2' ye göre değerlendirilmiştir (Adım 5). Bu değerlendirme sonucunda baskınlık tespit edilmez ise, mevcut çizelge $\pi = \{\pi_s, j, \pi_m, i, \pi_e\}$ değişmeden kalır (Adım 5) ve j işi bir önceki pozisyonda bulunan iş ile güncellenir (Adım 4). Aksi halde, baskınlık tespit edilir ise, mevcut çizelge $\pi = \{\pi_s, i, \pi_m, j, \pi_e\}$ olarak güncellenir ve aynı işlemler bu yeni çizelge için tekrarlanır (Adım 6). Her $i = \pi(l),\ l = \{2,3,\cdots,n\}$ işi ile tüm $j = \pi(p),\ p = \{1,2,\cdots,l-1\}$ işleri arasındaki baskınlık değerlendirmesi tamamlandığında (Adım 7) muhtemel gelişmeler kontrol edilmiş ve tamamlanmış olur. Son olarak, SHK-1 operatörü vasıtasıyla karıştırma işlemi tamamlanır (Adım 8).

6.2. $1|LT + F, s_j|C_{max}$ Problemine Ait Çözüm Yöntemleri

Literatürde bulunan $1|L|C_{max}$ problemi [21, 100], $1|L + F|C_{max}$ problemi [124, 144, 204], $1|s_j|C_{max}$ problemi [205, 206] ve $1|L, s_j|C_{max}$ [10] problemi kolay problemler sınıfındadır ve işlem süresi $O(n \log n)$ olan SPT yöntemi ile optimize edilebilmektedir. Tanımlanan $1|LT + F, s_j|C_{max}$ problemi BZAS, iş üzerinde geçen zamana bağlı öğrenme etkisi ve ayar süresine bağlı unutma etkisi içermektedir. Tanımlanan bu problem için aşağıdaki Teorem 6.1 ve 6.2' de optimum bir çizelgenin bazı genel özellikleri anlatılmış olsa da, kesin çözüm için bir PZ tespit edilememiştir. Yaygın olarak bilinen ve bunlardan türetilebilen PZ' lar, tanımlanan bu problem için yetersiz kaldığından dolayı, bu incelemenin yerinde bir çaba olup olmadığı, ancak problemin çözüm karmaşıklığı tespit edilerek belirlenebilir [31]. Bu nedenle, ilgili problemin çözüm karmaşıklığı ve karmaşıklık neticesinde önerilen kesin çözüm yöntemleri ilerleyen alt bölümlerde anlatılmıştır. Bölüm 6.2.2' de, tanımlanan problemden türetilen karar probleminin NP-tam olduğu gösterilmiştir. Dolayısıyla, bu problemin etkin (polinom zamanlı) bir yöntemle kesin çözümü mümkün değildir.

Buradan hareketle, $1|LT+F,s_j|C_{max}$ problemi için kesin çözüm yöntemi olarak Bölüm 6.2.3' de TDOP ve Bölüm 6.2.4' de DP geliştirilmiştir. Ayrıca, önerilen DP sözde-polinom zamanda çözüm üretebildiği için $1|LT+F,s_j|C_{max}$ probleminin normal şiddette NP-zor olduğu ıspatlanmıştır. Yaklaşık çözüm elde etmek için SPT kuralı ve Bölüm 6.1' de anlatılan metasezgisel yöntemler kullanılabilir.

6.2.1. $1|LT+F,s_j|C_{max}$ Probleminin Optimalite Özellikleri

Bu bölümde, global optimum bir çizelgenin $\pi=\{\pi',\pi''\}$ bazı özelliklerine değineceğiz. İlk olarak, tanımlanan probleme ait teoremlerin anlatımını sadeleştirmek için aşağıdaki yardımcı önerme verilmiştir.

Yardımcı Önerme 6.1. Verilen $x\geq 0$, $t\geq 0$ ve $\lambda\geq 1$ için $\phi(\lambda,t)=\lambda f(x+t)-\lambda f(x)-f(x+\lambda t)+f(x)\geq 0$ olmaktadır.

Ispat. Fonksiyon $\phi(\lambda,t)$' nin λ ve t' ye göre 1. türevini aldığımızda,

$$\phi'(\lambda,t)=\begin{cases}\partial\phi(\lambda,t)/\partial\lambda=\phi_x(\lambda,t)=f(x+t)-f(x)-tf'(x+\lambda t)\\ \partial\phi(\lambda,t)/\partial t=\phi_y(\lambda,t)=\lambda f'(x+t)-\lambda f'(x+\lambda t)\end{cases} \quad (6.51)$$

olmaktadır. Tanımdan bilindiği üzere $f'(x)$ azalan fonksiyon olduğu için $f'(x+t)-f'(x+\lambda t)\geq 0$ ve dolayısıyla, $\phi_y(\lambda,t)\geq 0$ olur.

Diğer kısmi türev $\phi_x(\lambda,t)$' nin analizi için 'Orta Değer Teoremi (ODT)' kullanılabilir. ODT' ne göre denklem 6.52' de gösterilen ifade için $x\leq\delta\leq x+t$ aralığında bir δ değeri bulunmaktadır.

$$f(x+t)-f(x)=tf'(\delta) \quad (6.52)$$

Buradan yola çıkarak, $\phi_x(\lambda,t)=tf'(\delta)-tf'(x+\lambda t)=t[f'(\delta)-f'(x+\lambda t)]\geq 0$ olur, çünkü $\delta\leq x+t\leq x+\lambda t$ ve $f'(x)$ azalan fonksiyondur. Sonuç olarak,

$\phi'(\lambda, t) \geq 0$ olduğundan dolayı, $\phi(\lambda, t)$ artan fonksiyondur. Değişkenlerin alabileceği en küçük değerler $t = 0$ ve $\lambda = 1$ için $\phi(1,0) = f(x) - f(x) - f(x) + f(x) = 0$ olur ve böylece, ıspat tamamlanmış olur. ■

Teorem 6.1. Tanımlanan $1|LT + F, s_j|C_{max}$ problemine ait $\pi = \{\pi', \pi''\}$ optimum çizelgesindeki π' ve π'' alt çizelgelerinin her ikisi de SPT sıralamasıdır.

Ispat. Aykırılıktan yola çıkarak bu teorem ıspatlanmıştır. SPT kuralını ihlal eden en az iki işin bulunduğu π_1 çizelgesinin optimum olduğunu varsayalım. Bu çizelgede $p_j \geq p_i$' dir ve SPT kuralını ihlal ettiği için j işi i işinden önce çizelgelenmiştir. Bu i ve j işleri arasında yer değişimi yapılarak elde edilen çizelge π_2 olsun. Bu şekilde i ve j işleri için 2 farklı durumdan söz edilebilir.

İlk durumda, i ve j işleri π' alt çizelgesindedir, dolayısıyla $\pi_1 = \{\dot{\pi}, j, i, \ddot{\pi}, \pi''\}$ ve $\pi_2 = \{\dot{\pi}, i, j, \ddot{\pi}, \pi''\}$ olmaktadır. Böylece π_1 çizelgesinde, $\dot{\pi}$ ve $\ddot{\pi}$ sırasıyla j işinden önceki ve i işinden sonraki işlerin oluşturduğu alt çizelgelerdir (π_2 çizelgesinde bunun tersi geçerlidir) ve $x = \sum_{k\in\dot{\pi}} p_k$ olduğu düşünülürse

$$C_{max}^{\pi_1} = \sum_{k\in\pi_1} s_k + \sum_{k\in\dot{\pi}} p_k^A + \sum_{k\in\{\ddot{\pi},\pi''\}} p_k^A + p_j[1 - f(x)] + p_i\big[1 - f\big(x + p_j\big)\big] \quad (6.52)$$

$$C_{max}^{\pi_2} = \sum_{k\in\pi_2} s_k + \sum_{k\in\dot{\pi}} p_k^A + \sum_{k\in\{\ddot{\pi},\pi''\}} p_k^A + p_i[1 - f(x)] + p_j[1 - f(x + p_i)] \quad (6.53)$$

olur. Denklem 6.52 ve 6.53' den yola çıkarak $\Delta C_{max}^{\pi_1-\pi_2} = C_{max}^{\pi_1} - C_{max}^{\pi_2}$ denklem 6.54' de gösterildiği gibi olur.

$$\begin{aligned}\Delta C_{max}^{\pi_1-\pi_2} &= p_j[1 - f(x)] + p_i\big[1 - f\big(x + p_j\big)\big] - p_i[1 - f(x)] - p_j[1 - f(x + p_i)] \\ &= p_j[f(x + p_i) - f(x)] - p_i\big[f\big(x + p_j\big) - f(x)\big] \\ &= p_i\big[\big(p_j/p_i\big)\big(f(x + p_i) - f(x)\big) - f\big(x + p_j\big) + f(x)\big] \quad (6.54)\end{aligned}$$

Devamında, denklem 6.54' deki $(p_j/p_i)\big(f(x+p_i)-f(x)\big)-f\big(x+p_j\big)+f(x)$ ifadesinde p_j/p_i ve p_i yerine sırasıyla λ ve t yazıldığında, $\Delta C_{max}^{\pi_1-\pi_2}=p_i\phi(\lambda,t)$ olur. Bu durumda, $\phi(\lambda,t)\geq 0$ (Yardımcı Önerme 6.1) olduğu için $\Delta C_{max}^{\pi_1-\pi_2}\geq 0$ olur ve bu durum π_1 çizelgesinin optimalliğine aykırıdır.

İkinci durumda, i ve j işleri π'' alt çizelgesindedir, dolayısıyla $\pi_1=\{\pi',\dot{\pi},j,i,\ddot{\pi}\}$ ve $\pi_2=\{\pi',\dot{\pi},i,j,\ddot{\pi}\}$ olmaktadır. İlk duruma benzer şekilde $x=\sum_{k\in\{\pi',\dot{\pi}\}}p_k$ olduğu düşünülürse

$$C_{max}^{\pi_1} = \sum_{k\in\pi_1}s_k+\sum_{k\in\{\pi',\dot{\pi}\}}p_k^A+\sum_{k\in\ddot{\pi}}p_k^A+p_j\big[1-h\big(x,s_j\big)\big]+ p_i\big[1-h\big(x+p_j,s_i\big)\big] \quad (6.55)$$

$$C_{max}^{\pi_2} = \sum_{k\in\pi_2}s_k+\sum_{k\in\{\pi',\dot{\pi}\}}p_k^A+\sum_{k\in\ddot{\pi}}p_k^A+p_i[1-h(x,s_i)]+ p_j\big[1-h\big(x+p_i,s_j\big)\big] \quad (6.56)$$

olur. Denklem 6.55 ve 6.56' dan yola çıkarak $\Delta C_{max}^{\pi_1-\pi_2}$ denklem 6.54' de gösterildiği gibi olur. Görüldüğü üzere, bu durum da π_1 çizelgesinin optimalliğine aykırıdır. Sonuç olarak, optimum bir çizelgede π' ve π'' alt çizelgelerinin her ikisi de SPT kuralına uygun olmalıdır ve ıspat böylece tamamlanmıştır. ■

Doğal Sonuç 6.1. Tanımlanan $1\big|LT+F,s_j\big|C_{max}$ probleminde eğer $\forall j\in N$ için $s_j=0$ ise SPT sıralaması optimumdur.

Teorem 6.2. Tanımlanan $1\big|LT+F,s_j\big|C_{max}$ probleminde eğer $\pi'=\{\hat{\pi},j\}$ olan bir $\pi=\{\pi',\pi''\}$ çizelgesi optimum ise $\forall i\in\pi''$ için $\hat{P}+p_i>k_0$ olur.

Ispat. Optimum olan $\pi_1=\{\hat{\pi},j,i,\ddot{\pi}\}$ çizelgesinin $\pi'=\{\hat{\pi},j\}$ ve $\pi''=\{i,\ddot{\pi}\}$ alt çizelgelerinden oluştuğunu varsayalım. Böylece, teorem 6.1' den yola çıkarak π' ve π'' alt çizelgelerinin SPT kuralına uygun olduğu ve $p_i=\min_{k\in\pi''}\{p_k\}$ olduğu rahatlıkla söylenebilir. Bu nedenle, teoremin ıspatı için sadece $i\in\pi''$ işini dikkate almak yeterlidir.

Aykırılığa dayanarak, i işi için $\hat{P} + p_i \leq k_0$ olduğunu düşünelim. Böylece, $p_i < p_j$ olduğu gayet açıktır. Anlatımı kolaylaştırmak için $\pi_T = \{\hat{\pi}, i, j\}$ ve $\pi_R = \{\hat{\pi}, j, i\}$ olan iki alt çizelgeye yoğunlaşalım. Buna göre, π_T çizelgesinin $\pi_T' = \{\hat{\pi}, i, j\}$ ve $\pi_T'' = \emptyset$ alt çizelgelerinden, π_R çizelgesinin de $\pi_R' = \{\hat{\pi}, j\}$ ve $\pi_R'' = \{i\}$ alt çizelgelerinden oluştuğu rahatlıkla söylenebilir. Eğer $x = \sum_{k\in\hat{\pi}} p_k$ kabul edersek, her iki çizelge için maksimum tamamlanma zamanları

$$C_{max}^{\pi_T} = \sum_{k\in\pi_T} s_k + \sum_{k\in\hat{\pi}} p_k^A + p_i[1 - f(x)] + p_j[1 - f(x + p_i)] \tag{6.57}$$

$$C_{max}^{\pi_R} = \sum_{k\in\pi_R} s_k + \sum_{k\in\hat{\pi}} p_k^A + p_j[1 - f(x)] + p_i\big[1 - h\big(x + p_j, s_i\big)\big] \tag{6.58}$$

olur. Denklem 6.57 ve 6.58' den yola çıkarak, $\Delta C_{max}^{\pi_T - \pi_R}$ denklem 6.59' da gösterildiği gibi olur.

$$\begin{aligned}
\Delta C_{max}^{\pi_T - \pi_R} &= p_i\Big(h\big(x + p_j, s_i\big) - f(x)\Big) - p_j\big(f(x + p_i) - f(x)\big) \\
&= p_i\Big(f\big(x + p_j\big) - f(x_0)\mathrm{g}(s_i) - f(x)\Big) - p_j\big(f(x + p_i) - f(x)\big) \\
&= -\Big(p_j\big(f(x + p_i) - f(x)\big) - p_i\Big(f\big(x + p_j\big) - f(x)\Big) + \\
&\quad p_i f(x_0)\mathrm{g}(s_i)\Big) \\
&= -p_i\Big(\big(p_j/p_i\big)\big(f(x + p_i) - f(x)\big) - f\big(x + p_j\big) + f(x) + \\
&\quad f(x_0)\mathrm{g}(s_i)\Big)
\end{aligned} \tag{6.59}$$

Daha sonra, denklem 6.59' daki $\big(p_j/p_i\big)\big(f(x + p_i) - f(x)\big) - f\big(x + p_j\big) + f(x) + f(x_0)\mathrm{g}(s_i)$ ifadesinde p_j/p_i ve p_i yerine sırasıyla λ ve t yazıldığında, $\Delta C_{max}^{\pi_T - \pi_R} = -p_i\big(\phi(\lambda, t) + f(x_0)\mathrm{g}(s_i)\big)$ olur. Böylece, $f(x_0)\mathrm{g}(s_i) \geq 0$ ve $\phi(\lambda, t) \geq 0$ (Yardımcı Önerme 6.1) olduğundan $\Delta C_{max}^{\pi_T - \pi_R} \leq 0$ olur. Eğer π_1 çizelgesindeki i ve j işleri arasında yer değişimi yapılırsa $\pi_2 = \{\hat{\pi}, i, j, \ddot{\pi}\}$ çizelgesi elde edilir. Bu durumda, $\pi_1 = \{\pi_R, \ddot{\pi}\}$ ve $\pi_2 = \{\pi_T, \ddot{\pi}\}$ olduğu için $C_{max}^{\pi_2} \leq C_{max}^{\pi_1}$ olur. Görüldüğü üzere, bu durum π_1 çizelgesinin optimalliğine aykırıdır ve ıspat böylece tamamlanmıştır. ■

6.2.2. $1|LT + F, s_j|C_{max}$ Probleminin Çözüm Karmaşıklığı

Bu bölümde, $1|LT + F, s_j|C_{max}$ probleminin çözüm karmaşıklığı incelenmiştir. Tanımlanan problemin NP-zor olup olmadığını ıspatlamak için NP-tam olan 'Eşit Elemanlı Bölünme' (Equal Cardinality Partition) [31] probleminden $1|LT + F, s_j|C_{max}$ probleminin karar versiyonuna polinom zamanlı bir dönüşüm tanımlanmıştır. Dönüşümle ilgili detaylar aşağıda verilmiştir.

Eşit Elemanlı Bölünme (Prob-1): Pozitif tamsayılardan oluşan $V = \{a_1, a_2, \cdots, a_{2m}\}$ kümesi $2m$ boyutludur ve B pozitif bir tamsayı olmak üzere $\sum_{i \in V} a_i = 2B$' dir. Verilen bu V kümesi, $\sum_{i \in V_1} a_i = \sum_{i \in V_2} a_i = B$ ve $|V_1| = |V_2| = m$ eşitliklerini sağlayan V_1 ve V_2 alt kümelerine ayrılabilir mi?

$1|LT + F, s_j|C_{max}$ probleminin karar versiyonu (Prob-2): $1|LT + F, s_j|C_{max}$ probleminin özellikleri ve kısıtları altında tek makinede işlem görecek olan bir takım işlerin, verilen bir C değerine göre, $C_{max}^{\pi} \leq C$ olan bir π çizelgesi meydana getirmeleri mümkün müdür?

Görüldüğü üzere, dönüşüm için tanımlanan problemlerin her ikisi de karar problemidir ve her iki problemin de cevabı 'evet' ya da 'hayır' olacak şekilde sadece 2 ihtimallidir. Prob-2' nin $1|LT + F, s_j|C_{max}$ probleminden tek farkı, amaç fonksiyonu için bir üst sınır belirlenmiş olmasıdır. $1|LT + F, s_j|C_{max}$ probleminde optimum sonuç aranırken, Prob-2' de sadece C değerine eşit veya daha küçük bir değer aranmaktadır. Bu nedenle, Prob-1 için verilmiş olan girdileri kullanan sistematik bir ilişki tanımlayarak, Prob-2 için gerekli olan tüm girdileri oluşturmak gerekmektedir. Bu çerçevede, NP-tam olan Prob-1' den çözüm karmaşıklığı henüz bilinmeyen Prob-2' ye polinom zamanlı bir dönüşüm tanımlanmıştır.

Tanımlanan dönüşüme göre;

$$|N| = n = 2m + 1, \qquad k_0 = mB + B, \qquad x_0 = \min_{j \in N}\{p_j\}$$

sırasıyla, Prob-2 için tasarlanan örneğe ait iş sayısı, eşik değeri ve normalizasyon değeridir. Öğrenme, unutma ve bağıl fonksiyonlar aşağıdaki gibi düzenlenmiştir.

$$f(x) = x/(x+1), \qquad \mathrm{g}(y) = y/(y+1), \qquad h(x,y) = f(x) - f(x_0)\mathrm{g}(y).$$

Oluşturulan Prob-2 örneğinde, işlem süreleri ve ayar süreleri $j = 1,2,\cdots,2m$ için

$$p_j = B + a_j; \quad s_j = B + a_j$$

olan $2m$ adet iş tanımlanmıştır. Bu işler, atanan değerler nedeniyle 'bölünme işleri' olarak adlandırılmıştır. Ayrıca, bölünme işlerinin iki eşit parçaya ayrılmalarını sağlayacak bir iş daha tanımlanmıştır ve bu iş 'zorlayıcı iş' J_e olarak adlandırılmıştır. Zorlayıcı işin işlem ve ayar süreleri sırasıyla;

$$p_e = 2B(mB + B + 1)(mB + B)^2, \qquad s_e = 2(mB + B + 1)(mB + B)^2 - 1$$

ve son olarak, karar değeri;

$$C = 2(mB + 2B + 1)(mB + B)^2 + 4mB + 4B - 1.$$

olarak uyarlanmıştır. Görüldüğü üzere, tanımlanan bu dönüşüm polinom zamanda $O(2m)$ yapılabilir. Aşağıda, dönüşümle elde edilen bu Prob-2 örneğinin optimum çizelgesini tanımlayan bir takım özellikler anlatılmıştır. Bu özellikler, Prob-1 ile Prob-2 arasındaki ilişkiyi ıspatlayan temel bilgilerdir.

Özellik 6.1. Oluşturulan Prob-2 örneğinin optimum çizelgesi $\pi = \{\pi', \pi''\}$ ise; (a) J_e zorlayıcı iş π' alt çizelgesinin son pozisyonunda yani, $\pi' = \{\hat{\pi}, J_e\}$ ve (b) $\hat{P} = \sum_{j\in\hat{\pi}} p_j = mB + B$ olmalıdır.

Ispat. (a) Teorem 6.1' e göre, Prob-2 örneğinin optimum çizelgesinde zorlayıcı iş ya π' ya da π'' alt çizelgelerinden birinin son pozisyonunda olmalıdır. Bu nedenle, sadece bu iki durumu incelemek yeterlidir. Genelliği kaybetmeden, zorlayıcı işin π' alt çizelgesinin son pozisyonunda olduğu durumu temsilen π_1 çizelgesinin, en küçük

maksimum tamamlanma zamanına (denklem 6.60) sahip çizelge olduğunu varsayalım. Bu durumda, π_1 çizelgesinin tanımlanan optimum çizelgeye eşit olduğunu da kabul etmiş oluruz.

$$C_{max}^{\pi_1} = s_e + \sum_{j=1}^{2m} s_j + \sum_{j\in\hat{\pi}_1} p_j^A + p_e^A + \sum_{j\in\pi_1''} p_j^A \tag{6.60}$$

Buradan, $\sum_{j\in\hat{\pi}_1} p_j^A + \sum_{j\in\pi_1''} p_j^A < \sum_{j=1}^{2m} p_j = 2mB + 2B$ olduğu için denklem 6.60' daki eşitlik denklem 6.61' deki eşitsizliğe dönüşür.

$$C_{max}^{\pi_1} < s_e + \sum_{j=1}^{2m} s_j + 2mB + 2B + p_e^A \tag{6.61}$$

Eşik değeri $k_0 = mB + B$ ve $\hat{P}_1 = mB + B$ olduğu için $p_e^A = p_e\big(1 - f(mB + B)\big)$ ve buradan $p_e^A = 2B(mB + B + 1)(mB + B)^2\big(1 - (mB + B/mB + B + 1)\big) = 2B(mB + B)^2$ olur. Böylece, denklem 6.62' de belirtildiği üzere $C_{max}^{\pi_1} < C$ olur.

$$\begin{aligned} C_{max}^{\pi_1} < {} & 2(mB + B + 1)(mB + B)^2 - 1 \\ & +4mB + 4B + 2B(mB + B)^2 = C \end{aligned} \tag{6.62}$$

Genelliği kaybetmeden, zorlayıcı işin π'' alt çizelgesinin son pozisyonunda olduğu durumu temsilen π_2 çizelgesinin, en küçük maksimum tamamlanma zamanına (denklem 6.63) sahip çizelge olduğunu varsayalım.

$$C_{max}^{\pi_2} = s_e + \sum_{j=1}^{2m} s_j + \sum_{j=1}^{2m} p_{[j]}^A + p_e^A \tag{6.63}$$

Buradan, $\sum_{j=1}^{2m} p_{[j]}^A$ ifadesini denklem 6.63' den çıkardığımızda denklem 6.64' deki eşitsizlik elde edilir.

$$C_{max}^{\pi_2} > s_e + \sum_{j=1}^{2m} s_j + p_e^A \tag{6.64}$$

Zorlayıcı iş çizelgenin son işi olduğu için $p_e^A = p_e\big(1 - h(2mB + 2B, s_e)\big)$ ve buradan yola çıkarak,

$$p_e^A = 2B(mB+B+1)(mB+B)^2\left(1-\frac{2mB+2B}{2mB+2B+1}+\frac{x_0}{x_0+1}\times\frac{s_e}{s_e+1}\right),$$

$$> 2B(mB+B+1)(mB+B)^2\left(1-\frac{2mB+2B}{2mB+2B+1}+\frac{1}{B}\times\frac{s_e}{s_e+1}\right),$$

$$> 2B(mB+B+1)(mB+B)^2\left(\frac{2(mB+B+1)(mB+B)^2-1}{2B(mB+B+1)(mB+B)^2}\right),$$

$$> 2(mB+B+1)(mB+B)^2-1$$

olur. Neticede, denklem 6.64' deki ifade denklem 6.65' de verildiği gibi olur.

$$C_{max}^{\pi_2} > 4(mB+B+1)(mB+B)^2+2mB+2B-2>C \tag{6.65}$$

Denklem 6.62 ve 6.65 ile belirtilenlere göre, $C_{max}^{\pi_1} < C_{max}^{\pi_2}$ ve dolayısıyla, Prob-2 örneğinin optimum çizelgesinde zorlayıcı iş π' alt çizelgesinin son pozisyonunda olmalıdır. Böyle, ıspatın (a) kısmı tamamlanmıştır.

(b) Bu kısma ait ıspatta, sadece $\hat{P} \leq mB+B$ durumunu incelemek yeterli olacaktır, çünkü $\hat{P} > mB+B$ durumunda zorlayıcı işin π'' alt çizelgesinin son pozisyonunda olması gerekir. Ayrıca, $\hat{P} > mB-B$ olmalıdır, çünkü $\hat{P} \leq mB-B$ durumunda herhangi bir $j \in \pi''$ işi için $mB-B+p_j = mB+a_j < mB+B$ olacağından, bu durum Teorem 6.2' yi ihlal edecektir. Genelliği kaybetmeden, $\hat{P} = mB+B-\Delta$ ve $1 \leq \Delta < 2B$ şartlarına uygun π_3 çizelgesinin, en küçük maksimum tamamlanma zamanına (denklem 6.66) sahip çizelge olduğunu varsayalım.

$$C_{max}^{\pi_3} = s_e + \sum_{j=1}^{2m} s_j + \sum_{j\in\hat{\pi}_3} p_j^A + p_e^A + \sum_{j\in\pi_3''} p_j^A \tag{6.66}$$

Buradan, $\sum_{j\in\hat{\pi}_3} p_j^A + \sum_{j\in\pi_3''} p_j^A$ ifadesini denklem 6.66' dan çıkardığımızda denklem 6.67' deki eşitsizlik elde edilir.

$$C_{max}^{\pi_3} > s_e + \sum_{j=1}^{2m} s_j + p_e^A \tag{6.67}$$

Eşik değeri $k_0 = mB + B$ ve $\hat{P}_3 = mB + B - \Delta$ olduğu için $p_e^A = p_e\big(1 - f(mB + B - \Delta)\big)$ ve buradan

$$C_{max}^{\pi_3} > 2(mB + B + 1)(mB + B)^2 - 1 + 2mB + 2B + p_e\big(1 - f(mB + B - \Delta)\big) \quad (6.68)$$

olur. Denklem 6.68' in sağ tarafındaki ifadeyi $\vartheta(\Delta) = 2(mB + B + 1)(mB + B)^2 - 1 + 2mB + 2B + p_e\big(1 - f(mB + B - \Delta)\big)$ fonksiyonu olarak tanımlayalım. Bu durumda, $f'(x) > 0$ ve $mB + B - \Delta > 0$ olduğundan dolayı, fonksiyonun 1. türevi $\vartheta'(\Delta) = p_e f'(mB + B - \Delta) > 0$ olur. Bunun neticesinde, denklem 6.68' in sağ tarafındaki ifade Δ' ya göre artan fonksiyondur ve dolayısıyla, sadece $\Delta = 1$ durumunu dikkate aldığımızda, denklem 6.68' in sağ tarafındaki ifadeyle birlikte $C_{max}^{\pi_3}$ değerini de minimize etmiş oluruz. Buradan hareketle, $p_e^A = 2B(mB + B + 1)(mB + B)^2\big(1 - (mB + B - 1/mB + B)\big) = 2B(mB + B + 1)(mB + B)$ olur. Böylece, $C_{max}^{\pi_3} > C$ olur (denklem 6.69) ve ıspatın (b) kısmı tamamlanmıştır.

$$C_{max}^{\pi_3} > 2(mB + B + 1)(mB + B)(mB + 2B) + 2mB + 2B - 1 > C \quad (6.69)$$

Sonuç olarak, özellik 6.1' de belirtilen her iki (a) ve (b) şartları da sağlanmış ve ıspat tamamlanmıştır. ■

Özellik 6.2. Oluşturulan Prob-2 örneğinin optimum çizelgesi $\pi = \{\pi', \pi''\}$ ve $\pi' = \{\hat{\pi}, J_e\}$ ise; π' alt çizelgesindeki bölünme işlerinin sayısı m olmalıdır, yani $|\hat{\pi}| = m$ olmalıdır.

Ispat. Özellik 6.1' den yola çıkarak, Prob-2 örneğinin optimum çizelgesinde $\hat{P} = \sum_{i \in \hat{\pi}} p_i = mB + B$ olmalıdır. Buradan hareketle, $|\hat{\pi}| = m$ ıspatı için $|\hat{\pi}| = m - 1$ ve $|\hat{\pi}| = m + 1$ durumlarını incelemek yeterli olacaktır.

İlk olarak, $|\hat{\pi}| = m - 1$ olduğunu varsayalım. Bu durumda, $\hat{P} = \sum_{i=1}^{m-1} B + a_i = B(m - 1) + \sum_{i=1}^{m-1} a_i < B(m - 1) + 2B = mB + B$, yani $\hat{P} < mB + B$ olur. İkinci olarak, $|\hat{\pi}| = m + 1$ olduğunu varsayalım. Bu durumda, $\hat{P} = \sum_{i=1}^{m+1} B + a_i =$

$B(m+1) + \sum_{i=1}^{m+1} a_i > mB + B$, yani $\hat{P} > mB + B$ olur. Sonuç olarak, $\hat{P} = mB + B$ durumu sadece $|\hat{\pi}| = m$ için geçerlidir ve ıspat tamamlanmıştır. ■

Teorem 6.3. Tanımlanan $1|LT + F, s_j|C_{max}$ bir NP-zor problemdir.

Ispat. Prob-1' e ait verilen bir problem örneğinden, Prob-2' ye ait bir problem örneği polinom zamanda oluşturulmuştur. Özellik 6.1 ve 6.2' ye dayanarak, aşağıdaki çift yönlü önermeyi test edebiliriz:

"**Ancak ve ancak** Prob-1' in cevabı 'evet' ise, Prob-2' nin cevabı 'evet', yani $C_{max}^{\pi} \leq C$ olur."

"Ancak" Tanımlanan V_1 ve V_2 alt kümelerinin Prob-1 için çözüm ürettiğini varsayalım. Bu durumda, $\sum_{i \in V_1} a_i = \sum_{i \in V_2} a_i = B$ ve $|V_1| = |V_2| = m$ olur. Bu çözüme dayanarak, özellik 6.1 ve 6.2' ye göre $\hat{\pi} = \{J_j | a_j \in V_1\}$, $\pi'' = \{J_j | a_j \in V_2\}$ ve $\pi' = \{\hat{\pi}, J_e\}$ olan bir çizelge $\pi = \{\pi', \pi''\}$ meydana getirdiğimizde, $|\hat{\pi}| = |\pi''| = m$ ve $\hat{P} = mB + B$ olur. Bu çizelge için $C_{max}^{\pi} \leq C$ olduğundan dolayı, Prob-1' e çözüm oluşturan bir bölünme Prob-2' ye de çözüm oluşturur.

"ve Ancak" Tanımlanan π çizelgesinin Prob-2 için bir çözüm, yani $C_{max}^{\pi} \leq C$ olduğunu varsayalım. Bu durumda, özellik 6.1 ve 6.2' ye göre, $\pi = \{\pi', \pi''\}$, $\pi' = \{\hat{\pi}, J_e\}$, $|\hat{\pi}| = |\pi''| = m$ ve $\hat{P} = mB + B$ olur. Bu çözüme dayanarak, $V_1 = \{a_j | J_j \in \hat{\pi}\}$ ve $V_2 = \{a_j | J_j \in \pi''\}$ alt kümelerini meydana getirdiğimizde, $\sum_{i \in V_1} a_i = \sum_{i \in V_2} a_i = B$ ve $|V_1| = |V_2| = m$ olur. Görüldüğü üzere, bu bölünme Prob-1 için çözümdür. Dolayısıyla, Prob-2' ye çözüm oluşturan bir çizelge Prob-1' e de çözüm oluşturur.

Sonuç olarak, belirtilen çift yönlü önerme doğrulanmıştır. Bilindiği üzere, Prob-1 NP-tam bir problem olduğu için [31] Prob-2 de NP-tam bir problemdir ve polinom zamanda çözümü mümkün değildir. Prob-2, bir optimizasyon problemi olan $1|LT + F, s_j|C_{max}$ probleminin karar versiyonu olduğu için $1|LT + F, s_j|C_{max}$ bir NP-zor problemdir ve böylece ıspat tamamlanmıştır. ■

6.2.3. Tam Sayılı Doğrusal Olmayan Programlama Modeli

Bu bölümde, tanımlanan $1|LT+F, s_j|C_{max}$ problemi aşağıdaki değişkenler kullanılarak, bir TDOP olarak ifade edilmiştir.

Girdiler

s_j, p_j $(\forall j \in N)$, k_0 ve $p_{\max} = \max_{j\in N}\{p_j\}$.

Karar Değişkeni

$$x_j = \begin{cases} 1 & \text{eğer } j \text{ işi } \pi' \text{ alt çizelgesinde ise} \\ 0 & \text{diğer durumda} \end{cases} \qquad \forall j \in N$$

Yardımcı Değişkenler

$$V = \max_{j\in\pi'}\{p_j\}$$

$$w_j = \begin{cases} 1 & \text{eğer } p_j = \max_{i\in\pi'}\{p_i\} \text{ ise} \\ 0 & \text{diğer durumda} \end{cases} \qquad \forall j \in N$$

Model

$$\min \sum_{j=1}^{n}\left(s_j + x_j p_j\left(1 - f\left(\sum_{i=1}^{j-1} x_i p_i\right)\right) + p_j(1-x_j)\left(1 - h\left(\sum_{i=1}^{n} x_i p_i + \sum_{i=1}^{j-1} p_j(1-x_j), s_j\right)\right)\right)$$

Kısıtlar

$$\sum_{j=1}^{n} x_j p_j \geq k_0 + 1 \tag{6.70}$$

$$\sum_{j=1}^{n} x_j p_j \leq k_0 + V \tag{6.71}$$

$$V \geq x_j p_j \qquad \forall j \in N \tag{6.72}$$

$$V \leq w_j p_j + (1 - w_j) p_{\max} \qquad \forall j \in N \qquad (6.73)$$

$$x_j - w_j \geq 0 \qquad \forall j \in N \qquad (6.74)$$

$$\sum_{j=1}^{n} w_j = 1 \qquad (6.75)$$

$$x_j \in \{0,1\}, w_j \in \{0,1\}, V \in \mathbb{Z}^+ \qquad \forall j \in N \qquad (6.76)$$

Modelde görüldüğü üzere, denklem 6.70 ve 6.71' e göre işlerin π' alt çizelgesine atanması sağlanır. Denklem 6.72-6.75, π' alt çizelgesindeki son işi tespit etmek için gerekmektedir. Tüm bu doğrusal kısıtlar altında (denklem 6.70-6.76), Teorem 6.1' e göre tasarlanan ve doğrusal olmayan amaç fonksiyonuyla maksimum tamamlanma zamanı minimize edilir. Geliştirilen modelde, işlerin SPT kuralına göre etiketlenmesi gerekmektedir. Bu sıralamada işlem süreleri eşit olan işler arasında, ayar süresine göre azalan sıralama yapılmalıdır.

6.2.4. Dinamik Programlama Algoritması

Bu bölümde, Rudek' in [135] $1|LT|\sum w_j C_j$ problemi için önerdiği DP, $1\left|LT + F, s_j\right|C_{max}$ problemi için uyarlanmıştır. Olurlu bir çözüm, farklı özellikleri bulunan alt çizelgelerden oluştuğu için $1|LT|\sum w_j C_j$ ve $1\left|LT + F, s_j\right|C_{max}$ benzer parçalı yapıya sahiptir. Bu parçalı yapı, kesikli zaman aralıklarına yapılan atamaları kullanarak ilerleyen bir DP kullanmaya olanak sağlamaktadır. Bilindiği üzere, maksimum tamamlanma zamanı işlerin gerçekleşen işlem sürelerinin toplamına eşittir ve bir işin gerçekleşen işlem süresi de kendisinden önce çizelgelenen işlerin normal işlem süreleri toplamına bağlıdır. Bu nedenle, tamsayılarla ifade edilebilen evrelerden oluşan bir DP' nı, tanımlanan problemin kesin çözümü için tasarlamak mümkündür. Bölüm 5.2' deki tanıma göre, $\pi = \{\pi', \pi''\}$ denklem 5.8 ve 5.9' u sağlayan olurlu bir çizelgedir ve her iş π' veya π'' alt çizelgelerinden sadece birinde bulunabilir. Bu nedenle, bir işin gerçekleşen işlem süresini hesaplamak için π' ve π'' alt çizelgelerinin zaman sınırlarını bilmek gerekmektedir. Eğer olurlu bir $\pi =$

$\{\pi', \pi''\}$ çizelgesinin $\pi' = \{\hat{\pi}, j\}$ alt çizelgesindeki son iş j belirlenir ise, $\hat{P}$ değeri $[\max\{k_0 - p_j + 1, 0\}, \min\{k_0, P - p_j\}]$ aralığında olur. Buradaki $P = \sum_{k \in N} p_k$ tüm işlerin normal işlem sürelerinin toplamıdır, ayrıca $P'' = \sum_{k \in \pi''} p_k$ değeri de hesaplamalarda kullanılmak üzere, π'' alt çizelgesindeki işlerin normal işlem sürelerinin toplamıdır. Görüldüğü üzere, $P'' = P - P' = P - \hat{P} - p_j$ olduğundan dolayı, π' alt çizelgesindeki son iş (j) belirlendiğinde $\hat{P}$ değerinin bulunduğu aralık netleşirken, bu aralıktan seçilen $\hat{P}$ değerine göre de P'' belirlenmiş olur.

DP (π_{SPT}, k_0, P) → (π^*):

Adım 1: $l \leftarrow 1, \pi \leftarrow \pi_{SPT}, \pi^* \leftarrow \pi_{SPT}$ ve $T^* \leftarrow T^{\pi_{SPT}}$.

Adım 2: $j \leftarrow \pi(l)$ ve $m \leftarrow l$.

Adım 3: $\pi(m) \leftarrow \pi(m-1)$ ve $m \leftarrow m - 1$.

Adım 4: Eğer $m > 1$ ise Adım 3' e git. Aksi halde $\pi(1) \leftarrow j$.

Adım 5: $p \leftarrow \max\{k_0 - p_j + 1, 0\}$ ve $k \leftarrow 1$.

Adım 6: $\hat{P} \leftarrow p$ ve $P'' \leftarrow P - \hat{P} - p_j$.

Adım 7: $t' \leftarrow 0$.

Adım 8: $t'' \leftarrow 0$.

Adım 9: $(k, t', t'') \leftarrow F_j(k, t', t'')$ ve $t'' \leftarrow t'' + 1$.

Adım 10: Eğer $t'' \leq P''$ ise Adım 9' a git. Aksi halde $t' \leftarrow t' + 1$.

Adım 11: Eğer $t' \leq \hat{P}$ ise Adım 8' e git. Aksi halde $k \leftarrow k + 1$.

Adım 12: Eğer $k \leq n$ ise Adım 7' ye git.

Adım 13: Eğer $T^* < (n, \hat{P}, P'')$ ise $T^* \leftarrow (n, \hat{P}, P'')$ ve $\pi^* \leftarrow \pi^{(n, \hat{P}, P'')}$.

Adım 14: $p \leftarrow p + 1$.

Adım 15: Eğer $p \leq \min\{k_0, P - p_j\}$ ise Adım 6' ya git. Aksi halde $m \leftarrow 1$.

Adım 16: $\pi(m) \leftarrow \pi(m+1)$ ve $m \leftarrow m + 1$.

Adım 17: Eğer $m < l$ ise Adım 16' ya git. Aksi halde $\pi(l) \leftarrow j$.

Adım 18: $l \leftarrow l + 1$.

Adım 19: Eğer $l \leq n$ ise Adım 2' ye git.

Şekil 6.18. Dinamik programlama algoritması

Buradan hareketle, kalan diğer işlerin ($k \in N\backslash j$) atanabileceği $k \in \hat{\pi}$ ve $k \in \pi''$ durumlarını göz önünde bulundurarak, bu işler için DP' nın (Şekil 6.18) tüm evrelerindeki gerçekleşen işlem süreleri hesaplanabilir. Teorem 6.1' e göre, π' ve π'' alt çizelgelerinin her ikisi de SPT sıralamasına uygun olduğu için geliştirilen DP' nın (Şekil 6.18) evrelerinde de bu sıralama gözetilmiş ve dolayısıyla, ihlal eden evreler dikkate alınmamıştır.

İlk olarak, kalan işler ($N\backslash j$) SPT kuralına göre etiketlenir (Adım 1). Bu etiketleme esnasında, işlem süreleri eşit olan işler arasında ayar süresine göre azalan sıralama yapılır ve etiket 1 değeri j işini temsil eder (Adım 2-4). Tanımlanan $F_j(k, t', t'')$ fonksiyonu, bir $\pi = \{\hat{\pi}, j, \pi''\}$ çizelgesine atanan ilk k işe ait en küçük maksimum tamamlanma zamanını temsil etmektedir. Ayrıca, bu fonksiyon DP' nın (k, t', t'') evresine karşılık gelmektedir. Belirlenen bir j işi (Adım 2) ve bu işe göre belirlenen bir $\hat{P} \in \left[\max\{k_0 - p_j + 1, 0\}, \min\{k_0, P - p_j\}\right]$ değeri (Adım 5) için denklem 6.77' teki başlangıç değerleri dikkate alınarak,

$$F_j(1, t', t'') = \begin{cases} s_j + p_j\left(1 - f\left(\hat{P}\right)\right), & \text{eğer } t' = 0 \text{ ve } t'' = 0 \text{ ise} \\ +\infty, & \text{diğer durumda} \end{cases} \tag{6.77}$$

her $k = 2,3,\cdots,n$, her $t' = 0,1,\cdots,\hat{P}$ ver her $t'' = 0,1,\cdots,P''$ için denklem 6.78' deki yineleme ile $\pi = \{\hat{\pi}, j, \pi''\}$ çizelgesine ait en küçük C_{max}^{π} değeri hesaplanır (Adım 6-13).

$$F_j(k, t', t'') = \min\{F^*; F^{**}; F_j(k-1, t', t'')\} \tag{6.78}$$

Denklem 6.78' deki F^* ve F^{**} sırasıyla

$$\begin{cases} F_j(k-1, t'-p_k, t'') + s_k + p_k\left(1 - f(t'-p_k)\right), & \text{eğer } t'-p_k \geq 0 \text{ ise} \\ +\infty, & \text{diğer durumda} \end{cases}$$

ve

$$\begin{cases} F_j(k-1, t', t''-p_k) + s_k + p_k\big(1 - h(P' + t'' - p_k, s_k)\big), & \text{eğer } t' - p_k \geq 0 \text{ ise} \\ +\infty, & \text{diğer durumda} \end{cases}$$

ile gösterildiği gibi hesaplanmaktadır. Denklem 6.78' deki yineleme, belirlenmiş bir j işi (dolayısıyla ilgili çizelge $\pi = \{\hat{\pi}, j, \pi''\}$ yapıda olur) ve belirlenmiş bir $\hat{P}$ değeri için $\left(n, \hat{P}, P''\right)$ evresindeki en küçük C_{max}^{π} değerini tespit eder. Bundan dolayı, her $j \in N$ işi ve j işine göre belirlenen her $\hat{P} \in \left[\max\{k_0 - p_j + 1, 0\}, \min\{k_0, P - p_j\}\right]$ değeri için, denklem 6.77' deki ayarlarla başlayarak, denklem 6.78' deki yineleme $\left(n, \hat{P}, P''\right)$ evresi hesaplanana kadar tekrarlandığında, denklem 6.79 ile gösterilen optimum çözüm elde edilmiş olur (Adım 1-19).

$$C_{max}^{\pi} = \min_{j \in N} \left\{ \min_{\hat{P} \in [\max\{k_0 - p_j + 1, 0\},\, \min\{k_0, P - p_j\}]} \left\{ F_j\left(n, \hat{P}, P''\right) \right\} \right\} \tag{6.79}$$

Teorem 6.4. Tanımlanan $1\big|LT + F, s_j\big|C_{max}$ probleminin optimum çözümü için geliştirilen DP' nın koşum süresi en fazla $O(nP^3)$ olmaktadır.

Ispat. Belirlenmiş bir $j \in N$ işi ve bu işe göre belirlenmiş bir $\hat{P}$ değeri dikkate alındığında, $\left(n, \hat{P}, P''\right)$ evresindeki en küçük maksimum tamamlanma zamanını hesaplamak için denklem 6.78' deki yineleme $(n-1)\hat{P}P''$ kez tekrarlanır. Dolayısıyla, koşum süresi en fazla $O(nP^2)$ olur. $\hat{P}$ değerinin tanımlandığı aralığa göre, koşum süresi $O(nP^2)$ olan adımlar, her bir $j \in N$ işi için en fazla p_j kez tekrarlanır. Bunun neticesinde, algoritmanın koşum süresi en fazla $O\left(\left(\sum_{j \in N} p_j\right) nP^2\right) = O(nP^3)$ olur. ■

Doğal Sonuç 6.2. Tanımlanan $1\big|LT + F, s_j\big|C_{max}$ problemi normal şiddette NP-zor problemdir.

Ispat. Teorem 6.3 ve 6.4' e göre, tanımlanan problem normal şiddette NP-zor problemdir. ■

7. ÇÖZÜM YÖNTEMLERİNİN UYGULAMASI

Bu bölümde, geliştirilen çözüm yöntemlerinin performansını değerlendirmek için yapılan bir takım deneysel çalışmalar her yönüyle ele alınmıştır. Tanımlanan her iki problem için gerçekleştirilen uygulamalar ve bunların sonuçları farklı alt bölümlerde verilmiştir. Önerilen tüm çözüm yöntemleri (TDOP dışında), C programlama dilinde kodlanmış, Intel(R) Core(TM) i7-6700K CPU 4.00 GHz hızında ve 16.00 GB ön belleği olan kişisel bir bilgisayarda çalıştırılmıştır.

7.1. $1|LP, s_{ij}|\sum_j T_j$ Problemine Ait Yöntemlerin Uygulaması

Literatürde, $1|s_{ij}|\sum_j T_j$ problemi için bilinen bazı soru setleri bulunmaktadır [13, 221, 265]. Bunun gibi teslim tarihi içeren problem tiplerinde, soru örneğine ait işlem süreleri kullanılarak, polinom zamanda hesaplanabilen optimum C_{max} değeri veya tahmini bir C_{max} değeri, teslim tarihi türetme sürecinde bir parametre olarak kullanılmaktadır. Söz konusu $1|s_{ij}|\sum_j T_j$ problemi de, polinom zamanda ancak tahmini bir C_{max} değerinin tespit edilebildiği bir yapıdadır. Soru örneği türetme sürecinde kullanılan tahmin yöntemi, C_{max} için tam anlamıyla bir alt sınır [11] üretmese de, ortalama bir değer verebilmektedir. Bunun yanında, tanımlanan $1|LP, s_{ij}|\sum_j T_j$ problemi öğrenme etkisi içerdiğinden ve öğrenme etkisinin C_{max} değeri üzerinde azaltıcı bir etkisi olduğundan dolayı, literatürde bulunan soru setlerini olduğu gibi kullanmak, çözüm yöntemlerinin performansını değerlendirme konusunda yanlıtıcı olacaktır. Bu nedenle, Cicirello' nun [265] önerdiği soru türetme sistemi, tanımlanan $1|LP, s_{ij}|\sum_j T_j$ problemi için yeniden uyarlanmıştır.

Soru türetme sürecinde kullanılan parametreler, teslim tarihi sıkılık faktörü (τ), teslim tarihi aralık faktörü (R) ve ayar süresi şiddet faktörüdür (η). Türetilen soru örnekleri ayrıca, iş sayısı (n) ve öğrenme etkisi faktörüne (L) göre de biçimlendirilmiştir. Her parametreye ait seviyeler ve değerleri Çizelge 7.1' de verilmiştir. Her bir iş sayısına (n) göre, en az bir parametresi farklı olan 36 soru seti

türetilmiştir ve her bir soru seti de 20 soru örneği içermektedir. Böylece, toplam 7920 adet soru örneği türetilmiştir. Bu süreçte kullanılan değişkenler ve belirlenen değerler aşağıda verildiği gibidir.

p_j	normal işlem süreleri üniform dağılım [50, 150].
s_{ij}	ayar süreleri üniform dağılım [0, $2\bar{s}$].
d_j	teslim tarihleri τ olasılıkla üniform dağılım [$\bar{d}(1-R)$, $\bar{d}$] ve $1-\tau$ olasılıkla üniform dağılım [$\bar{d}$, $\bar{d}+\left(C_{max}-\bar{d}\right)R$].
$\bar{p}$	normal işlem sürelerinin ortalaması ($\bar{p}$ = 100).
$\bar{s}$	ayar sürelerinin ortalaması ($\bar{s}=\bar{p}\eta$).
$\bar{d}$	teslim tarihlerinin ortalaması ($\bar{d}=(1-\tau)C_{max}$).
C_{max}	maksimum tamamlanma zamanı ($C_{max}=n.\beta.\bar{s}+\hat{C}_{max}$), [265]' de $C_{max}=n(\bar{p}+\beta.\bar{s})$ kullanılmıştır.
β	C_{max} tahminleyicisi faktörü [265].
$\hat{C}_{max}$	ayar sürelerinin olmadığı ve öğrenme etkisi altındaki maksimum tamamlanma zamanı ($\hat{C}_{max}=\sum_{r=1}^{n} p^{A}_{\pi(r)}$).
$p^{A}_{\pi(r)}$	gerçekleşen işlem zamanı ($p^{A}_{\pi(r)}=p_{\pi(r)}r^{a}$).
a	öğrenme etkisi çarpanı $a=\log_2 L$.
r	π çizelgesindeki pozisyonlar.
π	normal işlem sürelerine p_j göre SPT sıralaması.

Çizelge 7.1. Kullanılan parametre değerleri

Parametreler	Seviye	Değerler
n	11	15, 20, 25, 30, 35, 40, 60, 80, 100, 200, 500
τ	3	0.3, 0.6, 0.9
R	2	0.25, 0.75
η	2	0.25, 0.75
L	3	0.7, 0.8, 0.9

7.1.1. Sezgisel Yöntemlerin Parametre Ayarları

Çözüm yöntemlerinin etkili sonuçlar elde etmesi için yapısal özellikleri kadar, kullanılan bazı parametre değerlerinin uyumu da önemlidir. Bu bölümde, geliştirilen GA ve DKA' larının performansını artıracak parametre değerlerinin tespiti için bir takım deneyler yapılmıştır.

Yapılan deneylerin problem boyutundan etkilenmemesi için, iş sayısı $n = 500$ ve 200 dışında diğer tüm soru setlerinin son 5 problemi kullanılmıştır. Böylece, sadece parametre ayarlarında kullanılmak üzere $s(PS) = 1620$ problemden oluşan bir soru seti PS oluşturulmuştur. Tüm algoritmalar her bir problem için 5 kez çalıştırılmıştır ve her bir koşum $x = (1,2,\cdots,5)$ ile ifade edilmiştir. Herhangi bir probleme ait tüm koşumlar ATCS ile elde edilen sonuca göre değerlendirilmiştir. Böylece, i. problemin $(i \in PS)$ x. koşumuna ait sonucu $T^{x(i)}$ olarak ifade edilirken, ATCS ile elde edilmiş sonucu $T^{ATCS(i)}$ ile ifade edilmiştir. Daha sonra, iyileşme yüzdesi $V^{x(i)} = \left(T^{x(i)}/T^{ATCS(i)}\right) \times 100$ ile hesaplanmıştır. Yine aynı problemin koşum ortalaması $V^i = \sum_{x=1}^{5} V^{x(i)}/5$ ile hesaplanırken, deney ortalaması $\bar{V}_{Exp} = \sum_{i \in PS} V^i/s(PS)$ ile hesaplanmıştır. Burada görülen Exp ifadesi yapılan deneyin adını ifade etmektedir. Benzer şekilde, yapılan deneylerin ortalama koşum zamanları da saniye birimiyle $\bar{t}_{Exp}$ olarak ifade edilmiştir ve aynı yöntemle hesaplanmıştır.

7.1.1.1. GA' lar İçin Parametre Ayarları

İlk olarak minimum ve maksimum sıcaklık değerleri belirlenmiştir, çünkü sıcaklık parametresi uygunluk değerini etkileyen önemli bir değişkendir. Bölüm 6.1.2.3' de belirtilen uygunluk değeri (0,1] aralığında bir değer almaktadır. Böylece, denklem 6.49' de belirtilen sapma oranı $\Delta T = 1000$ olan kalitesiz bir çözümün, denklem 6.50' de belirtilen uygunluk değerinin 0.36 gibi anlamlı bir değer alması için, sıcaklık değerinin 1000 olması gerekmektedir. Benzer şekilde, $\Delta T = 100$ olan bir çözümün aynı uygunluk değerini alması için, sıcaklık değerinin 100 olması gerekmektedir. Buradan yola çıkarak, sıcaklık maksimum ve minimum değerleri

sırasıyla, 1000 ve 100 olarak belirlenmiştir. Çünkü $\Delta T = 1000$ olan bir çözüm, popülasyonda çeşitliliğin artırılması için yerel optimumdan yeterince uzak bir çözümdür ve $\Delta T < 100$ olan bir çözüm de, yerel optimuma ulaşmak için yeterince iyi bir çözümdür. Sıcaklık sınırları belirlendikten sonraki ilk deneyler, başlangıç sıcaklığı ve sıcaklık değişim oranı için yapılmıştır. Çizelge 7.2' de, yapılan deneylerin sonuçları ile birlikte başlangıç sıcaklığı için test edilen değerler, sıcaklık artış ve azalış oranları için test edilen değerler birlikte verilmiştir. Değişkenlerin tüm kombinasyonlarını içerecek şekilde toplam 12 deney yapılmıştır. Deney ITR10, diğer tüm deneylerden daha iyi sonuç verdiğinden dolayı, başlangıç sıcaklığı, sıcaklık artış ve azalış oranları için ITR10 deneyine ait değerler kabul edilmiştir.

Çizelge 7.2. Başlangıç sıcaklığı, sıcaklık artış ve azalış oranları için yapılan deney sonuçları

Exp	Başlangıç Sıcaklığı	Sıcaklık Azalış Oranı	Sıcaklık Artış Oranı	$\bar{V}_{Exp}$	$\bar{t}_{Exp}$
ITR1	100	0.95	1.05	74.4	1.66
ITR2	100	0.95	1.10	74.5	1.67
ITR3	100	0.90	1.05	74.5	1.63
ITR4	100	0.90	1.10	74.4	1.62
ITR5	500	0.95	1.05	73.0	2.35
ITR6	500	0.95	1.10	73.0	2.33
ITR7	500	0.90	1.05	73.1	2.13
ITR8	500	0.90	1.10	73.2	2.20
ITR9	1000	0.95	1.05	72.6	2.53
ITR10	**1000**	**0.95**	**1.10**	**72.3**	**2.47**
ITR11	1000	0.90	1.05	72.5	2.38
ITR12	1000	0.90	1.10	72.4	2.39

İkinci olarak, popülasyon miktarı, sonlandırma sayacı ve sıcaklık sayacı için deneyler yapılmıştır. İlgili parametreler için test edilen değerlere ait tüm

kombinasyonlar denenmiştir. Çizelge 7.3' de bu değerler ve deney sonuçları birlikte verilmiştir. Popülasyon miktarının 100 olduğu deneyler, 200 olduğu deneylere göre daha etkili sonuçlar vermiştir. PTT17 deneyi $\bar{V}_{Exp}$ değerine göre en iyi sonuca sahip olsa da, $\bar{t}_{Exp}$ değerindeki dikkate değer farktan dolayı PTT15 deneyindeki parametre değerleri tercih edilmiştir. Bunun yanında, çaprazlama ve mutasyon oranları sırasıyla 0.95 ve 0.5 olarak belirlenmiştir.

Çizelge 7.3. Popülasyon miktarı, sonlandırma ve sıcaklık sayaçları için yapılan deney sonuçları

Exp	Popülasyon Miktarı	Sonlandırma Sayacı	Sıcaklık Sayacı	$\bar{V}_{Exp}$	$\bar{t}_{Exp}$
PTT1	100	10	2	73.3	1.31
PTT2	200	10	2	73.6	1.34
PTT3	100	20	2	73.0	2.14
PTT4	200	20	2	73.3	2.10
PTT5	100	30	2	72.4	2.68
PTT6	200	30	2	73.0	2.68
PTT7	100	10	5	72.4	2.86
PTT8	200	10	5	72.9	2.82
PTT9	100	20	5	72.2	4.17
PTT10	200	20	5	72.5	4.04
PTT11	100	30	5	72.0	5.50
PTT12	200	30	5	72.3	5.17
PTT13	100	10	10	72.0	4.65
PTT14	200	10	10	72.7	4.37
PTT15	100	20	10	71.6	7.26
PTT16	200	20	10	72.2	6.89
PTT17	**100**	**30**	**10**	**71.4**	**10.03**
PTT18	200	30	10	72.1	9.22

Çizelge 7.4. Komşuluk yapıları ve sıralaması için yapılan deney sonuçları

Exp	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\bar{V}_{Exp}$	$\bar{t}_{Exp}$	Exp	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\bar{V}_{Exp}$	$\bar{t}_{Exp}$
NS1	1	2	3	72.19	0.47	NS31	1	4	5	75.54	0.37
NS2	1	3	2	71.82	0.46	NS32	1	5	4	75.85	0.35
NS3	2	1	3	72.93	0.49	NS33	4	1	5	75.57	0.38
NS4	2	3	1	73.19	0.43	NS34	4	5	1	76.16	0.38
NS5	3	1	2	72.50	0.48	NS35	5	1	4	78.11	0.36
NS6	3	2	1	72.64	0.41	NS36	5	4	1	77.77	0.38
NS7	1	2	4	75.48	0.35	NS37	2	3	4	73.30	0.42
NS8	1	4	2	74.65	0.39	NS38	2	4	3	72.80	0.47
NS9	2	1	4	78.51	0.37	NS39	3	2	4	72.73	0.42
NS10	2	4	1	78.02	0.37	NS40	3	4	2	72.30	0.47
NS11	4	1	2	74.60	0.41	NS41	4	2	3	72.64	0.46
NS12	4	2	1	75.77	0.36	NS42	4	3	2	72.29	0.45
NS13	1	2	5	74.04	0.40	NS43	2	3	5	72.12	0.47
NS14	1	5	2	73.69	0.39	NS44	2	5	3	72.08	0.52
NS15	2	1	5	76.16	0.41	NS45	3	2	5	71.15	0.44
NS16	2	5	1	76.68	0.40	**NS46**	**3**	**5**	**2**	**70.96**	**0.46**
NS17	5	1	2	75.41	0.41	NS47	5	2	3	72.71	0.48
NS18	5	2	1	76.33	0.39	NS48	5	3	2	71.91	0.46
NS19	1	3	4	74.52	0.39	NS49	2	4	5	76.78	0.39
NS20	1	4	3	73.76	0.42	NS50	2	5	4	77.56	0.37
NS21	3	1	4	78.23	0.33	NS51	4	2	5	75.03	0.37
NS22	3	4	1	77.89	0.33	NS52	4	5	2	74.91	0.39
NS23	4	1	3	73.82	0.45	NS53	5	2	4	77.33	0.36
NS24	4	3	1	74.86	0.39	NS54	5	4	2	76.34	0.39
NS25	1	3	5	72.11	0.46	NS55	3	4	5	73.76	0.39
NS26	1	5	3	71.71	0.48	NS56	3	5	4	74.04	0.35
NS27	3	1	5	73.39	0.41	NS57	4	3	5	72.89	0.43
NS28	3	5	1	73.40	0.37	NS58	4	5	3	72.64	0.47
NS29	5	1	3	72.81	0.50	NS59	5	3	4	73.48	0.41
NS30	5	3	1	73.07	0.43	NS60	5	4	3	73.19	0.49

7.1.1.2. DKA' lar İçin Parametre Ayarları

İlk olarak kullanılacak komşuluk yapılarını belirleyen deneyler yapılmıştır. Bölüm 6.1.4.3' de bahsedilen 5 farklı komşuluk yapısından en iyi sonuç veren 3' ünü ve bunların sırasını belirlemek için 60 adet deney yapılmıştır. Bu deneyler esnasında, karıştırma operatörü olarak deneyde kullanılan komşuluk yapıları kullanılmış ve karıştırma miktarı olarak $n/3$ değeri belirlenmiştir.

Çizelge 7.4' de, deneylerin sonuçlarıyla birlikte, içerdiği komşuluk yapıları ve sıralamaları verilmiştir. Çizelge 7.4' de görülen 1,2,3,4 ve 5 sayıları sırasıyla, iki işin yer değişimi, geriye kaydırma, ileriye kaydırma, ikili yer değişimi ve ikili kaydırma yöntemlerini ifade etmektedir. Sonuç ortalamaları $\bar{V}_{Exp}$ bakımından, deney NS46 diğer deneylere göre daha başarılı olmuştur ve $\bar{t}_{Exp}$ değerleri açısından NS46 ile diğer deneyler arasında anlamlı bir fark görülmemektedir. Ayrıca, $n = 15$ dışındaki diğer tüm problem boyutlarında, NS46 diğer deneylere göre daha iyi sonuçlar vermiştir. Bu deneyler sonucunda DKA' larda, ileriye kaydırma, ikili kaydırma ve geriye kaydırma komşuluk yapıları bu sıra ile tercih edilmiştir.

Çizelge 7.5. Karıştırma miktarı için yapılan deney sonuçları

Exp	t	$\bar{V}_{Exp}$	$\bar{t}_{Exp}$	Exp	t	$\bar{V}_{Exp}$	$\bar{t}_{Exp}$
SH1	0	80.60	0.07	SH4	10	71.03	1.96
SH2	**2**	**70.22**	**0.75**	SH5	20	72.25	2.59
SH3	6	70.70	1.35	SH6	30	73.53	2.55

İkinci olarak, karıştırma miktarı (t) için deneyler yapılmıştır. Bu deneylerde 0, 2, 6, 10, 20 ve 30 değerleri kullanılmıştır. Çizelge 7.5' de görüldüğü üzere, SH2 deneyi $\bar{V}_{Exp}$ değerine göre daha başarılı olmuştur. Ancak, deney sonuçları iş boyutuna göre ayrıca incelendiğinde, SH3 deneyinin $n \leq 40$ boyutlu problemlerde daha başarılı olduğu görülmüştür. Bu nedenle, karıştırma miktarı $n \leq 40$ için 6, aksi halde 2

olarak belirlenmiştir. Son olarak, algoritmanın sonlandırma sayacı için, gelişme olmayan ardışık 10 iterasyon belirlenmiştir.

7.1.2. Çözüm Yöntemlerinin Performans Değerlendirmesi

Bu bölümde, geliştirilen yöntemlerle elde edilen problem çözümleri ve bu çözümler üzerindeki değerlendirmeler anlatılmıştır. DS için, ortalama ve maksimum düğüm sayıları, ortalama ve maksimum koşum süresi (CPU - sn.) ve son olarak, belirlenen sürede optimum çözümüne ulaşılabilen problem sayısı (OS) kaydedilmiştir. DS' nın maksimum koşum süresi, her bir problem için 3600 sn. olarak belirlenmiştir. DS' nın performansıyla ilgili olarak, Çizelge 7.6 ve 7.7' de görüldüğü üzere, iş sayısı n büyüdükçe düğüm sayısı ve koşum süresi artmaktadır. Bu durum öğrenme etkisi için de benzerdir. Öğrenme etkisi zayıfladığında (en zayıf öğrenme etkisi 0.9, daha sonra 0.8), DS optimum çözüme ulaşmak için daha fazla düğüm ve zaman kullanmıştır.

Daha önce belirtildiği üzere, τ ve R sırasıyla geciken işlerin beklenen oranını ve $[d_{min}, d_{max}]$ aralığını belirleyen parametrelerdir. Her ikisi de soru türetme sürecinde, teslim tarihlerinin belirlenmesinde kullanılan parametreler olmasına rağmen, τ sonuçlar üzerinde daha etkili olmuştur. Teslim tarihi sıkılık faktöründeki (τ) değişimler dikkate alındığında, oluşturulan düğüm sayısındaki değişim nispeten daha fazla olmuştur.

Öğrenme etkisi değeri ve diğer problem parametreleri arasındaki ilişki Şekil 7.1' de görülmektedir ve 'average node' olarak belirtilen ortalama düğüm sayısı, DS ile çözülen tüm problem boyutlarının ortalamasıdır. Teslim tarihi sıkılık faktörü τ, teslim tarihi aralık faktörü R ve ayar süresi şiddet faktörü η arttıkça, ortalama düğüm miktarı ve ortalama koşum süresi azalmaktadır. Ancak parametre değişimlerinin etkisi eşit değildir. Teslim tarihi aralık faktörünü R dikkate aldığımızda, özellikle $L = 0.7$ ve 0.8 için, değişimin etkisi diğer parametrelere oranla biraz daha zayıftır. Ayar süresi şiddet faktörünü η dikkate aldığımızda ise, tüm öğrenme etkisi değerleri için, değişimin etkisi büyük oranda benzer olmuştur. Son olarak, DS' nın teslim tarihi sıkılık faktörüne τ daha duyarlı olduğunu söyleyebiliriz.

Önerilen sezgisel algoritmalar için ortalama hata oranları kaydedilmiştir. Her bir probleme ait hata oranı $V^{x(i)}$ hesaplanmıştır. Buradaki, $x = 1,2,\cdots,36$ değeri herhangi bir n değerine ait problem setini, $i = 1,2,\cdots,20$ değeri ise, bu problem setlerinden herhangi birine ait olan bir problemi ifade etmektedir. Böylece, herhangi bir n değerine ait problem setindeki i. problemin yaklaşık çözümü $T^{x(i)}$ olarak ifade edilirken, kesin çözüm veya sezgiseller arasındaki en iyi yaklaşık çözüm de $T_{min}^{x(i)}$ ile ifade edilmiştir. Böylece, $V^{x(i)} = \left[(T^{x(i)} - T_{min}^{x(i)})/T_{min}^{x(i)}\right] \times 100$ ile hesaplanmıştır. Buradan yola çıkarak, herhangi bir n değerine ait her bir problem setinin ortalama hata oranı $\bar{V}_x = \sum_{i \in x} V^{x(i)}/20$ formülüyle ve herhangi bir sezgisele ait ortalama hata oranı da $\bar{V} = \sum_{x=1}^{36} \bar{V}_x/36$ formülüyle hesaplanmıştır. Aynı yöntemle, herhangi bir sezgisele ait ortalama koşum süresi de $\bar{t}$ (CPU – sn.) hesaplanmıştır. Aşağıda verilen tüm tablolarda, sezgisel yöntem adının altındaki hücrelerde $\bar{V}_x$ değerleri, son satırda ise, yine aynı sezgisele ait $\bar{V}$ ve $\bar{t}$ değerleri verilmiştir.

Çizelge 7.8-7.11' de, sırasıyla $n = 15$, $n = 20$, $n = 25$ ve $n = 30$ boyutlu problemlere ait sezgisel çözümlerin DS' dan elde edilen optimum sonuçlara göre hata oranlarının ortalaması verilmiştir. Çizelgelerin her bir satırıyla ifade edilen soru setlerindeki en başarılı sezgisel yöntemin ortalama hata oranı ($\bar{V}_x$), her bir çizelgedeki en başarılı sezgisel yöntemin ortalama hata oranı ($\bar{V}$) ve ortalama koşum süresi ($\bar{t}$) koyu olarak vurgulanmıştır. En büyük $\bar{V}_x$ değerleri $\tau = 0.3$ iken gerçekleşmiştir ve tüm sezgiseller için $\bar{t}$ değeri 0.2 sn. altında kalmıştır. DKA1 ve DKA2, $\bar{V}$ değerlerine göre GA1 ve GA2' den daha başarılı olmuştur. Problem boyutu $n = 15$ iken, sezgisel yöntemler DKA1, DKA2, GA1 ve GA2' nin 720 problemden sırasıyla 546, 513, 478 ve 290 adet problemin optimum sonucuna ulaşabilmişlerdir. Problem boyutu $n = 20$ iken, aynı sırayla 298, 258, 229 ve 111 problemin optimum sonucuna, problem boyutu $n = 25$ iken 183, 169, 73 ve 61 problemin optimum sonucuna ve son olarak problem boyutu $n = 30$ iken 182, 175, 82 ve 61 problemin optimum sonucuna ulaşabilmişlerdir. Bu durum DKA1' in optimum sonuca ulaşmada daha başarılı olduğunu göstermektedir. GA1' in uygunluk fonksiyonunda sıcaklık parametresi kullanımının olumlu etkisi sonuçlara yansımıştır. Bunun yanında, $\tau = 0.3$ iken, GA2' nin GA1' e göre bazen daha iyi sonuçlar verdiği görülmektedir.

Çizelge 7.6. Problem boyutu $n = 15$ ve $n = 20$ için dal-sınır algoritmasının performansı

				$n = 15$					$n = 20$				
				Düğüm		CPU		OS	Düğüm		CPU		OS
L	τ	R	η	Max.	Ort.	Max.	Ort.		Max.	Ort.	Max.	Ort.	
0.7	0.3	0.25	0.25	73668	22504	0.64	0.22	20	3619816	1031866	60.41	18.17	20
			0.75	29572	7413	0.28	0.07	20	8483744	811090	135.76	12.93	20
		0.75	0.25	72061	17081	0.56	0.17	20	3268248	609988	50.38	11.25	20
			0.75	14858	3495	0.17	0.04	20	164947	24445	1.81	0.45	20
	0.6	0.25	0.25	3970	1533	0.06	0.03	20	66257	19078	1.57	0.53	20
			0.75	7021	2780	0.11	0.05	20	123396	43834	2.54	1.09	20
		0.75	0.25	5969	2385	0.09	0.04	20	150156	34760	4.02	0.85	20
			0.75	4991	2427	0.08	0.04	20	167263	50347	2.97	1.14	20
	0.9	0.25	0.25	2212	957	0.03	0.02	20	26943	7804	0.54	0.22	20
			0.75	2769	919	0.05	0.02	20	20621	7884	0.63	0.25	20
		0.75	0.25	2194	826	0.03	0.02	20	21245	9431	0.55	0.26	20
			0.75	2613	893	0.05	0.02	20	41444	13077	0.92	0.34	20

Çizelge 7.6. Devam

				$n = 15$					$n = 20$				
				Düğüm		CPU		OS	Düğüm		CPU		OS
L	τ	R	η	Max.	Ort.	Max.	Ort.		Max.	Ort.	Max.	Ort.	
0.8	0.3	0.25	0.25	133317	50212	1.02	0.45	20	9518171	2819967	161.52	45.35	20
			0.75	67781	26598	0.48	0.23	20	7712735	1537282	115.92	22.46	20
		0.75	0.25	126342	29307	1.13	0.27	20	18830791	3362969	231.24	44.66	20
			0.75	68663	10718	0.48	0.09	20	7598817	579053	95.96	7.48	20
	0.6	0.25	0.25	9890	3949	0.13	0.06	20	130934	54423	3.08	1.41	20
			0.75	7177	3315	0.09	0.05	20	237356	68305	6.03	1.74	20
		0.75	0.25	10368	3655	0.13	0.06	20	1158842	205771	21.92	4.32	20
			0.75	8636	3134	0.10	0.05	20	331934	111547	7.21	2.50	20
	0.9	0.25	0.25	4617	1370	0.06	0.03	20	54338	15753	0.95	0.36	20
			0.75	2699	1041	0.06	0.02	20	26660	12274	0.66	0.33	20
		0.75	0.25	7594	1646	0.07	0.03	20	37843	14785	0.70	0.34	20
			0.75	2625	1394	0.05	0.03	20	143864	25753	2.77	0.59	20

Çizelge 7.6. Devam

				$n = 15$					$n = 20$				
				Düğüm		CPU		OS	Düğüm		CPU		OS
L	τ	R	η	Max.	Ort.	Max.	Ort.		Max.	Ort.	Max.	Ort.	
0.9	0.3	0.25	0.25	305989	80945	2.50	0.69	20	319615284	33738982	3600.02	426.86	19
			0.75	263041	36869	1.86	0.31	20	10149915	2523604	136.39	36.02	20
		0.75	0.25	1461723	108558	7.98	0.72	20	82913184	8001895	827.43	99.80	20
			0.75	94761	20208	0.72	0.17	20	7343905	681498	99.90	10.74	20
	0.6	0.25	0.25	16814	6488	0.20	0.09	20	543625	221410	16.88	6.37	20
			0.75	11299	5466	0.16	0.07	20	1232421	177867	26.14	4.33	20
		0.75	0.25	81091	10104	0.78	0.11	20	1916269	522593	47.76	11.74	20
			0.75	20920	6593	0.23	0.09	20	986389	215184	17.82	4.56	20
	0.9	0.25	0.25	5920	1815	0.08	0.03	20	39880	12974	0.92	0.33	20
			0.75	3750	1844	0.07	0.03	20	66466	19843	1.28	0.45	20
		0.75	0.25	12340	2600	0.14	0.04	20	230893	32345	4.59	0.72	20
			0.75	11657	1974	0.10	0.03	20	76978	18607	1.52	0.44	20

Çizelge 7.7. Problem boyutu $n = 25$ ve $n = 30$ için dal-sınır algoritmasının performansı

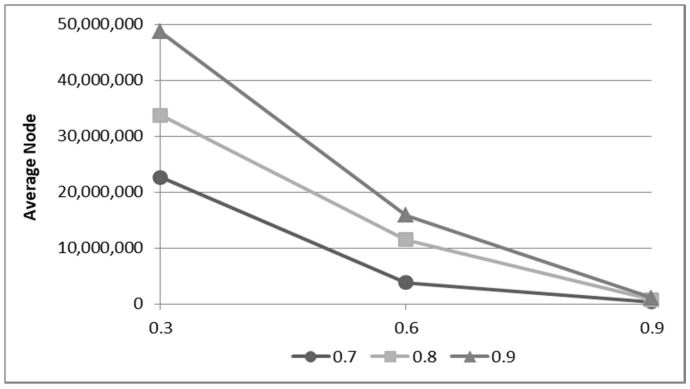
				$n = 25$					$n = 30$				
				Düğüm		CPU		OS	Düğüm		CPU		OS
L	τ	R	η	Max.	Ort.	Max.	Ort.		Max.	Ort.	Max.	Ort.	
0.7	0.3	0.25	0.25	141173492	85446261	3600.01	2445.65	11	130002736	104689576	3600.02	3600.01	0
			0.75	133687514	26467763	3600.01	667.72	18	175407259	16556876	3600.01	541.96	17
		0.75	0.25	196106327	46762024	3600.01	1099.57	16	183892087	78749797	3600.02	2782.35	7
			0.75	2188964	119589	39.36	2.35	20	8410671	663503	297.02	27.32	20
	0.6	0.25	0.25	1198882	314257	43.49	12.67	20	14684706	3919893	903.87	260.27	20
			0.75	2973552	970936	138.61	42.47	20	33899752	11052858	2125.44	743.96	20
		0.75	0.25	7498135	1583452	252.78	57.45	20	73889377	24289450	3600.02	1321.04	16
			0.75	2951984	777993	107.39	30.17	20	64154042	17879874	3600.01	1058.94	19
	0.9	0.25	0.25	332285	81526	10.99	3.08	20	4046352	838663	208.29	47.09	20
			0.75	373554	97477	13.85	3.58	20	6939022	1659832	381.87	89.98	20
		0.75	0.25	721672	168695	23.74	6.01	20	11399125	1533428	598.77	79.53	20
			0.75	562822	155876	17.46	5.48	20	5829314	1344799	330.69	76.90	20

Çizelge 7.7. Devam

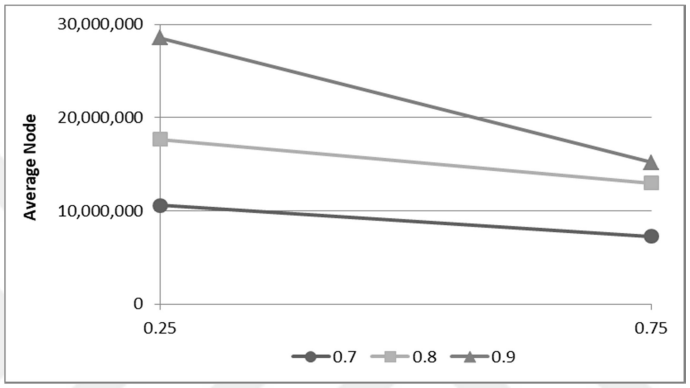
				$n = 25$					$n = 30$				
				Düğüm		CPU		OS	Düğüm		CPU		OS
L	τ	R	η	Max.	Ort.	Max.	Ort.		Max.	Ort.	Max.	Ort.	
0.8	0.3	0.25	0.25	173238502	127665258	3600.02	2946.81	7	146034301	109718091	3600.02	3600.01	0
			0.75	158917684	29443737	3600.00	681.99	19	106028197	85051509	3600.02	3245.56	4
		0.75	0.25	204613060	68986311	3600.01	1658.70	14	161623227	89383418	3600.02	3421.59	2
			0.75	2264099	315195	64.76	7.87	20	249461645	20427642	3600.00	303.13	19
	0.6	0.25	0.25	5025322	1327499	206.00	60.47	20	58363523	20521132	3600.01	1485.98	18
			0.75	4992531	1878103	194.12	78.95	20	71012866	37886788	3600.01	2519.99	15
		0.75	0.25	112452896	10778306	3600.01	370.82	19	93363403	55588896	3600.02	2769.97	9
			0.75	46547671	8442691	1617.02	293.85	20	82638980	47423445	3600.01	2398.59	12
	0.9	0.25	0.25	1007950	140767	35.07	5.20	20	17769970	2309233	939.68	125.40	20
			0.75	484219	142869	16.16	5.57	20	26271988	3418576	1364.25	185.67	20
		0.75	0.25	1349737	242796	44.77	8.73	20	11886677	2516232	630.25	140.55	20
			0.75	1178135	383989	39.80	13.51	20	12613186	2945524	674.92	165.71	20

Çizelge 7.7. Devam

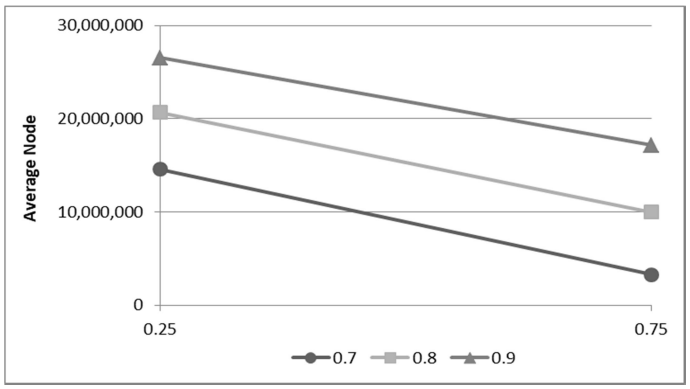
				$n = 25$					$n = 30$				
				Düğüm		CPU		OS	Düğüm		CPU		OS
L	τ	R	η	Max.	Ort.	Max.	Ort.		Max.	Ort.	Max.	Ort.	
0.9	0.3	0.25	0.25	225690118	184983749	3600.02	3588.96	1	147975331	101859506	3600.02	3600.01	0
			0.75	215039288	104368730	3600.01	2214.82	14	178047315	131941917	3600.02	3479.14	1
		0.75	0.25	274746927	81546360	3600.01	1580.14	15	173778162	82308439	3600.02	3600.01	0
			0.75	139361768	9304441	3568.49	226.01	20	149044874	36853606	3600.02	1054.29	16
	0.6	0.25	0.25	39198681	9169031	1706.74	434.80	20	62219904	45806594	3600.01	3497.61	2
			0.75	20297225	6409134	687.34	265.56	20	84240972	57429332	3600.01	3489.03	2
		0.75	0.25	144682772	13665218	3600.01	410.46	19	92825073	65474570	3600.02	3462.50	1
			0.75	16592683	5147968	656.51	198.62	20	89548001	49576141	3600.02	2705.55	8
	0.9	0.25	0.25	1372464	328595	50.12	12.41	20	8135749	2375847	394.75	139.87	20
			0.75	1991089	463758	67.12	16.23	20	10262649	3165726	545.25	185.26	20
		0.75	0.25	6749954	794769	240.46	27.85	20	23677776	6118751	1268.54	356.57	20
			0.75	1622621	458371	56.90	16.84	20	9358571	3348294	629.23	213.62	20



(a)



(b)



(c)

Şekil 7.1. Soru parametrelerinin, öğrenme şiddetine göre ortalama düğüm sayısı üzerindeki etkileri: (a) teslim tarihi sıkılık faktörü τ, (b) teslim tarihi aralık faktörü R, (c) ayar süresi şiddet faktörü η

Çizelge 7.8. Sezgisel algoritmaların $n = 15$ için ortalama hata oranları (%)

			$L = 0.7$				$L = 0.8$				$L = 0.9$			
τ	R	η	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2
0.3	0.25	0.25	2	**1**	2	7	**1**	**1**	**1**	11	**1**	2	**1**	5
		0.75	**14**	15	16	30	**6**	9	16	10	10	**8**	20	14
	0.75	0.25	**1**	2	4	18	5	**3**	6	12	**0**	2	6	12
		0.75	**1**	14	39	14	**24**	26	115	122	24	**11**	124	279
0.6	0.25	0.25	**0**	1	1	3	**0**	**0**	**0**	2	**0**	**0**	**0**	1
		0.75	**1**	2	4	7	**1**	2	2	6	**0**	1	1	4
	0.75	0.25	**0**	1	**0**	2	**0**	**0**	**0**	2	**0**	**0**	**0**	1
		0.75	1	**0**	2	8	**2**	**2**	**2**	6	1	1	**0**	4
0.9	0.25	0.25	**0**	**0**	**0**	1	**0**	**0**	**0**	1	**0**	**0**	**0**	1
		0.75	**0**	**0**	1	3	**0**	1	**0**	2	**0**	**0**	**0**	2
	0.75	0.25	**0**	**0**	**0**	1	**0**	**0**	**0**	1	**0**	**0**	**0**	**0**
		0.75	**0**	**0**	1	3	**0**	**0**	1	4	**0**	1	1	2
$\bar{V}$ ($\bar{t}$)			**DKA1 → 3 (0.01)**,				DKA2 → 3 (0.01),				GA1 → 10 (0.02),			
													GA2 → 17 (0.02)	

Çizelge 7.9. Sezgisel algoritmaların $n = 20$ için ortalama hata oranları (%)

			$L = 0.7$				$L = 0.8$				$L = 0.9$			
τ	R	η	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2
0.3	0.25	0.25	**5**	9	12	25	**3**	5	6	11	**4**	**4**	**4**	12
		0.75	**118**	140	192	281	**85**	130	146	121	28	**20**	37	35
	0.75	0.25	**5**	7	13	19	**3**	6	18	20	**4**	7	23	17
		0.75	10	**8**	27	42	45	**12**	59	17	25	**3**	115	29
0.6	0.25	0.25	2	**1**	**1**	4	**0**	1	1	4	**0**	1	1	2
		0.75	5	**4**	5	8	**3**	4	4	10	**3**	4	5	9
	0.75	0.25	**1**	**1**	2	3	**1**	**1**	**1**	3	**0**	**0**	2	3
		0.75	**5**	**5**	7	16	**3**	5	5	9	**3**	4	4	10
0.9	0.25	0.25	**0**	**0**	1	1	**0**	**0**	**0**	2	**0**	**0**	**0**	1
		0.75	**2**	**2**	**2**	5	**1**	**1**	**1**	3	**1**	**1**	**1**	3
	0.75	0.25	**0**	1	1	2	**0**	**0**	**0**	1	**0**	**0**	**0**	1
		0.75	2	**1**	2	5	**1**	**1**	**1**	4	**1**	**1**	**1**	2
$\bar{V}$ ($\bar{t}$)			**DKA1 → 10 (0.02)**,				DKA2 → 11 (0.02),				GA1 → 19 (0.05),			GA2 → 21 (0.04)

Çizelge 7.10. Sezgisel algoritmaların $n = 25$ için ortalama hata oranları (%)

			$L = 0.7$				$L = 0.8$				$L = 0.9$			
τ	R	η	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2
0.3	0.25	0.25	**9**	10	31	36	**3**	7	15	18	**2**	3	8	10
		0.75	**11**	24	21	35	592	**276**	662	717	42	**39**	64	56
	0.75	0.25	**11**	12	64	48	22	**11**	67	62	**8**	11	30	26
		0.75	**13**	**13**	84	40	29	**11**	41	30	**33**	94	87	82
0.6	0.25	0.25	**2**	4	6	8	**2**	**2**	4	5	**1**	**1**	3	4
		0.75	**6**	9	20	22	**7**	8	11	13	**3**	4	8	9
	0.75	0.25	**2**	4	8	10	2	**1**	4	6	**1**	**1**	3	4
		0.75	**6**	10	18	19	**8**	9	19	23	5	**4**	10	14
0.9	0.25	0.25	1	**0**	1	3	**1**	**1**	**1**	3	**0**	1	1	2
		0.75	**3**	4	5	7	**2**	**2**	4	7	2	**1**	3	3
	0.75	0.25	**1**	**1**	2	3	**1**	**1**	**1**	2	**0**	**0**	1	2
		0.75	**2**	3	5	6	**2**	**2**	4	5	2	**1**	3	4
$\bar{V}$ ($\bar{t}$)			DKA1 → 23 (0.06),				**DKA2 → 16 (0.05)**,				GA1 → 37 (0.07),			GA2 → 37 (0.06)

Çizelge 7.11. Sezgisel algoritmaların $n = 30$ için ortalama hata oranları (%)

			$L = 0.7$				$L = 0.8$				$L = 0.9$			
τ	R	η	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2
0.3	0.25	0.25	**7**	8	34	38	**2**	4	11	16	**3**	**3**	5	11
		0.75	**0**	2	2	1	97	**18**	160	181	**17**	**17**	43	51
	0.75	0.25	**29**	56	139	131	**14**	17	76	47	**7**	12	52	54
		0.75	**46**	72	185	143	14	**4**	22	39	9	**3**	62	10
0.6	0.25	0.25	**3**	5	6	8	**2**	3	3	7	**1**	**1**	**1**	3
		0.75	**15**	17	22	28	9	**8**	10	15	**2**	3	3	8
	0.75	0.25	4	**3**	5	9	**2**	**2**	3	9	**1**	**1**	**1**	4
		0.75	**9**	12	18	20	7	**6**	12	16	**3**	6	6	11
0.9	0.25	0.25	**1**	**1**	3	4	**1**	**1**	**1**	2	**1**	**1**	**1**	2
		0.75	**4**	**4**	5	8	**3**	4	5	5	**2**	**2**	3	5
	0.75	0.25	**2**	**2**	**2**	4	**1**	**1**	**1**	2	**1**	**1**	**1**	2
		0.75	**4**	**4**	5	8	**3**	**3**	4	6	**2**	3	3	5
$\bar{V}$ ($\bar{t}$)			DKA1 → 9 (0.11),				**DKA2 → 9 (0.11)**,				GA1 → 25 (0.14),			GA2 → 25 (0.09)

Çizelge 7.12. Sezgisel algoritmaların $n = 35$ için ortalama hata oranları (%)

			$L = 0.7$				$L = 0.8$				$L = 0.9$			
τ	R	η	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2
0.3	0.25	0.25	**10**	**10**	24	35	**3**	**3**	9	16	**2**	3	4	12
		0.75	257	**1**	388	635	59	**36**	256	171	**11**	15	28	37
	0.75	0.25	29	**23**	98	60	**15**	20	57	74	**5**	11	37	49
		0.75	**0**	**0**	**0**	**0**	**11**	21	183	92	9	**0**	27	23
0.6	0.25	0.25	**2**	**2**	3	8	2	**1**	3	5	1	**0**	2	4
		0.75	4	**3**	9	14	**2**	4	6	10	2	**1**	2	6
	0.75	0.25	2	**1**	2	7	**1**	2	3	6	**1**	**1**	2	3
		0.75	**6**	7	9	18	**3**	4	8	13	5	**2**	6	7
0.9	0.25	0.25	**0**	1	1	3	**1**	**1**	**1**	2	**0**	**0**	1	2
		0.75	**1**	**1**	3	5	**1**	**1**	2	4	**1**	**1**	**1**	3
	0.75	0.25	**1**	**1**	**1**	2	1	**0**	1	2	**0**	**0**	1	2
		0.75	**1**	**1**	**1**	3	**1**	**1**	**1**	3	**1**	**1**	**1**	3
$\bar{V}$ ($\bar{t}$)			DKA1 → 12 (0.19),				**DKA2 → 5 (0.20)**,				GA1 → 33 (0.26),			GA2 → 37 (0.15)

Çizelge 7.13. Sezgisel algoritmaların $n = 40$ için ortalama hata oranları (%)

			$L = 0.7$				$L = 0.8$				$L = 0.9$			
τ	R	η	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2
0.3	0.25	0.25	**5**	10	19	26	**5**	**5**	10	21	3	**2**	3	9
		0.75	**0**	14	24	1	44	**43**	113	68	**7**	13	27	28
	0.75	0.25	27	**14**	134	116	**13**	**13**	60	55	**4**	15	39	47
		0.75	**10**	142	141	34	**0**	1	170	253	96	**2**	163	119
0.6	0.25	0.25	2	**1**	2	4	**1**	2	2	5	**1**	**1**	2	4
		0.75	6	**4**	7	12	**2**	**2**	3	7	2	**1**	2	6
	0.75	0.25	**1**	3	3	8	**1**	**1**	3	6	**1**	**1**	2	5
		0.75	**4**	5	8	14	8	**4**	11	15	**3**	4	5	9
0.9	0.25	0.25	**1**	**1**	**1**	2	**0**	1	1	2	**0**	**0**	1	1
		0.75	2	**1**	**1**	4	**1**	**1**	**1**	2	**1**	**1**	**1**	2
	0.75	0.25	**0**	1	1	2	**0**	1	1	2	**0**	**0**	1	1
		0.75	**1**	2	2	4	2	**1**	2	3	**1**	**1**	**1**	3
$\bar{V}$ ($\bar{t}$)			**DKA1 → 7 (0.32)**,			DKA2 → 9 (0.35),			GA1 → 27 (0.45),			GA2 → 25 (0.26)		

Çizelge 7.14. Sezgisel algoritmaların $n = 60$ için ortalama hata oranları (%)

			$L = 0.7$				$L = 0.8$				$L = 0.9$			
τ	R	η	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2
0.3	0.25	0.25	19	**9**	37	37	6	**2**	13	19	3	**2**	5	12
		0.75	**0**	**0**	**0**	**0**	392	318	**314**	347	19	**4**	34	40
	0.75	0.25	41	**26**	286	341	61	**30**	235	300	20	**6**	73	56
		0.75	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
0.6	0.25	0.25	2	**1**	3	5	2	**1**	3	5	**1**	**1**	2	4
		0.75	5	**3**	11	13	3	**2**	6	8	2	**1**	6	8
	0.75	0.25	2	**1**	3	6	2	**1**	3	6	**1**	**1**	2	5
		0.75	7	**3**	16	18	6	**1**	12	12	**3**	**3**	4	8
0.9	0.25	0.25	**1**	**1**	**1**	2	1	**0**	1	2	**0**	**0**	1	1
		0.75	**1**	**1**	2	4	1	**0**	1	3	1	**0**	1	1
	0.75	0.25	1	**0**	1	2	1	**0**	1	2	1	**0**	**0**	1
		0.75	**1**	**1**	2	3	1	**0**	1	2	**1**	**1**	**1**	2
$\bar{V}$ ($\bar{t}$)			DKA1 → 17 (0.66),				**DKA2 → 12 (1.03)**,				GA1 → 30 (1.73),			GA2 → 35 (0.97)

Çizelge 7.15. Sezgisel algoritmaların $n = 80$ için ortalama hata oranları (%)

			$L = 0.7$				$L = 0.8$				$L = 0.9$			
τ	R	η	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2
0.3	0.25	0.25	13	**9**	40	38	3	**2**	11	18	5	**2**	4	12
		0.75	**0**	**0**	**0**	**0**	**94**	499	407	386	**12**	**12**	43	40
	0.75	0.25	**96**	110	471	554	146	**38**	395	236	**7**	12	63	63
		0.75	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
0.6	0.25	0.25	2	**1**	2	4	**1**	**1**	2	5	**1**	**1**	**1**	4
		0.75	9	**2**	10	10	**3**	**3**	**3**	7	**2**	**2**	3	7
	0.75	0.25	2	**1**	3	7	3	**0**	3	6	2	**1**	2	5
		0.75	5	**3**	12	14	**4**	5	8	10	4	**2**	5	9
0.9	0.25	0.25	**1**	**1**	**1**	2	**0**	**0**	1	1	**0**	**0**	**0**	1
		0.75	**1**	**1**	2	3	**1**	**1**	**1**	**1**	1	**0**	1	1
	0.75	0.25	**0**	**0**	1	2	**0**	**0**	**0**	1	**0**	**0**	**0**	1
		0.75	2	**1**	2	4	1	**0**	1	2	1	**0**	1	1
$\bar{V}$ ($\bar{t}$)			**DKA1 → 12 (1.81)**,				DKA2 → 20 (3.07),				GA1 → 42 (5.99),			
													GA2 → 40 (3.07)	

Çizelge 7.16. Sezgisel algoritmaların $n = 100$ için ortalama hata oranları (%)

			$L = 0.7$				$L = 0.8$				$L = 0.9$			
τ	R	η	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2
0.3	0.25	0.25	25	**4**	63	62	**3**	**3**	21	23	3	**1**	6	14
		0.75	**0**	**0**	**0**	**0**	64	**0**	48	11	13	**11**	45	37
	0.75	0.25	**26**	34	102	77	75	**36**	124	131	**3**	15	48	64
		0.75	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
0.6	0.25	0.25	2	**1**	4	4	2	**1**	3	5	2	**1**	2	4
		0.75	5	**2**	13	13	4	**1**	6	7	2	**1**	4	6
	0.75	0.25	**2**	**2**	3	7	**1**	**1**	3	5	**1**	**1**	2	4
		0.75	5	**3**	16	18	4	**2**	9	9	4	**1**	7	9
0.9	0.25	0.25	**1**	**1**	**1**	**1**	1	**0**	**0**	1	**0**	**0**	**0**	1
		0.75	**1**	**1**	2	3	1	**0**	1	1	**0**	**0**	**0**	**0**
	0.75	0.25	1	**0**	1	1	**0**	**0**	**0**	1	**0**	**0**	**0**	1
		0.75	1	**0**	2	2	1	**0**	1	1	**0**	**0**	1	1
$\bar{V}$ ($\bar{t}$)			DKA1 → 7 (3.67),				**DKA2 → 3 (6.57)**,				GA1 → 15 (10.11),			GA2 → 15 (5.78)

Çizelge 7.17. Sezgisel algoritmaların $n = 200$ için ortalama hata oranları (%)

			$L = 0.7$				$L = 0.8$				$L = 0.9$			
τ	R	η	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2
0.3	0.25	0.25	**0**	**0**	**0**	**0**	8	**6**	25	28	3	**1**	5	11
		0.75	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	18	**13**	88	62
	0.75	0.25	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	38	**14**	46	109
		0.75	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
0.6	0.25	0.25	3	**1**	6	7	**1**	**1**	2	5	1	**0**	1	4
		0.75	22	**7**	43	34	4	**2**	7	8	3	**1**	4	7
	0.75	0.25	4	**2**	5	9	2	**1**	3	7	1	**0**	1	4
		0.75	7	**4**	21	21	6	**2**	9	12	3	**1**	5	10
0.9	0.25	0.25	1	**0**	1	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
		0.75	1	**0**	2	2	1	**0**	1	1	**0**	**0**	**0**	**0**
	0.75	0.25	1	**0**	1	1	**0**	**0**	**0**	1	**0**	**0**	**0**	**0**
		0.75	2	**1**	2	2	1	**0**	1	1	**0**	**0**	**0**	**0**
$\bar{V}$ ($\bar{t}$)			DKA1 → 4 (37),				**DKA2 → 2 (80)**,				GA1 → 8 (153),			GA2 → 10 (76)

Çizelge 7.18. Sezgisel algoritmaların $n = 500$ için ortalama hata oranları (%)

			$L = 0.7$				$L = 0.8$				$L = 0.9$			
τ	R	η	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2	DKA1	DKA2	GA1	GA2
0.3	0.25	0.25	**0**	**0**	**0**	**0**	36	**5**	136	138	3	**1**	5	21
		0.75	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0.75	0.25	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
		0.75	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
0.6	0.25	0.25	6	**2**	12	19	2	**0**	3	8	1	**0**	2	6
		0.75	**0**	**0**	**0**	**0**	9	**7**	26	34	2	**0**	4	9
	0.75	0.25	3	**2**	11	14	3	**0**	5	9	1	**0**	3	6
		0.75	**63**	121	573	553	6	**3**	16	23	3	**2**	10	15
0.9	0.25	0.25	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
		0.75	1	**0**	1	2	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0.75	0.25	1	**0**	1	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
		0.75	**1**	**1**	2	2	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
$\bar{V}$ ($\bar{t}$)			**DKA1 → 4 (959)**,				DKA2 → 4 (2032),				GA1 → 23 (3999),			
													GA2 → 24 (1464)	

Çizelge 7.12-7.18' de $n > 30$ için verilen ortalama hata oranları hesaplanırken, $T_{min}^{x(i)}$ değeri için sezgisel yöntemler arasındaki en iyi sonuç kullanılmıştır. Çizelgelerin her bir satırıyla ifade edilen soru setlerindeki en başarılı sezgisel yöntemin ortalama hata oranı ($\bar{V}_x$), her bir çizelgedeki en başarılı sezgisel yöntemin ortalama hata oranı ($\bar{V}$) ve ortalama koşum süresi ($\bar{t}$) koyu olarak vurgulanmıştır. Bu boyuttaki problemler için elde edilen bulgular da, optimum sonuca göre elde edilen bulgulara benzerdir. Ancak, $\bar{V}$ değerlerine göre DKA2 daha başarılı olmuştur. Bu durum, karıştırma işlemindeki farklılıkla ilgidir. Bölüm 6.1.5' de anlatıldığı üzere DKA2' de, karıştırma işlemi yapmadan önce mevcut çözüm üzerinde baskınlık kuralları test edilmiştir. Bu testin özellikle büyük boyutlu problemlerde etkili olduğu söylenebilir. Tüm sezgiseller ve tüm L değerleri için en büyük $\bar{V}_x$ değeri, $\tau = 0.3$ iken meydana gelmiştir. GA1, $\bar{V}$ değerini dikkate aldığımızda GA2' den daha başarılı olsa da, özellikle $\tau = 0.3$ iken, GA2 bazen GA1' e göre daha başarılı olmuştur. Problem boyutu $n = 200$ için $\tau = 0.3$ ve $L < 0.9$ iken, toplam gecikme genellikle 0 olmuştur. Benzer şekilde, problem boyutu $n = 500$ için $\tau = 0.3$ iken, toplam gecikme genellikle 0 olmuştur. Ortalama koşum süresi $\bar{t}$' yi dikkate aldığımızda, DKA1 daha az koşum süresine sahipken, DKA2 ve GA1 daha büyük koşum süreleriyle çalışmıştır. Bu durum DKA2 ve GA1' in daha iyi sonuçlara ulaşabildiğinin bir başka göstergesidir, çünkü tüm sezgisellerin genel olarak tüm operatörleri ve sonlandırma koşulları aynıdır. Sonuç olarak, özellikle problem boyutu büyükken DKA2 diğer tüm yöntemlere nazaran daha başarılı ve GA1 genellikle GA2' den daha başarılı olmuştur.

7.2. $1|LT + F, s_j|C_{max}$ Problemine Ait Yöntemlerin Uygulaması

Tanımlanan $1|LT + F, s_j|C_{max}$ problemi, içerdiği öğrenme ve unutma modeli açısından yeni bir problem olduğundan ve ilk olarak bu tezde ele alındığından dolayı, literatürde bu yapı için uygun bir soru seti bulunmamaktadır. Bu nedenle, önerilen çözüm yöntemlerinin performansını değerlendirmek için yeni soru setleri oluşturulmuştur. Soru türetme sürecinde kullanılan parametreler, iş sayısı (n), olası en büyük işlem süresi ($\tilde{p}_{\max}$) ve eşik değeri oranıdır (σ). Her parametreye ait

seviyeler ve değerleri Çizelge 7.19' da verilmiştir. Parametrelere ait tüm kombinasyonlara göre, her birinde 30 soru örneği bulunan 36 soru seti oluşturulmuştur. Böylece, toplam 1080 adet soru örneği türetilmiştir. Bu süreçte kullanılan değişkenler ve belirlenen değerler aşağıda verildiği gibidir.

p_j	normal işlem süreleri üniform dağılım [1, $\tilde{p}_{\max}$].
s_j	ayar süreleri üniform dağılım [0, $\tilde{p}_{\max}$].
k_0	eşik değeri $\lfloor P\sigma \rfloor$, $P = \sum_{j=1}^{n} p_j$.
$1-(1+x)^a$	$f(x)$ öğrenme fonksiyonu $a = -0.152$.
$1-(1+y)^a$	g(y) unutma fonksiyonu.
$x_0 = \min_{j \in N}\{p_j\}$	normalizasyon değeri.

Çizelge 7.19. Kullanılan parametre değerleri

Parametreler	Seviye	Değerler
n	4	25, 50, 100, 200
$\tilde{p}_{\max}$	3	10, 50, 100
σ	3	0.25, 0.50, 0.75

7.2.1. Çözüm Yöntemlerinin Performans Değerlendirmesi

Bu bölümde, $1|LT+F, s_j|C_{max}$ problemine ait optimum çözümler için önerilen TDOP ve DP' nın çözüm süreleri değerlendirilmiştir. TDOP için uygun çözücü içeren LINGO 9.0 programı kullanılmıştır ve 'Nonlinear Solver' varsayılan ayarlarla çalıştırılmıştır. Her soru setindeki 30 problem örneğinin ardışık çözümünü pratikleştirmek için MS Excel 2010 ara yüzü kullanılmıştır. TDOP ve DP' nın koşum süreleri (CPU - sn.) sırasıyla Çizelge 7.20 ve 7.21' de verilmiştir. İlgili çizelgelerde, n, $\tilde{p}_{\max}$ ve σ değerlerine göre türetilmiş olan bir soru setinin çözümüne ait koşum süresi, $\tilde{p}_{\max}$ - σ satırı ve n sütunun kesiştiği hücrelerde sırasıyla en küçük (Min), ortalama (Ort) ve en büyük (Max) değerleriyle raporlanmıştır.

Çizelge 7.20. Tam sayılı doğrusal olmayan programlama modelinin çözüm süreleri (sn.)

		$n = 25$			$n = 50$			$n = 100$			$n = 200$		
$\tilde{p}_{\max}$	σ	Min	Ort	Max	Min	Ort	Max	Min	Ort	Max	Min	Ort	Max
10	0.25	41	315	1541	--	--	--	--	--	--	--	--	--
	0.50	4	68	723	474	5768	$>10^4$	--	--	--	--	--	--
	0.75	3	5	14	4	33	129	255	4749	$>10^4$	--	--	--
50	0.25	30	287	1685	--	--	--	--	--	--	--	--	--
	0.50	5	75	634	271	4254	$>10^4$	--	--	--	--	--	--
	0.75	1	5	20	3	53	264	305	3917	$>10^4$	--	--	--
100	0.25	23	308	1398	--	--	--	--	--	--	--	--	--
	0.50	2	99	782	32	4860	$>10^4$	--	--	--	--	--	--
	0.75	2	6	20	4	40	146	297	6041	$>10^4$	--	--	--

Koşum sürelerindeki artışa bağlı olarak TDOP ile en büyük $n = 100$ boyutlu problem örnekleri $\sigma = 0.75$ için çözülmüştür. Çizelge 7.20' de görüldüğü üzere, problem boyutundaki (n) artışa bağlı olarak TDOP' nın koşum süresinde daha şiddetli bir artış gerçekleşmiştir. Eldeki veriler ışığında, olası en büyük işlem süresinin ($\tilde{p}_{\max}$), modelin koşum süresine anlamlı bir etkisi olmadığı rahatlıkla söylenebilir. Bunun yanında, eşik değeri oranının (σ) artması koşum süresinin azalmasına yol açmıştır. Bu eğilimden yola çıkarak, $\sigma < 0.25$ durumunda gerçekleşecek olan koşum süresinin raporlanan değerlerden daha fazla olması beklenebilir. Ancak, eşik değeri oranının [0,1] aralığında olduğu dikkate alınırsa, eşik değeri oranıyla koşum süresi arasındaki ters orantılı ilişkinin bu aralığın bütünü için geçerli olduğu söylenemez. Örneğin, $k_0 = \lfloor P\sigma \rfloor < \min_{j \in N}\{p_j\}$ ifadesini sağlayan uygun bir σ değeri için π' alt çizelgesine atanabilecek iş sayısı 1 olur. Böylece, olurlu çözüm uzayının boyutu n' ye indirgenebilir ve TDOP' nın bu çözüm uzayındaki koşum süresi de en fazla $O(n)$ olur.

Görüldüğü üzere, eşik değeri π' alt çizelgesine atanabilecek iş sayısını ve dolayısıyla olurlu çözüm uzayını sınırlayan bir unsurdur. Örneğin, π' alt çizelgesine atanabilecek iş sayısı kesinlikle $|N'| = n'$ ise, olurlu çözüm uzayının boyutu $|D| \leq \binom{n}{n'}$ olur. Buradaki $\binom{n}{n'}$ ifadesi, n elemanlı bir kümenin n' elemanlı alt küme sayısıdır (kombinasyon). Böylece, π' alt çizelgesine atanabilecek iş sayısının alt ve üst sınırlarının sırasıyla n'_l ve n'_u olduğu düşünülürse, olurlu çözüm uzayının boyutu $|D| \leq \sum_{i=n'_l}^{n'_u} \binom{n}{i}$ olur. Bunun yanında, $\binom{n}{i}$ değeri $0 \leq i \leq \lfloor n/2 \rfloor$ aralığında artarken, $\lfloor n/2 \rfloor \leq i \leq n$ aralığında azalır. Daha açık bir ifadeyle, N' deki elemanlarla oluşturulacak alt kümelerin eleman sayısı $\lfloor N/2 \rfloor$' ye yakınlaştıkça, alt küme sayısı artmaktadır. Modelin koşum süresiyle olurlu çözüm uzayının büyüklüğü arasındaki ilişki bu açıdan ifade edildiğinde, $|D|_{\sigma=0.25} \geq |D|_{\sigma=0.5} \geq |D|_{\sigma=0.75}$ olur. Bu durumda, olurlu çözüm uzayının çoğunlukla π' alt çizelgesindeki iş sayısı n'_u değerine yakın olan çözümlerden meydana geldiği söylenebilir. Örneğin, $n = 25$ ve $\tilde{p}_{\max} = 10$ olan soru setindeki problem örneklerinde, $\sigma = 0.25$ için $13 \geq n'_u \geq 11$, $\sigma = 0.5$ için $17 \geq n'_u \geq 19$ ve $\sigma = 0.75$ için $23 \geq n'_u \geq 21$ olduğu gözlemlenmiştir.

Çizelge 7.21. Dinamik programlama algoritmasının çözüm süreleri (sn.)

		$n = 25$			$n = 50$			$n = 100$			$n = 200$		
$\tilde{p}_{\max}$	σ	Min	Ort	Max	Min	Ort	Max	Min	Ort	Max	Min	Ort	Max
10	0.25	<1	<1	<1	3	5	7	56	76	106	914	1339	1579
	0.50	<1	<1	<1	2	4	8	50	73	100	849	1258	1566
	0.75	<1	<1	<1	1	2	3	21	33	51	429	583	845
50	0.25	12	27	53	229	434	679	4472	7143	$>10^4$	--	--	--
	0.50	12	26	55	207	397	646	3977	6403	9692	--	--	--
	0.75	4	9	20	72	147	245	1543	2529	3977	--	--	--
100	0.25	67	181	307	1222	2954	5007	--	--	--	--	--	--
	0.50	60	172	324	1093	2637	4352	--	--	--	--	--	--
	0.75	26	59	96	416	981	1599	--	--	--	--	--	--

DP' nın koşum süresinin üst sınırı $O(nP^3)$ olarak belirlenmiştir (Teorem 6.4). Bu ifadeye göre DP, hem problem boyutundan hem de problem örneğindeki işlerin normal işlem sürelerinin toplamından etkilenmektedir ve Çizelge 7.21' de görülen koşum süreleri de bu durumu doğrulamıştır. Problem boyutundaki (n) artış beraberinde toplam işlem süresini de (P) artırdığı için koşum süresinde daha şiddetli bir artışa yol açmıştır. Benzer şekilde, olası en büyük işlem süresindeki ($\tilde{p}_{\max}$) artış da koşum süresinde artışa neden olmuştur. Ancak, buna bağlı artışın şiddeti problem boyutuna bağlı artışın şiddetinden daha az olmuştur. Örneğin, $\tilde{p}_{\max}$ değeri 50' den 100' e iki kat artarken DP' nın ortalama koşum süresi yaklaşık 7 kat artış göstermiş, n değeri ise 25' ten 50' ye iki kat artarken ortalama koşum süresi yaklaşık 16.5 kat artış göstermiştir.

Farklı σ değerleri için bu artış oranlarında anlamlı bir değişim olmamıştır. Örneğin, $n = 25$ boyutlu problem örneklerinde $\tilde{p}_{\max}$ değeri 50' den 100' e iki kat artarken, ortalama koşum süresindeki artış $\sigma = 0{,}25$ için yaklaşık 6.7 kat, $\sigma = 0{,}5$ için yaklaşık 6.6 kat ve $\sigma = 0{,}75$ için yaklaşık 6.6 kat olmuştur. Problem boyutu $n = 50$ olan örneklerde $\tilde{p}_{\max}$ değeri 50' den 100' e iki kat artarken, ortalama koşum süresindeki artış $\sigma = 0{,}25$ için yaklaşık 6.8 kat, $\sigma = 0{,}5$ için yaklaşık 6.7 kat ve $\sigma = 0{,}75$ için yaklaşık 6.7 kat olmuştur. Benzer şekilde, $\tilde{p}_{\max}$ değeri 10' dan 50' ye beş kat artarken, ortalama koşum süresindeki artış yaklaşık 100 kat olarak gerçekleşmiştir ve bu artış oranı farklı σ değerlerinden anlamlı bir şekilde etkilenmemiştir. Farklı σ değerleri açısından kendi içinde anlamlı bir değişimden bahsedilebilir. Örneğin, $\sigma = 0.75$ değerindeki koşum süresi, $\sigma = 0.25$ ve $\sigma = 0.5$ değerlerindeki koşum sürelerine nispeten daha az olmuştur.

Her iki çözüm yöntemi TDOP ve DP bir arada değerlendirildiğinde, $\tilde{p}_{\max} \geq 50$ ve $\sigma \geq 0.75$ olduğu durumda DOPT' nın DP' na göre daha başarılı olduğu söylenebilir. Bunun dışında diğer durumlarda ve özellikle $\tilde{p}_{\max} = 10$ olduğu durumda DP daha başarılı olmuştur.

8. SONUÇ VE ÖNERİLER

Bu tezde, değişken işlem sürelerinin ve beraberinde ayar sürelerinin olduğu tek makine çizelgeleme problemi ele alınmıştır. Değişken işlem süreleri çerçevesinde; zamana veya pozisyona bağlı öğrenme etkisi ve ayar süresine bağlı unutma etkisi dikkate alınmıştır. Öğrenme etkisi işlem sürelerini azaltan, unutma etkisi ise sadece öğrenmenin şiddetini azaltan unsurlar olarak kullanılmıştır. Bu kapsamda iki farklı çizelgeleme problemi tanımlanmıştır.

İlk problem, her işten önce BLAS olan ve işlem süreleri üzerinde pozisyona bağlı öğrenme etkisi olan tek makinede toplam gecikmenin minimizasyonu olarak tanımlanmıştır. Böylece, tanımlanan çizelgeleme problemi [29]' de belirtilen üçlü gösterime göre $1|LP, s_{ij}|\sum_j T_j$ ile ifade edilmiştir. NP-zor olan bu problemin küçük boyutlu örneklerinin kesin çözümü için bir alt sınır ve iki baskınlık kuralı içeren etkin bir DS geliştirilmiştir. Yapılan uygulama deneyleri, DS ile 30 veya daha az sayıda iş içeren problemlerin büyük kısmına ait optimum çözümlerin makul sürelerde bulunabildiğini göstermiştir. DS' na adapte edilen YA ile mevcut üst sınır değeri daha hızlı iyileştirilerek, muhtemel zaman kayıplarının önüne geçilmiştir. Ayrıca, dallanma aşamasında yatay ve dikey arama metotlarının sistematik kullanımı da koşum süresini oldukça azaltmıştır. Bu arama metotlarının verimli bir şekilde uygulanmasında, aktif düğüm adreslerinin saklandığı çok seviyeli zincir yapısı da etkili olmuştur.

Tanımlanan $1|LP, s_{ij}|\sum_j T_j$ probleminin büyük boyutlu örneklerine ait yaklaşık çözümleri elde etmek için popülasyon tabanlı sezgisel yöntem olan GA ve tek çözüm tabanlı sezgisel yöntem olan DKA geliştirilmiştir. Standart yapıdaki bir GA yanında, TB' deki sıcaklık parametresinin uygunluk fonksiyonuna adapte edilmesiyle farklılaştırılmış etkili bir GA daha geliştirilmiştir. Benzer şekilde, standart bir DKA yanında, baskınlık kuralları eklenerek zenginleştirilmiş karıştırma operatörü içeren bir başka DKA daha geliştirilmiştir. Yapılan uygulama deneyleri, tek çözüm tabanlı sezgisel yöntemlerin (DKA1 ve DKA2) daha başarılı olduğunu göstermiştir. Bunun temel nedeni, büyük ölçüde çözüm yöntemlerindeki yapısal farklılıklardır.

Tanımlanan problem BLAS ve öğrenme etkisi içerdiğinden dolayı, iki iş arasındaki basit bir yer değişimi bile toplam gecikmenin oldukça keskin bir şekilde değişimine yol açabilmektedir. Elbette bu durum, DKA' dan daha çok GA' yı olumsuz etkilemektedir. Her iki sezgisel yöntem de rassal işlemler içeren operatörlerden oluşsa da, DKA' daki karıştırma ve DKİ operatörlerini kontrol etmek ve yönlendirmek daha kolay olmaktadır. Böylece, lokal çözümlerden kurtulmayı sağlayan çeşitlendirme ve muhtemel iyileşmelere fırsat veren yoğunlaşma, DKA' da daha tutarlı bir şekilde gerçekleşmektedir. Bu nedenle, baskınlık kurallarını içeren DKA2 diğer sezgisel yöntemlerden daha başarılı olmuştur.

Uygulama deneyleri sonucunda GA' lar göz önüne alındığında, GA1 genellikle GA2' den daha başarılı olmuştur. Uygunluk fonksiyonuna adapte edilen sıcaklık parametresi, daha kötü amaç fonksiyonu değerine sahip bireylerin de genetik operatörler için seçilme şansını artırmıştır. Bu özellik GA1' na çeşitlendirme bakımından işlevsellik kazandırmıştır. Bu işlevsellik sonucunda, sıcaklık artışı olurlu çözüm uzayının farklı alanlarına geçişi sağlarken, sıcaklığın azalması lokal bölgelerdeki muhtemel iyileşmeleri yakalama fırsatı vermiştir.

İkinci problem, her işten önce BZAS olan ve işlem süreleri üzerinde hem zamana bağlı öğrenme etkisi hem de ayar sürelerine bağlı unutma etkisi olan tek makinede maksimum tamamlanma zamanının minimizasyonu olarak tanımlanmıştır. Böylece, tanımlanan çizelgeleme problemi [29]' de belirtilen üçlü gösterime göre $1|LT + F, s_j|C_{max}$ ile ifade edilmiştir. Çoğunlukla kesikli üretim süreçlerinde meydana gelen unutma etkisi, iş üzerinde geçen süre boyunca kazanılan tecrübenin bir miktar kaybına yol açmaktadır. Üretim sürecindeki kesinti ve duruşlar iş temposunun duruş öncesi seviyenin altına düşmesine ve dolayısıyla, öğrenme ile kazanılan verimliliğin bir miktar azalmasına sebep olmaktadır.

Bu çerçevede, her işlemden önce üretimin durmasına yol açan ayar süreleri, unutma etkisinin temel kaynağı olarak ele alınmıştır. İşlem süreleri öğrenme etkisinin, ayar süreleri de unutma etkisinin parametresi olduğundan, iki değişkenli yeni bir öğrenme-unutma modeli tanımlanmıştır. Global optimuma ait iki temel özellik ortaya konmuş ve buradan hareketle, tanımlanan problemin NP-zor olduğu ıspatlanmıştır.

Bu nedenle, optimum çözüm yöntemi olarak TDOP ve DP geliştirilmiştir. DP' nın koşum süresinin en fazla $O(nP^3)$ olduğu ve dolayısıyla sözde-polinom zamanlı bir çözüm yöntemi olduğu gösterilmiştir. Bu bulgu $1|LT+F,s_j|C_{max}$ probleminin normal şiddette NP-zor olduğunu desteklemiştir. Uygulama deneyleri, işlem süreleri arasındaki farkın büyük olduğu problem örneklerinde ve özellikle eşik değerinin büyük olduğu durumlarda, koşum süresi açısından TDOP' nın DP' na nispeten daha başarılı olduğunu göstermiştir. İşlem süreleri arasındaki farkın az olduğu problem örneklerinde ise, DP optimum sonuca daha hızlı ulaşmıştır.

Sonuçların verdiği motivasyonla, makine çizelgeleme alanındaki araştırmacıların son zamanlarda yoğun olarak üstüne düştüğü öğrenme ve unutma etkilerinin, günümüz endüstrisindeki planlamacıların ve karar vericilerin göz ardı etmemesi gereken konulardan olduğu rahatlıkla söylenebilir. Bunun yanında, ayar sürelerinin problem yapısında başlıca bir parametre olarak ifade edilmesi, çoğu zaman ihmal edilen veya işlem sürelerine dahil edilen bu olgunun sonuç üzerinde anlamlı etkilerinin olduğunu göstermiştir. Böylece, endüstriyel uygulamaların yürütülmesinde gösterilen hassasiyet, bu olguların dikkate alınmasıyla birlikte çizelgeleme alanında da önemli bulguların ortaya konmasına yardımcı olmuştur.

Gelecekteki araştırmalar kapsamında, DS' nın çeşitli baskınlık kuralları ve daha etkili bir alt sınırla geliştirilmesi mümkün olabilir. Tanımlanan problemlerin doğrusal olmayan yapıları yumuşatılarak doğrusal modeller türetilebilir ve bu modellerin daha kısa sürelerde çözülmesiyle sıkı alt sınırlar elde edilebilir. Kısıtların veya amaç fonksiyonunun modifiye edilmesiyle oluşturulan modellerin türettiği çözüm yapıları daha etkili, tutarlı ve hızlı sezgisel yöntemlerin geliştirilmesine de olanak sağlayabilir. Özellikle geliştirilen DP' ndan yola çıkarak, $1|LT+F,s_j|C_{max}$ problemi için (tamamen) polinom zamanlı yaklaşım şeması geliştirilebilir. Tanımlanan problemlerin farklı amaç fonksiyonları, farklı atölye ortamları ve farklı kısıtlarla genişletilmesi de ilerde yapılabilecek çalışmalar arasındadır. Ayrıca, çeşitli değişkenleri içeren farklı öğrenme ve unutma fonksiyonlarının tanımlanması da, bu tezde ele alınan çalışmanın etkilerinin geniş alana yayılmasını sağlayacaktır.

KAYNAKLAR

[1] Johnson, S.M., Optimal two- and three-stage production schedules with setup times included. Nav. Res. Logist. Q. 1 (1): 61–68, 1954.

[2] Smith, W.E., Various optimizers for single-stage production. Nav. Res. Logist. Q. 3 (1-2): 59–66, 1956.

[3] Muth, J.F., Thompson, G.L., Winters, P.R., Industrial scheduling. Prentice-Hall, New Jersey, 1963.

[4] Leung, J.Y.T., Handbook of scheduling: Algorithms, models, and performance analysis. CRC Press, Boca Raton, Florida, 2004.

[5] Pinedo, M.L., Planning and scheduling in manufacturing and services. Springer, New York, 2005.

[6] Pinedo, M., Chao, X., Operations scheduling. McGraw Hill, New York, 1999.

[7] Sen, T., Sulek, J.M., Dileepan, P., Static scheduling research to minimize weighted and unweighted tardiness: A state-of-the-art survey. Int. J. Prod. Econ. 83 (1): 1–12, 2003.

[8] Koulamas, C., The single-machine total tardiness scheduling problem: Review and extensions. Eur. J. Oper. Res. 202 (1): 1–7, 2010.

[9] Allahverdi, A., The third comprehensive survey on scheduling problems with setup times/costs. Eur. J. Oper. Res. 246 (2): 345–378, 2015.

[10] Muştu, S., Eren, T., The single machine scheduling problem with sequence-dependent setup times and a learning effect on processing times. Appl. Soft Comput. 71, 291–306, 2018.

[11] Tanaka, S., Araki, M., An exact algorithm for the single-machine total weighted tardiness problem with sequence-dependent setup times. Comput. Oper. Res. 40 (1): 344–352, 2013.

[12] Panwalkar, S.S., Dudek, R.A., Smith, M.L., Sequencing research and the industrial scheduling problem. In the Symposium on the Theory of Scheduling and Its Applications. Lecture Notes in Economics and Mathematical Systems (Operations Research, Computer Science, Social Science). pp. 29–38. Ed: by Elmaghraby, S.E. (ed.). Springer, Berlin, Heidelberg, 1973.

[13] Rubin, P.A., Ragatz, G.L., Scheduling in a sequence dependent setup environment with genetic search. Comput. Oper. Res. 22 (1): 85–99, 1995.

[14] Muştu, S., Eren, T., Maximum completion time under a learning effect in the permutation flowshop scheduling problem. Int. J. Ind. Eng. Theory Appl. Pract. 25 (2): 156–174, 2018.

[15] Badiru, A.B., Computational Survey of Univariate and Multivariate Learning Curve Models. IEEE Trans. Eng. Manag. 39 (2): 176–188, 1992.

[16] Biskup, D., A state-of-the-art review on scheduling with learning effects. Eur. J. Oper. Res. 188 (2): 315–329, 2008.

[17] Anzanello, M.J., Fogliatto, F.S., Learning curve models and applications: Literature review and research directions. Int. J. Ind. Ergon. 41 (5): 573–583, 2011.

[18] Azzouz, A., Ennigrou, M., Ben Said, L., Scheduling problems under learning effects: classification and cartography. Int. J. Prod. Res. 56 (4): 1642–1661, 2018.

[19] Stewart, R.D., Wyskida, R.M., Johannes, J.D., Cost Estimator's Reference Manual. John Wiley & Sons, New Jersey, 1995.

[20] Yelle, L.E., The learning curve: Historical review and comprehensive survey. Decis. Sci. 10 (2): 302–328, 1979.

[21] Biskup, D., Single-machine scheduling with learning considerations. Eur. J. Oper. Res. 115 (1): 173–178, 1999.

[22] Shtub, A., Levin, N., Globerson, S., Learning and forgetting industrial skills: an experimental model. Int. J. Hum. factors Manuf. 3 (3): 293–305, 1993.

[23] Li, X., Yang, Z., Ruiz, R., Chen, T., Sui, S., An iterated greedy heuristic for no-wait flow shops with sequence dependent setup times, learning and forgetting effects. Inf. Sci. (Ny). 453, 408–425, 2018.

[24] Coffman, E.G., Bruno, J.L., Computer and job-shop scheduling theory. John Wiley & Sons, New Jersey, 1976.

[25] Gawiejnowicz, S., A note on scheduling on a single processor with speed dependent on a number of executed jobs. Inf. Process. Lett. 57 (6): 297–300, 1996.

[26] Baker, K.R., Trietsch, D., Principles of Sequencing and Scheduling. John Wiley & Sons, New Jersey, 2018.

[27] Gawiejnowicz, S., Time-dependent scheduling. Springer Science & Business Media, Berlin, Heidelberg, 2008.

[28] Haupt, R., A survey of priority rule-based scheduling. OR Spektrum. 11 (1): 3–16, 1989.

[29] Graham, R.L., Lawler, E.L., Lenstra, J.K., Kan, A.H.G.R., Optimization and approximation in deterministic sequencing and scheduling: A survey. Ann. Discret. Math. 5, 287–326, 1979.

[30] Jouglet, A., Carlier, J., Dominance rules in combinatorial optimization problems. Eur. J. Oper. Res. 212 (3): 433–444, 2011.

[31] Garey, M.R., Johnson, D.S., Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, San Francisco, 1979.

[32] Arora, S., Barak, B., Computational complexity: a modern approach. Cambridge University Press, Cambridge, 2009.

[33] Papadimitriou, C.H., Computational complexity. John Wiley & Sons, New Jersey, 2003.

[34] Cook, S.A., The complexity of theorem-proving procedures. Proceedings of the third annual ACM symposium on Theory of computing, May, 1971, Shaker Heights, Ohio, USA, pp. 151–158, 1971.

[35] Karp, R.M., Reducibility among combinatorial problems. In the Complexity of computer computations. pp. 85–103. Ed: by Miller, R.E., Thatcher, J.W., and Bohlinger, J.D. (eds.). Springer, Boston, MA, 1972.

[36] Bellman, R., Dynamic Programming. Princeton University Press, Princeton, 1957.

[37] Alidaee, B., Womer, N.K., Scheduling with time dependent processing times: Review and extensions. J. Oper. Res. Soc. 50 (7): 711–720, 1999.

[38] Globerson, S., Levin, N., Shtub, A., The impact of breaks on forgetting when performing a repetitive task. IIE Trans. 21 (4): 376–381, 1989.

[39] Gupta, J.N.D., Gupta, S.K., Single facility scheduling with nonlinear processing times. Comput. Ind. Eng. 14 (4): 387–393, 1988.

[40] Browne, S., Yechiali, U., Scheduling deteriorating jobs on a single processor. Oper. Res. 38 (3): 495–498, 1990.

[41] Badiru, A.B., Multivariate analysis of the effect of learning and forgetting on product quality. Int. J. Prod. Res. 33 (3): 777–794, 1995.

[42] Cheng, T.C.E., Ding, Q., Lin, B.M.T., A concise survey of scheduling with time-dependent processing times. Eur. J. Oper. Res. 152 (1): 1–13, 2004.

[43] Sule, D.R., The effect of alternate periods of learning and forgetting on economic manufacturing quantity. AIIE Trans. 10 (3): 338–343, 1978.

[44] Conley, P., Experience curves as a planning tool. IEEE Spectr. 7 (6): 63–68, 2009.

[45] Wright, T.P., Factors Affecting the Cost of Airplanes. J. Aeronaut. Sci. 3 (4): 122–128, 1936.

[46] Chakravarty, A.K., Line Balancing With Task Learning Effects. IIE Trans. 20 (2): 186–193, 1988.

[47] Neidigh, R.O., Harrison, T.P., Optimising lot sizing and order scheduling with non-linear production rates. Int. J. Prod. Res. 48 (8): 2279–2295, 2010.

[48] Franceschini, F., Galetto, M., An empirical investigation of learning curve composition laws for quality improvement in complex manufacturing plants. J. Manuf. Technol. Manag. 15 (7): 687–699, 2004.

[49] Smunt, T.L., Watts, C.A., Improving operations planning with learning curves: Overcoming the pitfalls of “messy” shop floor data. J. Oper. Manag. 21 (1): 93–107, 2003.

[50] Smunt, T.L., A comparison of learning curve analysis and moving average ratio analysis for detailed operational planning. Decis. Sci. 17 (4): 475–495, 1986.

[51] Smunt, T.L., Incorporating learning curve analysis into medium-term capacity planning procedures: a simulation experiment. Manage. Sci. 32 (9): 1164–1176, 1986.

[52] Lapré, M.A., Mukherjee, A.S., Van Wassenhove, L.N., Behind the learning curve: linking learning activities to waste reduction. Manage. Sci. 46 (5): 597–611, 2000.

[53] Glock, C.H., Grosse, E.H., Jaber, M.Y., Smunt, T.L., Applications of learning curves in production and operations management: A systematic literature review. Comput. Ind. Eng. 131, 422–441, 2019.

[54] Baloff, N., Extension of the Learning Curve — Some Empirical Results. J. Oper. Res. Soc. 22 (4): 329–340, 1971.

[55] Carlson, J.G.H., Cubic learning curves-precision tool for labor estimating. Manuf. Eng. Manag. 71 (5): 22–25, 1973.

[56] Jong, J.R., The effects of increasing skill on cycle time and its consequences for time standards. Ergonomics. 1 (1): 51–60, 1957.

[57] Levy, F.K., Adaptation in the production process. Manage. Sci. 11 (6): 136–154, 1965.

[58] Pegels, C.C., On startup or learning curves: An expanded view. AIIE Trans. 1 (3): 216–222, 1969.

[59] Bevis, F.W., Towill, D.R., Finnlear, C., Prediction of operator performance during learning of repetitive tasks. Int. J. Prod. Res. 8 (4): 293–305, 1970.

[60] Knecht, G.R., Costing, technological growth and generalized learning curves. Oper. Res. Q. 25 (3): 487–491, 1974.

[61] Kvalseth, T.O., The effect of task complexity on the human learning function. Int. J. Prod. Res. 16 (5): 427–435, 1978.

[62] Mazur, J.E., Hastie, R., Learning as accumulation: A reexamination of the learning curve. Psychol. Bull. 85 (6): 1256–1274, 1978.

[63] Cheng, T.C.E., Wang, G., Single machine scheduling with learning effect considerations. Ann. Oper. Res. 98, 273–290, 2000.

[64] Howick, S., Eden, C., Learning in disrupted projects: On the nature of corporate and personal learning. Int. J. Prod. Res. 45 (12): 2775–2797, 2007.

[65] Steedman, I., Some improvement curve theory. Int. J. Prod. Res. 8 (3): 189–206, 1970.

[66] Jaber, M.Y., Bonney, M., Lot sizing with learning and forgetting in set-ups and in product quality. Int. J. Prod. Econ. 83 (1): 95–111, 2003.

[67] Boone, T., Ganeshan, R., Hicks, R.L., Learning and knowledge depreciation in professional services. Manage. Sci. 54 (7): 1231–1236, 2008.

[68] Sule, D.R., Effect of learning and forgetting on economic lot size scheduling problem. Int. J. Prod. Res. 21 (5): 771–786, 1983.

[69] Smunt, T.L., The impact of worker forgetting on production scheduling. Int. J. Prod. Res. 25 (5): 689–701, 1987.

[70] Bailey, C.D., Forgetting and the Learning Curve: A Laboratory Study. Manage. Sci. 35 (3): 340–352, 1989.

[71] Wixted, J.T., Ebbesen, E.B., On the Form of Forgetting. Psychol. Sci. 2 (6): 409–415, 1991.

[72] Dar-El, E.M., Ayas, K., Gilad, I., Predicting performance times for long cycle time tasks. IIE Trans. 27 (3): 272–281, 1995.

[73] Jaber, M.Y., Bonney, M., Production breaks and the learning curve: The forgetting phenomenon. Appl. Math. Model. 20 (2): 162–169, 1996.

[74] Salameh, M.K., Abdul-Malak, M.A.U., Jaber, M.Y., Mathematical modelling of the effect of human learning in the finite production inventory model. Appl. Math. Model. 17 (11): 613–615, 1993.

[75] Jaber, M.Y., Bonney, M., A comparative study of learning curves with forgetting. Appl. Math. Model. 21 (8): 523–531, 1997.

[76] Carlson, J.G., Rowe, A.J., How Much Does Forgetting Cost? Ind Eng. 8 (9): 40–47, 1976.

[77] Elmaghraby, S.E., Economic manufacturing quantities under conditions of learning and forgetting (EMQ/LaF). Prod. Plan. Control. 1 (4): 196–208, 1990.

[78] Chiu, H.N., Discrete time-varying demand lot-sizing models with learning and forgetting effects. Prod. Plan. Control. 8 (5): 484–493, 1997.

[79] Mc Creery, J.K., Krajewski, L.J., Improving performance using workforce flexibility in an assembly environment with learning and forgetting effects. Int. J. Prod. Res. 37 (9): 2031–2058, 1999.

[80] Nembhard, D.A., Uzumeri, M. V., Experiential learning and forgetting for manual and cognitive tasks. Int. J. Ind. Ergon. 25 (4): 315–326, 2000.

[81] Shafer, S.M., Nembhard, D.A., Uzumeri, M. V., The Effects of Worker Learning, Forgetting, and Heterogeneity on Assembly Line Productivity. Manage. Sci. 47 (12): 1639–1653, 2001.

[82] Nembhard, D.A., Shafer, S.M., The effects of workforce heterogeneity on productivity in an experiential learning environment. Int. J. Prod. Res. 46 (14): 3909–3929, 2008.

[83] Jaber, M.Y., Kher, H. V., Davis, D.J., Countering forgetting through training and deployment. Int. J. Prod. Econ. 85 (1): 33–46, 2003.

[84] Jaber, M.Y., Sikström, S., A numerical comparison of three potential learning and forgetting models. Int. J. Prod. Econ. 92 (3): 281–294, 2004.

[85] Sikström, S., Jaber, M.Y., The power integration diffusion model for production breaks. J. Exp. Psychol. Appl. 8 (2): 118–126, 2002.

[86] Davidovitch, L., Parush, A., Shtub, A., Simulation-based learning: The learning-forgetting-relearning process and impact of learning history. Comput. Educ. 50 (3): 866–880, 2008.

[87] Teyarachakul, S., Chand, S., Ward, J., Effect of learning and forgetting on batch sizes. Prod. Oper. Manag. 20 (1): 116–128, 2011.

[88] Glock, C.H., Jaber, M.Y., A multi-stage production-inventory model with learning and forgetting effects, rework and scrap. Comput. Ind. Eng. 64 (2): 708–720, 2013.

[89] Biel, K., Glock, C.H., Governing the dynamics of multi-stage production systems subject to learning and forgetting effects: A simulation study. Int. J. Prod. Res. 56 (10): 3439–3461, 2018.

[90] Korytkowski, P., Competences-based performance model of multi-skilled workers with learning and forgetting. Expert Syst. Appl. 77, 226–235, 2017.

[91] Kher, H. V., Malhotra, M.K., Philipoom, P.R., Fry, T.D., Modeling simultaneous worker learning and forgetting in dual resource constrained systems. Eur. J. Oper. Res. 115 (1): 158–172, 1999.

[92] Grosse, E.H., Glock, C.H., Jaber, M.Y., The effect of worker learning and forgetting on storage reassignment decisions in order picking systems. Comput. Ind. Eng. 66 (4): 653–662, 2013.

[93] Allahverdi, A., Soroush, H.M., The significance of reducing setup times/setup costs. Eur. J. Oper. Res. 187 (3): 978–984, 2008.

[94] Trovinger, S.C., Bohn, R.E., Setup Time Reduction for Electronics Assembly: Combining Simple (SMED) and IT-Based Methods. Prod. Oper. Manag. 14 (2): 205–217, 2009.

[95] Koulamas, C., Kyparisis, G.J., Single-machine scheduling problems with past-sequence-dependent setup times. Eur. J. Oper. Res. 187 (3): 1045–1049, 2008.

[96] Salehi Mir, M.S., Rezaeian, J., Mohamadian, H., Scheduling parallel machine problem under general effects of deterioration and learning with past-sequence-dependent setup time: heuristic and meta-heuristic approaches. Soft Comput. 2019.

[97] Fan, W., Pei, J., Liu, X., Pardalos, P.M., Kong, M., Serial-batching group scheduling with release times and the combined effects of deterioration and truncated job-dependent learning. J. Glob. Optim. 71 (1): 147–163, 2018.

[98] Pei, J., Liu, X., Liao, B., Pardalos, P.M., Kong, M., Single-machine scheduling with learning effect and resource-dependent processing times in the serial-batching production. Appl. Math. Model. 58, 245–253, 2018.

[99] Pei, J., Cheng, B., Liu, X., Pardalos, P.M., Kong, M., Single-machine and parallel-machine serial-batching scheduling problems with position-based learning effect and linear setup time. Ann. Oper. Res. 272 (1–2): 217–241, 2019.

[100] Kuo, W.H., Yang, D.L., Minimizing the total completion time in a single-machine scheduling problem with a time-dependent learning effect. Eur. J. Oper. Res. 174 (2): 1184–1190, 2006.

[101] Koulamas, C., Kyparisis, G.J., Single-machine and two-machine flowshop scheduling with general learning functions. Eur. J. Oper. Res. 178 (2): 402–407, 2007.

[102] Zhang, X., Yan, G., Huang, W., Tang, G., A note on machine scheduling with sum-of-logarithm-processing-time-based and position-based learning effects. Inf. Sci. (Ny). 187, 298–304, 2012.

[103] Wang, J.B., Wu, Y. Bin, Ji, P., Erratum: A revision of some single-machine and m-machine flowshop scheduling problems with learning considerations (Information Sciences). Inf. Sci. (Ny). 190, 227–232, 2012.

[104] Wang, J.J.B., Wang, J.J.B., Scheduling jobs with a general learning effect model. Appl. Math. Model. 37 (4): 2364–2373, 2013.

[105] Low, C., Lin, W.Y., Some scheduling problems with time-dependent learning effect and deteriorating jobs. Appl. Math. Model. 37 (20–21): 8865–8875, 2013.

[106] Cheng, T.C.E., Kuo, W.H., Yang, D.L., Scheduling with a position-weighted learning effect based on sum-of-logarithm-processing-times and job position. Inf. Sci. (Ny). 221, 490–500, 2013.

[107] Ji, M., Yao, D., Yang, Q., Cheng, T.C.E., Machine scheduling with DeJong's learning effect. Comput. Ind. Eng. 80 (1): 195–200, 2015.

[108] Zhu, Z., Sun, L., Chu, F., Liu, M., Single-machine group scheduling with resource allocation and learning effect. Comput. Ind. Eng. 60 (1): 148–157, 2011.

[109] Zhu, Z., Chu, F., Sun, L., Liu, M., Single machine scheduling with resource allocation and learning effect considering the rate-modifying activity. Appl. Math. Model. 37 (7): 5371–5380, 2013.

[110] Yin, Y., Xu, D., Huang, X., Notes on “Scheduling problems with general effects of deterioration and learning.” Inf. Sci. (Ny). 195, 296–297, 2012.

[111] Yin, Y., Xu, D., Huang, X., Notes on “some single-machine scheduling problems with general position-dependent and time-dependent learning effects.” Inf. Sci. (Ny). 181 (11): 2209–2217, 2011.

[112] Yin, Y., Xu, D., Huang, X., Erratum to “Single machine past-sequence-dependent setup times scheduling with general position-dependent and time-dependent learning effects” [Appl. Math. Modell. 35 (2011) 1388-1395]. Appl. Math. Model. 35 (12): 5936–5938, 2011.

[113] Yin, Y., Xu, D., Huang, X., Erratum to “Single-machine scheduling problems with both deteriorating jobs and learning effects” [Appl. Math. Modell. 34 (2010) 2831-2839]. Appl. Math. Model. 35 (12): 5933–5935, 2011.

[114] Yin, Y., Xu, D., Some single-machine scheduling problems with general effects of learning and deterioration. Comput. Math. with Appl. 61 (1): 100–108, 2011.

[115] Yin, Y., Wu, W.H., Wu, W.H., Wu, C.C., A branch-and-bound algorithm for a single machine sequencing to minimize the total tardiness with arbitrary release dates and position-dependent learning effects. Inf. Sci. (Ny). 256, 91–108, 2014.

[116] Yin, Y., Wu, C.C., Wu, W.H., Cheng, S.R., The single-machine total weighted tardiness scheduling problem with position-based learning effects. Comput. Oper. Res. 39 (5): 1109–1116, 2012.

[117] Yin, Y., Liu, M., Cheng, T.C.E., Wu, C.C., Cheng, S.R., Four single-machine scheduling problems involving due date determination decisions. Inf. Sci. (Ny). 251, 164–181, 2013.

[118] Yin, N., Kang, L., Wang, X.Y., Single-machine group scheduling with processing times dependent on position, starting time and allotted resource. Appl. Math. Model. 38 (19–20): 4602–4613, 2014.

[119] Yang, S.W., Wan, L., Yin, N., Research on single machine SLK/DIF due window assignment problem with learning effect and deteriorating jobs. Appl. Math. Model. 39 (15): 4593–4598, 2015.

[120] Yang, S.J., Yang, D.L., Single-machine scheduling simultaneous with position-based and sum-of-processing-times-based learning considerations under group technology assumption. Appl. Math. Model. 35 (5): 2068–2074, 2011.

[121] Yang, S.J., Single-machine scheduling problems simultaneously with deterioration and learning effects under deteriorating multi-maintenance activities consideration. Comput. Ind. Eng. 62 (1): 271–275, 2012.

[122] Xingong, Z., Yong, W., Single-machine scheduling CON/SLK due window assignment problems with sum-of-processed times based learning effect. Appl. Math. Comput. 250, 628–635, 2015.

[123] Wu, Y. Bin, Wang, X.Y., Ji, P., A note on "Single-machine scheduling problems with both deteriorating jobs and learning effects." Appl. Math. Model. 36 (12): 6341–6344, 2012.

[124] Wu, C.H., Lai, P.J., Lee, W.C., A note on single-machine scheduling with sum-of-processing-time-based learning and forgetting effects. Appl. Math. Model. 39 (2): 415–424, 2015.

[125] Wu, C.C., Lee, W.C., Liou, M.J., Single-machine scheduling with two competing agents and learning consideration. Inf. Sci. (Ny). 251, 136–149, 2013.

[126] Wang, X.Y., Wang, J.J., Scheduling problems with past-sequence-dependent setup times and general effects of deterioration and learning. Appl. Math. Model. 37 (7): 4905–4914, 2013.

[127] Wang, L., Jin, J., Wang, J.B., Ji, P., Research on scheduling problems with general effects of deterioration and learning. Inf. Sci. (Ny). 307 (6): 89–94, 2015.

[128] Wang, J.B., Liu, L., Wang, C., Single machine SLK/DIF due window assignment problem with learning effect and deteriorating jobs. Appl. Math. Model. 37 (18–19): 8394–8400, 2013.

[129] Wang, J.B., Li, J.X., Single machine past-sequence-dependent setup times scheduling with general position-dependent and time-dependent learning effects. Appl. Math. Model. 35 (3): 1388–1395, 2011.

[130] Wang, J.B., Hsu, C.J., Yang, D.L., Single-machine scheduling with effects of exponential learning and general deterioration. Appl. Math. Model. 37 (4): 2293–2299, 2013.

[131] Vahedi-Nouri, B., Fattahi, P., Rohaninejad, M., Tavakkoli-Moghaddam, R., Minimizing the total completion time on a single machine with the learning effect and multiple availability constraints. Appl. Math. Model. 37 (5): 3126–3137, 2013.

[132] Shen, L., Wu, Y. Bin, Single machine past-sequence-dependent delivery times scheduling with general position-dependent and time-dependent learning effects. Appl. Math. Model. 37 (7): 5444–5451, 2013.

[133] Rudek, R., Exact and parallel metaheuristic algorithms for the single processor total weighted completion time scheduling problem with the sum-of-processing-time based models. Comput. Oper. Res. 46, 91–101, 2014.

[134] Rudek, R., On single processor scheduling problems with learning dependent on the number of processed jobs. Appl. Math. Model. 37 (3): 1523–1536, 2013.

[135] Rudek, R., The single processor total weighted completion time scheduling problem with the sum-of-processing-time based learning model. Inf. Sci. (Ny). 199, 216–229, 2012.

[136] Qian, J., Steiner, G., Fast algorithms for scheduling with learning effects and time-dependent processing times on a single machine. Eur. J. Oper. Res. 225 (3): 547–551, 2013.

[137] Mor, B., Mosheiov, G., Scheduling a deteriorating maintenance activity and due-window assignment. Comput. Oper. Res. 57, 33–40, 2015.

[138] Luo, K., A scheduling model with a more general function of learning effects. Comput. Ind. Eng. 82, 159–166, 2015.

[139] Lu, Y.Y., Wei, C.M., Wang, J.B., Several single-machine scheduling problems with general learning effects. Appl. Math. Model. 36 (11): 5650–5656, 2012.

[140] Lu, Y.Y., Wang, J.J., Huang, X., Scheduling jobs with position and sum-of-processing-time based processing times. Appl. Math. Model. 39 (14): 4013–4021, 2015.

[141] Lu, Y.Y., Wang, J.B., Some single-machine scheduling with sum-of-processing-time-based and job-position-based processing times. Appl. Math. Model. 37 (10–11): 6695–6702, 2013.

[142] Li, D.C., Hsu, P.H., Solving a two-agent single-machine scheduling problem considering learning effect. Comput. Oper. Res. 39 (7): 1644–1651, 2012.

[143] Lee, W.-C., Lai, P.-J., Scheduling problems with general effects of deterioration and learning. Inf. Sci. (Ny). 181 (6): 1164–1170, 2011.

[144] Lai, P.J., Lee, W.C., Single-machine scheduling with learning and forgetting effects. Appl. Math. Model. 37 (6): 4509–4516, 2013.

[145] Jiang, Z., Chen, F., Wu, C., Minimizing the maximum lateness in a single-machine scheduling problem with the normal time-dependent and job-dependent learning effect. Appl. Math. Comput. 218 (18): 9438–9441, 2012.

[146] Jiang, Z., Chen, F., Kang, H., Single-machine scheduling problems with actual time-dependent and job-dependent learning effect. Eur. J. Oper. Res. 227 (1): 76–80, 2013.

[147] Bai, J., Wang, M.Z., Wang, J.B., Single machine scheduling with a general exponential learning effect. Appl. Math. Model. 36 (2): 829–835, 2012.

[148] Bai, J., Li, Z.R., Huang, X., Single-machine group scheduling with general deterioration and learning effects. Appl. Math. Model. 36 (3): 1267–1274, 2012.

[149] Ahmadizar, F., Hosseini, L., Bi-criteria single machine scheduling with a time-dependent learning effect and release times. Appl. Math. Model. 36 (12): 6203–6214, 2012.

[150] Rudek, R., The single machine total weighted completion time scheduling problem with the sum-of-processing time based models: Strongly NP-hard. Appl. Math. Model. 50, 314–332, 2017.

[151] Soper, A.J., Strusevich, V.A., Refined conditions for V-shaped optimal sequencing on a single machine to minimize total completion time under combined effects. J. Sched. 2019.

[152] Azadeh, A., Habibnejad-Ledari, H., Abdolhossein Zadeh, S., Hosseinabadi Farahani, M., A single-machine scheduling problem with learning effect, deterioration and non-monotonic time-dependent processing times. Int. J. Comput. Integr. Manuf. 30 (2–3): 292–304, 2017.

[153] Yin, Y., Xu, D., Huang, X., Notes on “The common due-date early/tardy scheduling problem on a parallel machine under the effects of time-dependent learning and linear and nonlinear deterioration.” Expert Syst. Appl. 38 (7): 9030–9031, 2011.

[154] Yeh, W.C., Lai, P.J., Lee, W.C., Chuang, M.C., Parallel-machine scheduling to minimize makespan with fuzzy processing times and learning effects. Inf. Sci. (Ny). 269, 142–158, 2014.

[155] Wang, X.Y., Wang, J.J., Scheduling deteriorating jobs with a learning effect on unrelated parallel machines. Appl. Math. Model. 38 (21–22): 5231–5238, 2014.

[156] Wang, C., Liu, C., Zhang, Z.H., Zheng, L., Minimizing the total completion time for parallel machine scheduling with job splitting and learning. Comput. Ind. Eng. 97, 170–182, 2016.

[157] Rostami, M., Pilerood, A.E., Mazdeh, M.M., Multi-objective parallel machine scheduling problem with job deterioration and learning effect under fuzzy environment. Comput. Ind. Eng. 85, 206–215, 2015.

[158] Liu, M., Parallel-machine scheduling with past-sequence-dependent delivery times and learning effect. Appl. Math. Model. 37 (23): 9630–9633, 2013.

[159] Lin, Y.K., Fast LP models and algorithms for identical jobs on uniform parallel machines. Appl. Math. Model. 37 (5): 3436–3448, 2013.

[160] Lee, W.C., Chuang, M.C., Yeh, W.C., Uniform parallel-machine scheduling to minimize makespan with position-based learning curves. Comput. Ind. Eng. 63 (4): 813–818, 2012.

[161] Ji, M., Tang, X., Zhang, X., Cheng, T.C.E., Machine scheduling with deteriorating jobs and DeJong's learning effect. Comput. Ind. Eng. 91, 42–47, 2016.

[162] Rudek, R., Parallel machine scheduling with general sum of processing time based models. J. Glob. Optim. 68 (4): 799–814, 2017.

[163] Rudek, R., Scheduling on parallel processors with varying processing times. Comput. Oper. Res. 81, 90–101, 2017.

[164] Duran Toksari, M., Güner, E., Reply to note on the common due-date early/tardy scheduling problem on a parallel machine under the effects of time-dependent learning and linear and nonlinear deterioration. Expert Syst. Appl. 39 (7): 6657–6658, 2012.

[165] Wu, C.C., Wu, W.H., Hsu, P.H., Lai, K., A two-machine flowshop scheduling problem with a truncated sum of processing-times-based learning function. Appl. Math. Model. 36 (10): 5001–5014, 2012.

[166] Wang, X.Y., Zhou, Z., Zhang, X., Ji, P., Wang, J.B., Several flow shop scheduling problems with truncated position-based learning effect. Comput. Oper. Res. 40 (12): 2906–2929, 2013.

[167] Wang, J.J., Zhang, B.H., Permutation flowshop problems with bi-criterion makespan and total completion time objective and position-weighted learning effects. Comput. Oper. Res. 58, 24–31, 2015.

[168] Wang, J.B., Wang, J.J., Flowshop scheduling with a general exponential learning effect. Comput. Oper. Res. 43, 292–308, 2014.

[169] Shiau, Y.R., Tsai, M.S., Lee, W.C., Cheng, T.C.E., Two-agent two-machine flowshop scheduling with learning effects to minimize the total completion time. Comput. Ind. Eng. 87, 580–589, 2015.

[170] Rudek, R., The computational complexity analysis of the two-processor flowshop problems with position dependent job processing times. Appl. Math. Comput. 221, 819–832, 2013.

[171] Rudek, A., Rudek, R., Makespan minimization flowshop with position dependent job processing times - Computational complexity and solution algorithms. Comput. Oper. Res. 40 (8): 2071–2082, 2013.

[172] Qin, H., Zhang, Z.H., Bai, D., Permutation flowshop group scheduling with position-based learning effect. Comput. Ind. Eng. 92, 1–15, 2016.

[173] Lu, Y.Y., Research on no-idle permutation flowshop scheduling with time-dependent learning effect and deteriorating jobs. Appl. Math. Model. 40 (4): 3447–3450, 2016.

[174] Liu, Y., Feng, Z., Two-machine no-wait flowshop scheduling with learning effect and convex resource-dependent processing times. Comput. Ind. Eng. 75 (1): 170–175, 2014.

[175] Kuo, W.H., Hsu, C.J., Yang, D.L., Worst-case and numerical analysis of heuristic algorithms for flowshop scheduling problems with a time-dependent learning effect. Inf. Sci. (Ny). 184 (1): 282–297, 2012.

[176] He, H., Minimization of maximum lateness in an m-machine permutation flow shop with a general exponential learning effect. Comput. Ind. Eng. 97, 73–83, 2016.

[177] Wu, C.C., Wang, D.J., Cheng, S.R., Chung, I.H., Lin, W.C., A two-stage three-machine assembly scheduling problem with a position-based learning effect. Int. J. Prod. Res. 56 (9): 3064–3079, 2018.

[178] Mor, B., Mosheiov, G., Shapira, D., Flowshop scheduling with learning effect and job rejection. J. Sched. 2019.

[179] Chung, Y.H., Tong, L.I., Bi-criteria minimization for the permutation flowshop scheduling problem with machine-based learning effects. Comput. Ind. Eng. 63 (1): 302–312, 2012.

[180] Li, L., Yan, P., Ji, P., Wang, J.B., Scheduling jobs with simultaneous considerations of controllable processing times and learning effect. Neural Comput. Appl. 29 (11): 1155–1162, 2018.

[181] Yin, N., Single machine due window assignment resource allocation scheduling with job-dependent learning effect. J. Appl. Math. Comput. 56 (1–2): 715–725, 2018.

[182] Wang, J.B., Liang, X.X., Group scheduling with deteriorating jobs and allotted resource under limited resource availability constraint. Eng. Optim. 51 (2): 231–246, 2019.

[183] Wang, J.B., Gao, M., Wang, J.J., Liu, L., He, H., Scheduling with a position-weighted learning effect and job release dates. Eng. Optim. 2019.

[184] Liu, S., Liu, X., Pei, J., Pardalos, P.M., Song, Q., Parallel-batching machines scheduling problem with a truncated time-dependent learning effect via a hybrid CS-JADE algorithm. Optim. Methods Softw. 35 (1): 116–141, 2020.

[185] Kong, M., Liu, X., Pei, J., Cheng, H., Pardalos, P.M., A BRKGA-DE algorithm for parallel-batching scheduling with deterioration and learning effects on parallel machines under preventive maintenance consideration. Ann. Math. Artif. Intell. 2018.

[186] Przybylski, B., A new model of parallel-machine scheduling with integral-based learning effect. Comput. Ind. Eng. 121, 189–194, 2018.

[187] Liao, B., Song, Q., Pei, J., Yang, S., Pardalos, P.M., Parallel-machine group scheduling with inclusive processing set restrictions, outsourcing option and serial-batching under the effect of step-deterioration. J. Glob. Optim. 2018.

[188] Expósito-Izquierdo, C., Angel-Bello, F., Melián-Batista, B., Alvarez, A., Báez, S., A metaheuristic algorithm and simulation to study the effect of learning or tiredness on sequence-dependent setup times in a parallel machine scheduling problem. Expert Syst. Appl. 117, 62–74, 2019.

[189] Wu, C.C., Lin, W.C., Zhang, X., Chung, I.H., Yang, T.H., Lai, K., Tardiness minimisation for a customer order scheduling problem with sum-of-processing-time-based learning effect. J. Oper. Res. Soc. 70 (3): 487–501, 2019.

[190] Shahvari, O., Logendran, R., A comparison of two stage-based hybrid algorithms for a batch scheduling problem in hybrid flow shop with learning effect. Int. J. Prod. Econ. 195, 227–248, 2018.

[191] Mousavi, S.M., Mahdavi, I., Rezaeian, J., Zandieh, M., An efficient bi-objective algorithm to solve re-entrant hybrid flow shop scheduling with learning effect and setup times. Oper. Res. 18 (1): 123–158, 2018.

[192] Bai, D., Tang, M., Zhang, Z.H., Santibanez-Gonzalez, E.D., Flow shop learning effect scheduling problem with release dates. Omega. 78, 21–38, 2018.

[193] Gao, F., Liu, M., Wang, J.J., Lu, Y.Y., No-wait two-machine permutation flow shop scheduling problem with learning effect, common due date and controllable job processing times. Int. J. Prod. Res. 56 (6): 2361–2369, 2018.

[194] Pargar, F., Zandieh, M., Kauppila, O., Kujala, J., The Effect of Worker Learning on Scheduling Jobs in a Hybrid Flow Shop: A Bi-Objective Approach. J. Syst. Sci. Syst. Eng. 27 (3): 265–291, 2018.

[195] Geng, X.N., Wang, J.B., Bai, D., Common due date assignment scheduling for a no-wait flowshop with convex resource allocation and learning effect. Eng. Optim. 51 (8): 1301–1323, 2019.

[196] Sun, X., Geng, X.N., Wang, J.B., Liu, F., Convex resource allocation scheduling in the no-wait flowshop with common flow allowance and learning effect. Int. J. Prod. Res. 57 (6): 1873–1891, 2019.

[197] Shi, H. Bin, Wang, J.B., Research on common due window assignment flowshop scheduling with learning effect and resource allocation. Eng. Optim. 2019.

[198] Zou, Y., Wang, D., Lin, W.C., Chen, J.Y., Yu, P.W., Wu, W.H., Chao, Y.P., Wu, C.C., Two-stage three-machine assembly scheduling problem with sum-of-processing-times-based learning effect. Soft Comput. 2019.

[199] Marichelvam, M.K., Geetha, M., Tosun, Ö., An improved particle swarm optimization algorithm to solve hybrid flowshop scheduling problems with the effect of human factors – A case study. Comput. Oper. Res. 114, 2020.

[200] Yang, W.H., Chand, S., Learning and forgetting effects on a group scheduling problem. Eur. J. Oper. Res. 187 (3): 1033–1044, 2008.

[201] Pan, E., Wang, G., Xi, L., Chen, L., Han, X., Single-machine group scheduling problem considering learning, forgetting effects and preventive maintenance. Int. J. Prod. Res. 52 (19): 5690–5704, 2014.

[202] Liao, W., Zhang, X., Jiang, M., Multi-objective group scheduling optimization integrated with preventive maintenance. Eng. Optim. 49 (11): 1890–1904, 2017.

[203] Li, X., Jiang, Y., Ruiz, R., Methods for Scheduling Problems Considering Experience, Learning, and Forgetting Effects. IEEE Trans. Syst. Man, Cybern. Syst. 48 (5): 743–754, 2018.

[204] Wu, C.H., Lee, W.C., Lai, P.J., Wang, J.Y., Some single-machine scheduling problems with elapsed-time-based and position-based learning and forgetting effects. Discret. Optim. 19, 1–11, 2016.

[205] Allahverdi, A., Gupta, J.N.D., Aldowaisan, T., A review of scheduling research involving setup considerations. Omega. 27 (2): 219–239, 1999.

[206] Allahverdi, A., Ng, C.T., Cheng, T.C.E., Kovalyov, M.Y., A survey of scheduling problems with setup times or costs. Eur. J. Oper. Res. 187 (3): 985–1032, 2008.

[207] Picard, J.-C., Queyranne, M., The Time-Dependent Traveling Salesman Problem and Its Application to the Tardiness Problem in One-Machine Scheduling. Oper. Res. 26 (1): 86–110, 1978.

[208] Ragatz, G.L., A branch-and-bound method for minimum tardiness sequencing on a single processor with sequence dependent setup times. Proceedings of the twenty-fourth annual meeting of the Decision Sciences Institute, November, 1993, pp. 1375–1377, 1993.

[209] Luo, X., Chu, C., Wang, C., Some dominance properties for single-machine tardiness problems with sequence-dependent setup. Int. J. Prod. Res. 44 (17): 3367–3378, 2006.

[210] Luo, X., Chu, F., A branch and bound algorithm of the single machine schedule with sequence dependent setup times for minimizing total tardiness. Appl. Math. Comput. 183 (1): 575–588, 2006.

[211] Bigras, L.P., Gamache, M., Savard, G., The time-dependent traveling salesman problem and single machine scheduling problems with sequence dependent setup times. Discret. Optim. 5 (4): 685–699, 2008.

[212] Kim, S.Y., Lee, Y.H., Agnihotri, D., A hybrid approach to sequencing jobs using heuristic rules and neural networks. Prod. Plan. Control. 6 (5): 445–454, 1995.

[213] Lee, Y.H., Bhaskaran, K., Pinedo, M., A heuristic to minimize the total weighted tardiness with sequence-dependent setups. IIE Trans. 29 (1): 45–52, 1997.

[214] Vepsalainen, A.P.J., Morton, T.E., Priority Rules for Job Shops with Weighted Tardiness Costs. Manage. Sci. 33 (8): 1035–1047, 2008.

[215] Ow, P.S., Morton, T.E., The Single Machine Early/Tardy Problem. Manage. Sci. 35 (2): 177–191, 1989.

[216] Raman, N., Rachamadugu, R. V., Talbot, F.B., Real-time scheduling of an automated manufacturing center. Eur. J. Oper. Res. 40 (2): 222–242, 1989.

[217] Tan, K.C., Narasimhan, R., Minimizing tardiness on a single processor with sequence-dependent setup times: A simulated annealing approach. Omega. 25 (6): 619–634, 1997.

[218] Armentano, V.A., Mazzini, R., A genetic algorithm for scheduling on a single machine with set-up times and due dates. Prod. Plan. Control. 11 (7): 713–720, 2000.

[219] Tan, K.C., Narasimhan, R., Rubin, P.A., Ragatz, G.L., A comparison of four methods for minimizing total tardiness on a single processor with sequence dependent setup times. Omega. 28 (3): 313–326, 2000.

[220] França, P.M., Mendes, A., Moscato, P., A memetic algorithm for the total tardiness single machine scheduling problem. Eur. J. Oper. Res. 132 (1): 224–242, 2001.

[221] Gagné, C., Price, W.L., Gravel, M., Comparing an ACO algorithm with other heuristics for the single machine scheduling problem with sequence-dependent setup times. J. Oper. Res. Soc. 53 (8): 895–906, 2002.

[222] Gupta, S.R., Smith, J.S., Algorithms for single machine total tardiness scheduling with sequence dependent setups. Eur. J. Oper. Res. 175 (2): 722–739, 2006.

[223] Subramanian, A., Battarra, M., Potts, C.N., An Iterated Local Search heuristic for the single machine total weighted tardiness scheduling problem with sequence-dependent setup times. Int. J. Prod. Res. 52 (9): 2729–2742, 2014.

[224] Eren, T., Güner, E., A bicriteria scheduling with sequence-dependent setup times. Appl. Math. Comput. 179 (1): 378–385, 2006.

[225] González, M.A., Palacios, J.J., Vela, C.R., Hernández-Arauzo, A., Scatter search for minimizing weighted tardiness in a single machine scheduling with setups. J. Heuristics. 23 (2–3): 81–110, 2017.

[226] Liao, C.J., Juan, H.C., An ant colony optimization for single-machine tardiness scheduling with sequence-dependent setups. Comput. Oper. Res. 34 (7): 1899–1909, 2007.

[227] Lin, S.W., Ying, K.C., Solving single-machine total weighted tardiness problems with sequence-dependent setup times by meta-heuristics. Int. J. Adv. Manuf. Technol. 34 (11–12): 1183–1190, 2007.

[228] Anghinolfi, D., Paolucci, M., A new ant colony optimization approach for the single machine total weighted tardiness scheduling problem. Int. J. Oper. Res. 5 (1): 44–60, 2008.

[229] Anghinolfi, D., Paolucci, M., A new discrete particle swarm optimization approach for the single-machine total weighted tardiness scheduling problem with sequence-dependent setup times. Eur. J. Oper. Res. 193 (1): 73–85, 2009.

[230] Tasgetiren, M.F., Pan, Q.K., Liang, Y.C., A discrete differential evolution algorithm for the single machine total weighted tardiness problem with sequence dependent setup times. Comput. Oper. Res. 36 (6): 1900–1915, 2009.

[231] Ying, K.C., Lin, S.W., Huang, C.Y., Sequencing single-machine tardiness problems with sequence dependent setup times using an iterated greedy heuristic. Expert Syst. Appl. 36 (3 PART 2): 7087–7092, 2009.

[232] Kirlik, G., Oguz, C., A variable neighborhood search for minimizing total weighted tardiness with sequence dependent setup times on a single machine. Comput. Oper. Res. 39 (7): 1506–1520, 2012.

[233] Guo, Q., Tang, L., An improved scatter search algorithm for the single machine total weighted tardiness scheduling problem with sequence-dependent setup times. Appl. Soft Comput. 29, 184–195, 2015.

[234] Xu, H., Lü, Z., Cheng, T.C.E., Iterated local search for single-machine scheduling with sequence-dependent setup times to minimize total weighted tardiness. J. Sched. 17 (3): 271–287, 2014.

[235] Xu, H., Lü, Z., Yin, A., Shen, L., Buscher, U., A study of hybrid evolutionary algorithms for single machine scheduling problem with sequence-dependent setup times. Comput. Oper. Res. 50, 47–60, 2014.

[236] González, M.A., Vela, C.R., An efficient memetic algorithm for total weighted tardiness minimization in a single machine with setups. Appl. Soft Comput. 37, 506–518, 2015.

[237] Subramanian, A., Farias, K., Efficient local search limitation strategy for single machine total weighted tardiness scheduling with sequence-dependent setup times. Comput. Oper. Res. 79, 190–206, 2017.

[238] Eren, T., Güner, E., Minimizing total tardiness in a scheduling problem with a learning effect. Appl. Math. Model. 31 (7): 1351–1361, 2007.

[239] Wu, C.C., Yin, Y., Wu, W.H., Chen, H.M., Cheng, S.R., Using a branch-and-bound and a genetic algorithm for a single-machine total late work scheduling problem. Soft Comput. 20 (4): 1329–1339, 2016.

[240] Toksarı, M.D., Güner, E., Minimizing the earliness/tardiness costs on parallel machine with learning effects and deteriorating jobs: A mixed nonlinear integer programming approach. Int. J. Adv. Manuf. Technol. 38 (7–8): 801–808, 2008.

[241] Toksarı, M.D., Güner, E., Parallel machine scheduling problem to minimize the earliness/tardiness costs with learning effect and deteriorating jobs. J. Intell. Manuf. 21 (6): 843–851, 2010.

[242] Yue, L., Guan, Z., Saif, U., Zhang, F., Wang, H., Hybrid Pareto artificial bee colony algorithm for multi-objective single machine group scheduling problem with sequence-dependent setup times and learning effects. Springerplus. 5 (1): 1593, 2016.

[243] Azadeh, A., Goodarzi, A.H., Kolaee, M.H., Jebreili, S., An efficient simulation–neural network–genetic algorithm for flexible flow shops with sequence-dependent setup times, job deterioration and learning effects. Neural Comput. Appl. 31 (9): 5327–5341, 2019.

[244] Jackson, J.R., Scheduling a production line to minimize maximum tardiness. Management Science Research Project, University of California, Los Angeles, Research Report 43, 1955.

[245] Belegundu, A.D., Chandrupatla, T.R., Optimization concepts and applications in engineering. Cambridge University Press, Cambridge, 2019.

[246] Yang, X.-S., Engineering optimization: an introduction with metaheuristic applications. John Wiley & Sons, New Jersey, 2010.

[247] Schrijver, A., Combinatorial optimization: polyhedra and efficiency. Springer Science & Business Media, Berlin, Heidelberg, 2003.

[248] Edmonds, J., Paths, trees, and flowers. Can. J. Math. 17, 449–467, 1965.

[249] Du, J., Leung, J.Y.-T., Minimizing Total Tardiness on One Machine is NP-Hard. Math. Oper. Res. 15 (3): 483–495, 1990.

[250] Holland, J.H., Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press, Cambridge, USA, 1992.

[251] Goldberg, D.E., Genetic algorithms. Pearson Education, India, 2006.

[252] Maheswaran, R., Ponnambalam, S.G., Aravindan, C., A meta-heuristic approach to single machine scheduling problems. Int. J. Adv. Manuf. Technol. 25 (7): 772–776, 2005.

[253] Chang, P.-C., Hsieh, J.-C., Liu, C.-H., A case-injected genetic algorithm for single machine scheduling problems with release time. Int. J. Prod. Econ. 103 (2): 551–564, 2006.

[254] M'Hallah, R., Minimizing total earliness and tardiness on a single machine using a hybrid heuristic. Comput. Oper. Res. 34 (10): 3126–3142, 2007.

[255] Dehghan, M., Dabaghi, A., Rabbani, M., Amalnick, S., Parast, M.Y., Jolai, F., Genetic algorithm for bi-criteria single machine scheduling problem of minimizing maximum earliness and number of tardy jobs. Appl. Math. Comput. 194 (2): 552–560, 2007.

[256] Chang, P.-C., Chen, S.-S., Fan, C.-Y., Mining gene structures to inject artificial chromosomes for genetic algorithm in single machine scheduling problems. Appl. Soft Comput. 8 (1): 767–777, 2008.

[257] Goldberg, D.E., Genetic algorithms in search, optimization, and machine learning. Addison-Wesley Reading Menlo Park, Boston, USA, 2013.

[258] Kellegöz, T., Toklu, B., Wilson, J., Comparing efficiencies of genetic crossover operators for one machine total weighted tardiness problem. Appl. Math. Comput. 199 (2): 590–598, 2008.

[259] Chou, F. Der, An experienced learning genetic algorithm to solve the single machine total weighted tardiness scheduling problem. Expert Syst. Appl. 36 (2): 3857–3865, 2009.

[260] Soltani, R., Jolai, F., Zandieh, M., Two robust meta-heuristics for scheduling multiple job classes on a single machine with multiple criteria. Expert Syst. Appl. 37 (8): 5951–5959, 2010.

[261] Goldberg, D.E., Deb, K., A Comparative Analysis of Selection Schemes Used in Genetic Algorithms. In the Foundations of Genetic Algorithms. pp. 69–93. Ed: by Rawlins, G.J.E. (ed.). Elsevier, Amsterdam, 1991.

[262] Mladenović, N., Hansen, P., Variable neighborhood search. Comput. Oper. Res. 24 (11): 1097–1100, 1997.

[263] Hansen, P., Mladenović, N., Todosijević, R., Hanafi, S., Variable neighborhood search: basics and variants. EURO J. Comput. Optim. 5 (3): 423–454, 2017.

[264] Kirkpatrick, S., Toulouse, G., Configuration space analysis of travelling salesman problems. J. Phys. 46 (8): 1277–1292, 1985.

[265] Cicirello, V.A., Weighted tardiness scheduling with sequence-dependent setups: A benchmark library. Intelligent Coordination and Logistics Laboratory, February, 2003, Robotics Institute, Carnegie Mellon University, Pittsburgh, pp. 189–198, 2003.

ÖZGEÇMİŞ

Adı Soyadı	:	Settar MUŞTU
Uyruğu	:	T.C.
Doğum Tarihi	:	01.01.1982
Doğum Yeri	:	Gaziantep
Medeni Hali	:	Evli
Yabancı Dil	:	İngilizce
E-mail	:	settarmustu@gmail.com

EĞİTİM

2009-2013	Yüksek Lisans	Kırıkkale Üniversitesi, Fen Bilimleri Enstitüsü, Endüstri Mühendisliği A.B.D., KIRIKKALE.
2000-2004	Lisans	Dokuz Eylül Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği Bölümü, İZMİR.
1993-2000	Lise	Fitnat Nuri Tekerekoğlu Anadolu Lisesi, GAZİANTEP.

İŞ DENEYİMİ

2007-2019	ŞDL Zincir Ltd. Şti. / KIRIKKALE	Fabrika Müdürü
2005-2007	Gülsan Holding A.Ş. / GAZİANTEP	Kalite Güvence Şefi

YAYINLAR (SCI)

1. Muştu, S., Eren, T., The single machine scheduling problem with sequence-dependent setup times and a learning effect on processing times. Applied Soft Computing. 71, 291–306, 2018.

2. Muştu, S., Eren, T., Maximum completion time under a learning effect in the permutation flowshop scheduling problem. International Journal of Industrial Engineering: Theory, Applications and Practice. 25 (2): 156–174, 2018.

YAYINLAR (DİĞER)

1. Muştu, S., Eren, T., Geliş zamanlarının farklı olduğu öğrenme etkili tek makine çizelgelemede toplam gecikmenin çözümü. Sosyal Bilimler Araştırma Dergisi. 4 (3): 11–34, 2015.

BİLDİRİLER

1. Muştu, S., Eren, T., İşlem süreleri üzerinde öğrenme etkisinin ve sıra bağımlı ayar sürelerinin olduğu tek makine çizelgeleme probleminde toplam ağırlıklı gecikmenin en küçüklemesi. 37. Yöneylem Araştırması ve Endüstri Mühendisliği Ulusal Kongresi, Temmuz, 2017, Yıldız Teknik Üniversitesi, İstanbul.

2. Muştu, S., Eren, T., Öğrenme etkili n iş m makineli akış tipi çizelgeleme probleminde maksimum tamamlanma zamanı en küçüklemesi. 16. Uluslararası Ekonometri, Yöneylem Araştırması ve İstatistik Sempozyumu, Mayıs, 2015, Trakya Üniversitesi, Edirne.

3. Muştu, S., Eren, T., Öğrenme etkili ve farklı geliş zamanlı tek makineli çizelgelemede toplam gecikme probleminin çözümü. 16. Uluslararası Ekonometri, Yöneylem Araştırması ve İstatistik Sempozyumu, Mayıs, 2015, Trakya Üniversitesi, Edirne.