KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

DOKTORA TEZİ

YAYILMA ZAMANLI GENELLEŞTİRİLMİŞ OPERASYONEL SABİT İŞ ÇİZELGELEME PROBLEMİ İÇİN BİR HİBRİT METASEZGİSEL MODEL ÖNERİSİ

AHMET CİHAN

KOCAELİ 2015

KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

DOKTORA TEZİ

YAYILMA ZAMANLI GENELLEŞTİRİLMİŞ OPERASYONEL SABİT İŞ ÇİZELGELEME PROBLEMİ İÇİN BİR HİBRİT METASEZGİSEL MODEL ÖNERİSİ

AHMET CİHAN

Prof.Dr.Nilgün FIĞLALI **Danışman, Kocaeli Üniv.**	....
Prof.Dr.Zerrin ALADAĞ **Jüri Üyesi, Kocaeli Üniv.**	....
Prof.Dr.Coşkun ÖZKAN **Jüri Üyesi, YTÜ**	....
Yrd.Doç.Dr.Gülşen AYDIN KESKİN **Jüri Üyesi, Kocaeli Üniv.**	....
Yrd.Doç.Dr.Ümit TERZİ **Jüri Üyesi, Beykent Üniv.**	....

Tezin Savunulduğu Tarih: 24.11.2015

ÖNSÖZ VE TEŞEKKÜR

Bu çalışmada, iş seçimi konusunda gerçek hayat uygulamalarında karşılaşılabilen yayılma zamanlı genelleştirilmiş operasyonel sabit iş çizelgeleme problemleri ele alınmıştır.

Çalışmanın tamamında kullanılan bilgisayarı BAP 2011-80 no'lu proje kapsamında sağlayan Kocaeli Üniversitesi'ne ve çalışanlarına teşekkür ederim.

Çalışmanın her aşamasında yardımlarını esirgemeyen danışmanım Prof. Dr. Nilgün Fığlalı'ya, beni her zaman destekleyen anabilim dalı başkanımız Prof. Dr. Alpaslan Fığlalı'ya, çalışmam ile alakalı olsun olmasın sabırla sorularımı cevaplayan Yrd. Doç. Dr. Ümit Terzi'ye teşekkürlerimi sunarım.

Ayrıca benim bu günlere gelmemi sağlayan ve her zaman destekleyen aileme; fikirleri ve bakış açısı ile çalışmalarımı daha farklı boyutlarda ele almamı sağlayan kardeşim Dr. Müh. Onur Cihan'a teşekkürü borç bilirim.

Aralık - 2015 Ahmet CİHAN

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

x^*	: En uygun x yöneyi, (x optimal)
∇	: Gradient, (Eğim)

Kısaltmalar

FIFO	: First In First Out (İlk Giren İlk Çıkar)
NP	: Non-Deterministic Polynomial (Belirsiz Polinom)
P	: Polynomial (Polinom)

YAYILMA ZAMANLI GENELLEŞTİRİLMİŞ OPERASYONEL SABİT İŞ ÇİZELGELEME PROBLEMİ İÇİN BİR HİBRİT METASEZGİSEL MODEL ÖNERİSİ

ÖZET

Bu tez çalışmasında, iş seçimi konusunda gerçek hayat uygulamalarında karşılaşılabilen yayılma zamanlı genelleştirilmiş operasyonel sabit iş çizelgeleme problemleri ele alınmıştır. Bu amaçla literatürde yer alan modeller incelenmiş, yayılma zamanlı operasyonel sabit iş çizelgeleme problemi için matematiksel temelli bir üst sınır modeli önerilmiştir. Literatürde problemin genelleştirilmiş modeli bulunmadığından genelleştirilmiş model için matematiksel model geliştirilmiş, günümüz teknolojisinde modelin çözebildiği problem büyüklüğü bulunmuştur. Matematiksel model ile çözülemeyen problemler için ise literatürde yayılma zamanlı operasyonel sabit iş çizelgeleme problemi için geliştirilmiş olan sezgisel ve metasezgisel yöntem, genelleştirilmiş probleme uyarlanmıştır. Önerilen algoritma ile elde edilen sonuçlar, en uygun çözüm değerinden sapma ve büyük problemlerde çözüm miktarındaki iyileşmeler ile sunulmaktadır.

Anahtar Kelimeler: Metasezgisel Yöntemler, Operasyonel Sabit İş Çizelgeleme, Uyumsuz Makine Kısıtları, Yayılma Zamanı Kısıtları.

A HYBRID METAHEURISTIC MODEL PROPOSAL FOR GENERALIZED OPERATIONAL FIXED JOB SCHEDULING PROBLEM WITH SPREAD TIME CONSTRAINTS

ABSTRACT

In this dissertation, we consider generalized operational fixed job scheduling problems with spread time constraints which finds applications in real world problems. For this purpose, we investigate existing methods in the literature and propose a mathematics based upper bound model for the operational fixed job scheduling problem with spread time constraints. Since there does not exist a generalized model for the problem in the literature, we develop a mathematical model for the generalized model and derive the size for which the model can be solved in today's technology. For the problems which cannot be solved by mathematical model, we adapt the heuristic and metaheuristic methods that are developed for spread time operational fixed job scheduling problems to the generalized problem. The results of the proposed algorithm are presented with the derivation from the optimal solution and its improvement for the large scale problems.

Keywords: Metaheuristic Methods, Operational Fixed Job Scheduling, Incompatible Machine Constraints, Spread Time Constraints.

GİRİŞ

İş çizelgeleme problemleri üretim, yönetim, planlama gibi alanlarda sıklıkla karşılaşılan problemlerdendir. Bir ya da birden fazla amacın gerçekleştirilmesi hedefi doğrultusunda, mevcut kaynakların yapılması gereken işlere nasıl atanması gerektiği kararının verildiği probleme iş çizelgeleme problemi adı verilmektedir (Pinedo, 2002). İş çizelgeleme problemleri; işlerin geliş zamanlarını, işlerin kaynaklarda işlenme sürelerini, işlerin teslim zamanlarını, işlerin işlenebilecekleri kaynakların özelliklerini, işlerin kaynaklarda nasıl ve ne zaman işlenebilecekleri bilgilerini içermektedir.

Gerek pratik, gerekse teorik öneminden dolayı oldukça fazla çalışılmış olunan iş çizelgeleme problemleri, problemin yapısına, kısıtlarına ve amaçlarına göre farklı biçimlerde ifade edilmekte ve farklı algoritmalar ile çözülmektedirler. Pratik kullanım alanlarında birçok belirsizlik içeren iş çizelgeleme problemleri, belirsizliklerin farklı dağılımlar yardımı ile modellenmesi durumunda rastgele(stochastic) çizelgeleme problemleri, her tür verinin kesin olarak bilindiğinin varsayıldığı durumlarda ise gerekirci(deterministic) çizelgeleme problemleri olarak sınıflandırılmaktadırlar.

İş çizelgeleme problemleri, hesapsal karmaşıklık(complexity) açısından birçok durumda belirsiz polinom zamanda(non-deterministic polynomial time) çözülebilen problemler (NP) kümesinde olmasının yanı sıra iş çizelgeleme problemlerinin bazı özel durumlarının polinom zamanda(polynomial time) çözülebilen problemler (P) kümesinde yer aldığı bilinmektedir.

Çalışmanın ikinci bölümünde iş çizelgeleme problemleri tanımlanarak, sınıflandırılması yapılmaktadır. Ele alınan problemin sınıflandırma içerisinde nerede olduğu, problemin karmaşıklığının ne olduğu ile ilgili bilgiler verilmektedir.

Çalışmanın üçüncü bölümünde matematiksel en uygunlama(optimization) problemlerinin önemli olanları ve çözümde kullanılan temel matematiksel yöntemler genel ifadeleri ile verilmektedir. Bu bölümde ayrıca çalışmada üst sınır için kullanılabileceği belirtilen yarı tanımlı programlama modelinin matematiksel ifadesi açıklanmaktadır.

Çalışmanın dördüncü bölümünde büyük en uygunlama problemlerinin çözümünde kullanılan sezgisel(heuristic) ve metasezgisel(metaheuristic) yöntemler açıklanmaktadır.

Çalışmanın beşinci bölümünde, metasezgisel yöntemlerin parametrelerinin problem büyüklüğüne göre belirlenmesine yönelik yöntemler verilmektedir.

Çalışmanın altıncı bölümünde literatürde tanımlanan problem, probleme dair açgözlü sezgisel ve metasezgisel algoritmalar verilmekte, geliştirilen yarı tanımlı programlama gevşetme modeli verilmektedir. Ayrıca genelleştirilmiş probleme dair matematik programlama modeli ve literatürde bulunan genetik algoritmadan geliştirilen metasezgisel algoritma da bu bölümde verilmektedir.

Çalışmanın yedinci bölümünde yayılma zamanlı operasyonel sabit iş çizelgeleme problemleri için üst sınır bulmakta kullanılabilecek yarı tanımlı programlama modelinden elde edilen sonuçlar ve bu sonuçların değerlendirilmesi verilmektedir. Daha sonra ise genelleştirilmiş problem için geliştirilen yöntemin küçük problemler ile parametrelerinin belirlenmesi ve küçük problemlerde elde edilen en uygun çözümden uzaklık başarımı verilmektedir. Ayrıca büyük problemlerde elde edilen iyileştirme yüzdeleri ve tek bir problem için uzun süreli çalışma sonuçları sunulmaktadır.

1. İŞ ÇİZELGELEME PROBLEMLERİ

İş çizelgeleme problemleri gerçek hayatta çok değişik biçimlerde karşılaşılan problem türlerindendir. Farklı amaçlar ve farklı koşullar doğrultusunda farklı şekillerde ifade edilen iş çizelgeleme problemleri, verilerin belirli ve kesin olması durumunda gerekirci çizelgeleme problemleri, verilerin bir dağılıma uygun olması durumunda rastgele iş çizelgeleme problemleri olarak sınıflandırılmaktadırlar.

Gerekirci çizelgeleme problemlerinde kesinlik içeren bilgiler ile çalışıldığından rastgele çizelgeleme problemlerine göre daha kolay modellenebilmektedirler. Gerçek hayatta karşılaşılan birçok çizelgeleme problemi belirsizlikler içermektedir ve bu sebeple rastgele çizelgeleme problemi biçiminde modellenmeleri daha uygundur. Ancak rastgele çizelgeleme problemlerinin modellenmeleri ve çözüm yöntemleri gerekirci çizelgeleme problemlerinin modellenmeleri ve çözüm yöntemleri ile kıyaslandığında daha karmaşıktırlar. Bu sebeple gerçek hayatta karşılaşılan birçok çizelgeleme problemi, kabuller yapılarak gerekirci iş çizelgeleme problemi biçiminde modellenerek çözülmektedir.

Bir diğer tür sınıflandırma ise, problemin koşullarına göre değişken iş sıralama problemleri ve sabit iş sıralama problemleri biçiminde yapılmaktadır.

Çalışma boyunca işler i ve j alt indisleri ile makineler k alt indisi ile j işinin k makinesinde işleneceği süre p_{jk} ile j işinin hazır olma zamanı r_j ile, j işinin teslim zamanı d_j ile, j işinin k makinesindeki getirisi w_{jk} ile ifade edilecektir.

1.1. Gerekirci İş Çizelgeleme Problemleri

Gerekirci iş çizelgeleme problemleri r_j, d_j, p_{jk}, w_j değerlerinin sayıl(scalar) olduğu problemlerdir. Problemler amaçlarına ve kısıtlarına göre farklı biçimlerde ifade edilmektedirler. Birçok durumda gerekirci iş çizelgeleme problemleri (makine türü, makine işleme karakteristiği, amaç) üçlüsü(triplet) ile temsil edilirler.

Literatürde en sık karşılaşılan gerekirci iş çizelgeleme problemleri makine karakteristikleri aşağıda verilmektedir:

Tek makine problemleri (1): Çalışan tek bir makinenin olduğu problem türü. Bu problem türü, diğer problem türlerinin en basit formu olduğundan önem taşımaktadır. Birçok durumda tek makine problemleri polinom zaman karmaşıklığında çözülebildiklerinden dolayı birçok sezgisel yöntemin geliştirilmesinde kullanılmaktadırlar.

Paralel eş makine problemleri (P_m): Birden çok sayıda (m adet) aynı makinenin bulunduğu problem türüdür. Bu problem türünde makineler birbirinin aynısıdır, makineler arasında farklılık gözetilmemektedir.

Paralel farklı makine problemleri (Q_m): Birden çok (m adet) ve farklı özelliklerdeki makinelerin olduğu problem türüdür. Farklı makinelerin bir işlemi işten bağımsız olarak farklı özellikler (hız, teknik v.b.) ile yapabildiğini ifade etmektedir.

Paralel ilişkisiz (bağımsız) makine problemleri (R_m): Birden çok (m adet) ve ilişkisiz makinelerin olduğu problem türüdür. Farklı makinelerin bir işlemi işe bağımlı olarak farklı özellikler ile yapabildiğini ifade etmektedir.

Akış tipi çizelgeleme problemleri (F_m): Birden çok (m adet) makinenin olduğu, işlerin bu makinelerde aynı yol ile işlendiği problem türüdür. Genellikle bu problem türünde, makine kuyruklarında ilk gelen ilk hizmet görür kuralı kullanılmaktadır. Eğer makine kuyruklarında FIFO kuralı kullanılıyor ise problem devşirim (permutation) türü akış tipi çizelgeleme problemi olarak adlandırılmaktadır.

Esnek akış tipi çizelgeleme problemleri (FF_c): Birden çok makinenin olduğu, her bir kademede (c adet) paralel çalışabilen makinelerin olduğu ve işlerin bu kademelerde aynı yol ile işlenmesi gereken problem türüdür. Genellikle bu problem türünde, makine kademeleri kuyruklarında ilk gelen ilk hizmet görür kuralı kullanılmaktadır.

Atölye tipi çizelgeleme problemleri (J_m): Birden çok (m adet) makinenin olduğu, işlerin bu makinelerde işlenme yollarının önceden belirli olduğu problem türüdür. Bu problem türünde bir işin, işlenme yolundaki makinelerde işlenme sırası belirli olmasının yanı sıra aynı iş bir makinede birden çok defa işlenebilmektedir.

Esnek atölye tipi çizelgeleme problemleri (FJ_c): Birden çok makinenin olduğu, her bir kademede (c adet) paralel çalışabilen makinelerin olduğu ve işlerin bu kademelerde işlenme yollarının önceden belirli olduğu problem türüdür. Bu problem türünde bir işin, işlenme yolundaki makinelerde işlenme sırasının belirli olmasının yanı sıra aynı iş bir makine grubunda birden çok defa işlenebilmektedir.

Açık iş çizelgeleme problemleri (O_m): Birden çok (m adet) makinenin olduğu, işlerin bu makinelerin tamamında güzergah gözetmeksizin işlenmesinin gerektiği problem türüdür.

En bilinen gerekirci iş çizelgeleme problemleri makine işleme karakteristikleri aşağıda verilmektedir:

Hazır olma zamanı (r_i): İşlerin hazır olma zamanlarının olduğu ve bu hazır olma zamanlarından önce işlenmeye başlanamayacakları problem türüdür.

Hazırlık süresi (s_{ijk}): İşlerin makinelerde işlenmelerinden önce makinede bir önceki işten o işe geçiş için belirli bir sürenin gerektiği problem türüdür. Bu problem türünde hazırlık süresi makineden makineye değişebileceği gibi işlemden işleme de değişebilmektedir.

Öne alım (prmp): İşlerin makinelerde işlenirken işlemenin durdurularak bir diğer işin makineye alınabileceği problem türüdür. Bu problem türünde durdurulan iş daha sonra tamamlanmak zorundadır.

Teknolojik öncelik (prec): İşlerin makinelerde işlenebilmeleri için bir veya daha fazla işin daha önce işlenmesinin tamamlanmış olması gereken problem türüdür. Akış ve atölye tipi iş sıralama problem türlerinin her işlem adımında önceki işlemlerin tamamlanmış olma gereksiniminden dolayı işlemler açısından teknolojik öncelik kısıtlarına sahiptirler.

Arıza (brkdwn): İşlerin makinelerde işlenmeleri için makinelerin sürekli olarak hazır olmadığı, makine arıza sürelerinin bulunduğu problem türüdür.

Makine uygunluğu (M_j): İşlerin, paralel makine problemindeki bazı makinelere uygun olmadığı ve işlerin sadece uygun olan makinelerde işlenebileceği problem türüdür.

Devşirim (prmu): İşlere ait sıralamanın bir devşirim ile ifade edilebileceği, genellikle FIFO kuralının işletildiği akış tipi çizelgeleme problem türüdür. Bu problem türünde çözüm yöneyi(vector) bir devşirim ile ifade edilmektedir.

Bloklama (block): İşlerin, makine kuyrukları dolayısı ile bir makinedeki işin bitmesine rağmen bir sonra işleneceği makineye gidemediği, işin işlendiği makineden serbest bırakılamadığı problem türüdür. Bu problem türü genellikle, makinelerde kuyruk miktarının sınırlı olduğu akış tipi problemlerde görülmektedir.

Beklemesizlik (nwt): İşlerin, makine kuyruklarında bekleyemediği problem türüdür. Makinelerde bekleme yapılamadığından işlerin başlangıç zamanlarının geciktirilmesi gerekebilmektedir. Bu problem türü genellikle, akış tipi problemlerde görülmektedir.

Yeniden dolaşım (recrc): İşlerin aynı makineye birden çok defa girmesi gereken problem türüdür. Bu problem türü genellikle, atölye tipi problemlerde görülmektedir.

Apaçık (self-explanatory): İşler ile bazı ilgili bilgilerin aynı olduğu problem türüdür. Bu problem türünde;

- Her işin teslim zamanları aynı ise $d_j = d$
- Her işin işlenme süreleri aynı ise $p_j = p$
- Her işin hazır olma zamanları aynı ise $r_j = r$

ile ifade edilir.

En bilinen gerekirci iş çizelgeleme problemleri amaçları aşağıda verilmektedir:

En küçükleme amacı ile;

- En büyük iş tamamlanma süresi (c_{enb}): Çizelgede en son tamamlanacak olan işin en erken zamanda bitmesini amaçlayan problem türüdür.

- İş teslim hedefinden ayrılma süreleri toplamı ($\sum L_j$): Çizelgede işlerin teslim edilmeleri gereken zamanda teslim edilmesini amaçlayan problem türüdür. Bu problem türünde işlerin teslim sürelerinden önce tamamlanmaları durumunda bekleme yapacakları, daha geç tamamlanmaları durumunda müşteri memnuniyetsizliği veya tazminat gibi bazı maliyetlere katlanılacağı varsayılmaktadır.

- İş gecikme süreleri toplamı ($\sum T_j$): Çizelgede işlerin teslim edilmeleri gereken zamandan en az gecikme ile teslim edilmesini amaçlayan problem türüdür. Bu problem türünde sadece müşteri memnuniyetsizliği veya tazminat gibi maliyetlere katlanılacağı varsayılmaktadır.

- Geciken iş sayısı toplamı ($\sum U_j$): Çizelgede işlerin teslim edilmeleri gereken zamanda teslim edilmesini, en az sayıda işin gecikmesini amaçlayan problem türüdür. Bu problem türünde, gecikme olması durumunda işin ne zaman teslim edildiğinin müşteri açısından bir öneminin olmadığı düşünülmektedir.

- En büyük teslim hedefinden ayrılma süresi (L_{enb}): Çizelgede işlerin teslim hedefinden en büyük sapmasını mümkün olduğunca azaltılmasını amaçlayan problem türü.

- Toplam ağırlıklı tamamlanma süresi ($\sum w_j c_j$): Çizelgede işlerin tamamlanma sürelerinin belirli w_j ağırlıkları ile ağırlıklandırılması ve bu ağırlıkların toplanması ile oluşan değerin en küçük olmasını amaçlayan problem türü. Bu problem türünde erken tamamlanması gereken işler ile erken tamamlanması çok önemli olmadığı düşünülen işler arasında bir ayrım yapılmaktadır.

- Toplam ağırlıklı iş gecikme süresi ($\sum w_j T_j$): Çizelgede işlerin gecikme sürelerinin belirli w_j ağırlıkları ile ağırlıklandırılması ve bu ağırlıkların toplanması ile oluşan değerin en küçük olmasını amaçlayan problem türü. Bu problem türünde işin gecikmesinin önemli olduğu ve çok önemli olmadığı durumlar arasında bir ayrım yapılmaktadır.

- Toplam ağırlıklı geciken iş sayısı ($\sum w_j U_j$): Çizelgede işlerin teslim edilmeleri gereken zamanda teslim edilememesinin w_j ağırlığı ile ağırlıklandırılması ve bu ağırlıkların toplanması ile oluşan değerin en küçük olmasını amaçlayan problem türü. Bu problem türünde, gecikme olması durumunda hangi işin gecikmemesinin daha uygun olabileceği konusunda bir ayrım yapılmaktadır.

- Makine sayısı (m): Çizelgede kullanılacak işlerin tamamının yapılmasının zorunlu olduğu problemlerde kullanılacak makine sayısının en küçük olmasını amaçlayan problem türü.

En büyükleme amacı ile;

- İş ağırlıkları toplamı ($\sum w_j$): Çizelgede alınmış olan j işlerine ait getiri miktarları toplamını en büyük yapmayı amaçlayan problem türü.

Aralık çizelgeleme problemleri, gerekirci iş çizelgeleme problemlerinin bir alt dalı olarak, işlerin işlenmeye hazır olma (r_j) zamanlarının ve işlerin işlenmeleri durumunda bitecekleri (d_j) zamanların gerekirci olduğu ve her bir işin belirli bir getirisi veya ağırlığı (w_j) olduğu durumlarda karşılaşılan problemler olarak tanımlanmaktadır. Aralık çizelgeleme problemleri, işin işlenme süresi ile işlenebileceği zaman aralığına göre değişken iş çizelgeleme problemleri ve sabit iş çizelgeleme problemleri biçiminde incelenmektedir.

1.1.1. Değişken iş çizelgeleme problemleri

Değişken iş çizelgeleme problemleri işin hazır olma zamanı (r_j) ile teslim zamanı (d_j) arasındaki farkın ($d_j - r_j$), işin yapılması için gerekli olan işlem süresinden (p_j) büyük olduğu problemlerdir. Bu problemlerde kullanılacak kaynak seçilirken işin belirli bir zaman penceresi içerisinde istenilen bir zamandan önce başlayamaması durumu söz konusu olmaktadır. Değişken iş çizelgeleme problemlerine uygulamada rastlanmakta olup, akış türü iş çizelgeleme problemleri ve atölye türü iş çizelgeleme problemleri bu kategoride ele alınmaktadır.

1.1.2. Sabit iş çizelgeleme problemleri

Sabit iş çizelgeleme problemleri ile literatürde, çizelgeleme problemlerinin özelleşmiş bir alt dalı olarak karşılaşılmaktadır (Rossi ve diğerleri, 2010; Eliiyi ve Azizoğlu, 2006). Eğer her bir j işi için işlem süresi (p_j) işin bitiş zamanı (d_j) ile işin hazır olma zamanı (r_j) arasındaki farka eşit ise - işin hazır olma zamanından sonra bekletilmesi mümkün değil ise - bu tür problemlere sabit iş çizelgeleme problemi adı verilmektedir.

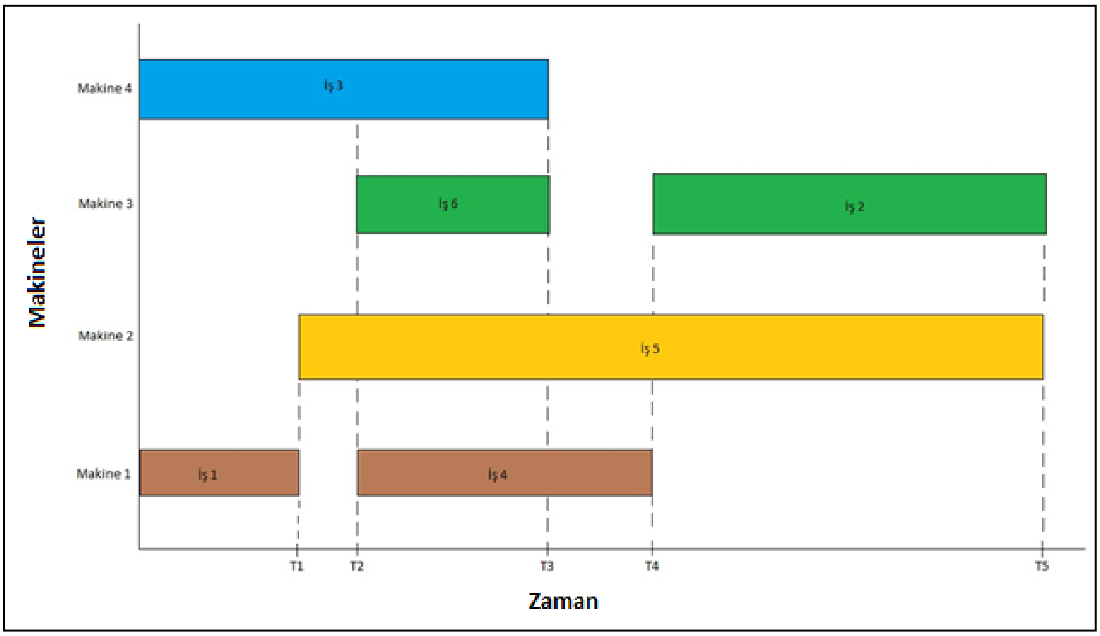
Literatürde sabit iş çizelgeleme problemleri, amaç fonksiyonunun yapısına göre de kendi içerisinde ikiye ayrılmaktadır. Eğer başlama ve bitiş zamanları sabit olan işlerin tamamının yapılması zorunlu ise, verimlilik için en az sayıda makine ile bu işlerin tamamlanması veya en az maliyet ile amacın gerçekleştirilmesi gerekmektedir. Bütün işlerin yapılma zorunluluğu olan ve gerekli en az sayıda makine sayısının veya en düşük maliyetin bulunmasını amaçlayan bu probleme taktik sabit iş çizelgeleme problemi adı verilmektedir. Sabit iş çizelgeleme probleminin bir diğer türü ise, mevcut makine sayısının sabit olduğu durumda amaç fonksiyonunun en yüksek getirili işlerin seçilmesini gerektiren operasyonel sabit iş çizelgeleme problemidir.

Bu problem türünde, eğer işlerin getiri miktarları (w_j) eşit ise en fazla sayıda işin seçilmesi gerektiği açıktır. Tek bir makinenin bulunduğu durumda problemin polinom zaman karmaşıklığı ile çözülebildiği bilinmektedir.

1.1.2.1.Taktik sabit iş çizelgeleme problemi

Taktik sabit iş çizelgeleme problemi, mevcut işlerin tamamının, işlerin yarıda bırakılmaksızın işlenmesi için gerekli makine sayısının en küçüklenmesini amaçlayan problem türüdür. Problem, standart biçimi ile polinom zaman karmaşıklığı ile çözülebilen (P) problemler sınıfına girmektedir. Aynı zaman diliminde işlenmesi zorunlu olan işlerin en fazla kaç adet olduğunun tespit edilmesi ile problem çözülmektedir.

Örnek bir problem Şekil 1.1 ile verilmektedir. Bu problemde işlenmesi gereken sabit başlangıç ve bitiş zamanına sahip altı adet iş bulunmaktadır. İşlerin başlama ve bitiş süreleri gantt şemasında görülmektedir. Problemde iş 1 ile iş 3'ün 0 ile T1 zamanı arasında örtüştüğü, iş 3 ile iş 5'in T1 ile T3 zamanı arasında örtüştüğü, iş 6 ile iş 3, iş 4 ve iş 5'in T2 ile T3 zamanı arasında örtüştüğü, iş 4 ile iş 5'in T2 ile T4 zamanı arasında örtüştüğü, ve son olarak iş 2 ile iş 5'in T4 ile T5 zamanı arasında örtüştüğü görülmektedir. Kullanılması gereken en az makine sayısının bulunması için problemdeki her işin yapılması zorunlu olduğundan dolayı en çok iş örtüşmesinin olduğu zaman aralığında kaç adet örtüşme olduğunun belirlenmesi, problemin çözümü için yeterli olmaktadır.



Şekil 1.1. Örnek bir taktik sabit iş çizelgeleme problemi

Taktik sabit iş çizelgeleme probleminin, uçuş bakım personeli planlamasında temel model olarak kullanıldığı bilinmektedir (Eliiyi ve diğerleri, 2009). Benzer bir problem yüksek seviyeli elektronik devre sentezi işleminde, elektronik çip tasarımında ortaya çıkmaktadır (Rossi ve diğerleri, 2010).

1.1.2.2.Operasyonel sabit iş çizelgeleme problemi

Operasyonel sabit iş çizelgeleme probleminde, sabit iş çizelgeleme problemlerinde olduğu gibi işlerin başlama ve bitiş zamanlarının yanı sıra mevcut makine sayısı da bilinmektedir. Taktik sabit iş çizelgeleme probleminden farklı olarak işlerin tamamının yapılması zorunluluğu olmayıp, mevcut makineler ile hangi işlerin hangi makineler tarafından yapılacağının seçimi önem taşımaktadır. Problemde sadece bir makine bulunuyor ise, problemin ağ modeli oluşturularak en uzun yol algoritmaları ile polinom zamanda en uygun çözüm elde edilebilmektedir (Eliiyi ve Azizoğlu, 2006).

Taktik veya operasyonel sabit iş çizelgeleme problemlerine gerçek hayatta tanımlandığı biçimleri ile çok sık karşılaşılmamaktadır. Gerçek hayat uygulamalarında üretim ve hizmet araçlarının zamansal değeri de bulunduğundan birçok firma büyük projelerinde makine kiralama yoluna gitmektedir. Sistemdeki her makinenin belirli bir süre çalıştırılması durumunun makine kiralama olarak düşünülmesi gerçek hayata daha uygun bir problem olup, bu durum yayılma zamanı kısıtları olarak adlandırılmaktadır. Bakım planlaması yapılan durumlarda, makinelerin belirli zaman çalıştıktan sonra belirli bir süre durdurularak bakımlarının yapılması, sonrasında tekrar çalışılması gerektiğinde yayılma zamanı kısıtlarının önem taşıyacağı görülmektedir. Hizmet sektöründe hizmeti sunan personel veya hizmet aracının belirli bir süre için elde bulundurulduğu düşünüldüğünde, problemde yayılma zamanı kısıtları ile karşılaşılmaktadır.

Taktik sabit iş çizelgeleme problemi, örtüşen işlerin sayısının bulunması ile kolay bir biçimde çözülebilirken yayılma zamanı kısıtları ile çalışılması durumunda problemin çok daha zor çözülebildiği bilinmektedir. Problemin NP-Zor problemler sınıfına girdiği bilinmektedir (Eliiyi ve Azizoğlu, 2006).

Literatürde, yayılma zamanı kısıtlı taktik sabit iş çizelgeleme problemi ile otobüs çizelgeleme problemi olarak karşılaşılmaktadır (Solyalı ve Özpeynirci, 2009).

1.2. Rastgele İş Çizelgeleme Problemleri

Rastgele iş çizelgeleme problemleri, gerekirci iş çizelgeleme problemlerinin aksine r_j, d_j, p_{jk}, w_{jk} değerlerinin sayıl değil, belirli olasılık dağılım fonksiyonları ile ifade edilen rastgele(random) değişkenler olduğu problemlerdir. Problemler, amaçlarına ve kısıtlarına göre farklı biçimlerde ifade edilmektedirler. Ayrıca çizelgelemenin yapılacağı anda iş ile ilgili bütün bilgilerin bilinip bilinmemesi durumuna göre çevrimiçi çizelgeleme ve rastgele çizelgeleme olarak sınıflandırılmaktadır.

Rastgele çizelgelemede, tüm işler ile ilgili bilgiler önceden bilinmekte ancak gerekirci modellerin aksine işlerin işlem süreleri rastgele değişkenler ile ifade edilmektedir. İşlerin gerçekleşen tamamlanma süreleri sadece işlerin tamamlanması durumunda belirlenecektir. Karşılık gelen rastgele değişkenlerin dağılım fonksiyonlarının veya en azından birinci momentlerinin önceden bilindiği varsayılmaktadır (Megow ve diğerleri, 2006).

Çevrimiçi çizelgelemede varsayım, çizelgeleme yapacak kişiye işlerin durumları ile ilgili bilgilerin parçalı olarak geldiğidir. İşlerin durumları ile ilgili bilgiler; çevrimiçi liste modelinde işler teker teker, çevrimiçi zaman modelinde ise belirli zamanlarda elde edilmektedir. Bir iş geldiğinde işlem süresi ile ilgili bilgiler ortaya çıkmakta ve kararlar, gelecek işler ile ilgili herhangi bir bilgi olmadan verilmektedir (Megow ve diğerleri, 2006).

Rastgele iş çizelgeleme problemlerine farklı çözüm yaklaşımları geliştirilmiştir. Klasik yöntemlerin temelinde rastgele iş çizelgeleme problemini gerekirci iş çizelgeleme problemine dönüştürmek yer almaktadır. Bu amaçla olasılık fonksiyonları ile ifade edilen parametrelerin sayıl değerler haline getirilmesi gereklidir. Bu işlem rastgele değişken olan p_{jk} için Eşitlik (1.1) ile ifade edilmektedir.

$$\tilde{p}_{jk} = f\left(p_{jk}\right) \tag{1.1}$$

Elde edilen $\tilde{p}_{jk}$ değeri artık rastgele değişken değil bir sayıl değer olmaktadır.

Gürbüzlük faktörü yönteminde rastgele değişkenin ortalama bilgisi kullanılarak sayıl değer elde edilmektedir. Gürbüzlük faktörü R ve ortalaması $\bar{p}_{jk}$ olan rastgele değişken için sayıl değer Eşitlik (1.2) ile ifade edilmektedir.

$$\tilde{p}_{jk} = R\,\bar{p}_{jk} \tag{1.2}$$

Görüldüğü üzere gürbüzlük faktörü yönteminde rastgele değişkene ait sadece ortalama bilgisi kullanılmaktadır. Sadece ortalama değerinin kullanımı gürbüz bir sayıl değer elde etmek için yeterli bilgi sunmamaktadır. Ayrıca gürbüzlük faktörünün iyi ve gürbüz bir çözüm elde edebilmek için seçilmesi başlı başına bir sezgisel süreçtir (Blokland, 2012).

Gürbüzlük yüzdesi yöntemi, rastgele değişkenin dağılım fonksiyonunun belirli bir X yüzdesini sayıl değer elde etmek için kullanmaktadır. Rastgele değişkene ait birikimli dağılım fonksiyonunun tersini alan yöntemde, dağılıma ait bütün parametrelerin bilindiği varsayılmaktadır. Yöntem Eşitlik (1.3) ile ifade edilmektedir. Eşitlikteki $\mathrm{CDF}(x)$ fonksiyonu kullanılan dağılıma ait birikimli olasılık dağılım fonksiyonudur.

$$\tilde{p}_{jk} = \mathrm{CDF}^{-1}(x) \tag{1.3}$$

Rastgele değişkeni temsil eden dağılım fonksiyonuna göre gürbüzlük faktörü yöntemi ile gürbüzlük yüzdesi yöntemi önemli derecede ilişkili olabilmektedir (Blokland, 2012).

Klasik olmayan yöntemlerde ise rastgele değişkenleri tek bir sayıl değere indirgeme işlemi yerine birden fazla sayıl değere indirgeme ile çalışılmaktadır.

İşlem zamanı örneklemesinde rastgele değişken için belirli bir sayıda örnek alınmakta ve alınan örneklerin ortalaması sayıl değer olarak kullanılmaktadır. Sayıl değer bir örnekleme sürecinin sonunda elde edildiği için işlemin tekrar edilmesi ile farklı bir sayıl değer elde edilebilmektedir. Bu yöntem, dağılım ortalamasına yakın ancak küçük miktarlarda sapma içeren sayıl değerler üretmektedir. Elde edilen değerler çözümde kullanılacak arama yöntemlerinde her bir yinelemede tekrar üretilerek sayıl değer olarak kullanılmaktadır (Blokland, 2012).

Bir diğer yöntem ise sonuç örneklemesidir. Sonuç örneklemesi, işlem zamanı örneklemesinin her rastgele değişken için dağılım ortalamasına yakın sonuçlar kullanımının önüne geçmektedir. Her bir rastgele değişken için dağılımdan bir rastgele sayıl değer üretilmekte, üretilen sayıl değer de arama algoritmasında kullanılmaktadır. Arama algoritması çok tekrarlı çalıştırılarak elde edilen sonuçların ortalaması alınmaktadır. Yöntemde rastgele değişkenlerin iyimser ve kötümser değerlerinin bulunma olasılıkları mevcuttur (Blokland, 2012).

2. MATEMATİKSEL EN UYGUNLAMA

Bir en uygunlama problemi matematiksel olarak Sistem (2.1) ile ifade edilmektedir.

$$\text{En Küçükle}_x \left(\text{En Büyükle}_x\right) f_0(x) \tag{2.1a}$$

Aşağıdaki kısıtlar altında:

$$f_i(x) \leq 0 \ \forall i \in \{1, .., m\} \tag{2.1b}$$

$$x \in \text{Tanım Kümesi } X \tag{2.1c}$$

En uygunlama problemi, gerçek hayatta bir fiziksel soruna karşılık gelebileceği gibi fiziksel olmayan bir soruna da karşılık gelebilir. Fiziksel olsun veya olmasın mevcut problemin matematiksel olarak ifade edilmesine matematiksel modelleme adı verilmektedir.

Matematiksel en uygunlamada problemin özel durumları için farklı başarımlara sahip çözüm yöntemleri geliştirilmiştir. Matematiksel yöntemler analitik çözüm yöntemleri olabileceği gibi sayısal(numeric) çözüm yöntemleri de olabilir.

Diferansiyellenebilir bir $f_0(x)$ fonksiyonunu en küçük yapan x değerleri için gerek ve yeter koşullar bilinmektedir. Bu koşullar birinci ve ikinci mertebe koşullar olarak ayrılmaktadırlar. Birinci mertebe gerek koşul diferansiyellenebilir $f_0(x)$ fonksiyonu için Eşitlik (2.2) ile ifade edilmektedir.

$$f_0'\left(x^*\right) = 0 \tag{2.2}$$

Birinci mertebe gerek koşul Fermat teoremi olarak bilinir ve çözümü kritik noktaları verir (Azimli, 2011). Bu kritik noktalar aynı zamanda uç noktaları(extremum point) da kapsamaktadır. Bu kritik noktalar, fonksiyonu en küçük ve en büyük yapmasının yanı sıra eyer noktalarını da içermektedir.

İkinci mertebe gerek koşul ise iki defa diferansiyellenebilen $f_0(x)$ fonksiyonu için Eşitsizlik (2.3) ile ifade edilmektedir.

$$f_0''(x^*) \geq 0 \qquad (2.3)$$

Gerek koşulların sağlanması fonksiyonun x^* noktasında en küçük olması için yeterli değildir. x^* noktası bir en küçük nokta olabileceği gibi eyer noktası da olabilmektedir. Yeterlilik için gerekli koşul ise yüksek mertebeden türevler ile verilmektedir. $f_0(x)$ fonksiyonunun x^* noktasında sıfırdan farklı ilk türevi n. mertebeden ise yeterlilik için koşul Eşitsizlik (2.4) ile ifade edilmektedir.

$$f_0^{(n)}(x^*) > 0 \text{ ve n çift} \qquad (2.4)$$

Bu durumda fonksiyonun x^* noktasında bir yerel en küçüğü olduğu söylenebilir. Tek değişkenli fonksiyonlarda geçerli olan gerek ve yeter koşullar çok değişkenli fonksiyonlar için de yazılabilmektedir. Bu durumda fonksiyonun birinci mertebeden diferansiyeli bir yöney, ikinci mertebeden diferansiyeli ise simetrik bir dizey(matrix) olmaktadır. Fonksiyonun ikinci mertebeden diferansiyeli olan bu dizeye hessian dizeyi adı verilmektedir. Fonksiyonun birinci mertebeden diferansiyeli olan yöneyin x^* noktasında sıfır olması ve hessian dizeyinin x^* noktasında artı(positive) tanımlı olması durumunda x^* noktası en uygunluk koşullarını sağlamaktadır.

$f_0(x)$ fonksiyonunun dışbükeylik(convexity) koşullarını sağlaması durumunda ise fonksiyonun en uygun noktası kritik noktadır ve bu noktada aldığı değerden daha küçük bir değeri başka bir noktada alamamaktadır (Azimli, 2011).

Uygulamadaki birçok gerçek hayat problemi, tek bir fonksiyonun en uygunlanması biçiminde modellenmemekte, başka kısıtlayıcılar ile karşımıza çıkmaktadır. Bu ayrımın sonucunda kısıtsız ve kısıtlı en uygunlama olmak üzere iki ayrı alan ortaya çıkmıştır.

2.1. Kısıtsız En Uygunlama

Kısıtsız en uygunlama, model değişkenlerinin tanım kümesinin $x \in R^n$ olduğu, bu tanım kümesi kısıtı haricinde hiçbir kısıtın bulunmadığı durumda $f_0(x)$ fonksiyonunun en küçük olduğu yöney olan x^* yöneyinin araştırılmasıdır. Problem için sonsuz adımda problemi çözen algoritmalar geliştirilmiştir. Sonsuz sayıda adımda çözümün bulunacak olmasına karşın yeterli hassasiyette olan bir çözüm bulunduğunda algoritmanın durdurulması mühendislik uygulamaları açısından yeterli olmaktadır. Çözüm yöneyinin, en uygun çözüme ne kadar yaklaştığı hesaplanabilmekte ve durdurma kriteri olarak kullanılabilmektedir. Literatürde sıklıkla çözüm yöneyindeki değişimin miktarı ve birinci mertebe gerek koşula yaklaşıklığının ne kadar olduğu algoritma durdurma kriteri olarak kullanılmaktadır.

Kısıtsız en uygunlama problemlerinin çözümü için kullanılan algoritmalar, bir x_0 başlangıç noktası ile başlanarak noktanın belirli bir yönde hareket ettirilmesine dayanmaktadır. Bir sonraki yinelemede ise başlangıç noktası olarak yeni hesaplanmış nokta kullanılmaktadır. Genel halde yöntemler Eşitlik (2.5) ile ifade edilmektedir.

$$x_{k+1} = x_k + \alpha_k \Delta x_k \tag{2.5}$$

k'inci yinelemede bulunulan nokta x_k ile, gidilecek yön Δx_k ile adım büyüklüğü ise α_k ile ifade edilmektedir. Kullanılan algoritmalar bir yön belirlemekte ve bu yönde ne kadar hareket edileceğini adım büyüklüğü ifade etmektedir. Algoritmalar, kullandıkları yön yöneyleri ve adım büyüklüklerine göre isimlendirilmektedirler.

2.1.1. Azalan eğim yöntemi

Azalan(descent) eğim(gradient) yönteminde yön yöneyi olarak birinci mertebeden eğim kullanılmakta, adım büyüklüğü ise en iyi adım büyüklüğü olacak biçimde kesin olarak hesaplanarak veya geri arama algoritması adı verilen algoritmalar ile belirlenmektedir. Genel halde azalan eğim yöntemi Eşitlik (2.6) ile ifade edilmektedir.

$$x_{k+1} = x_k + \alpha_k\left(-\nabla f_0(x_k)\right) \tag{2.6}$$

Yöntem birinci mertebeden diferansiyel bilgisini kullandığından yineleme hızı yüksek olmakla beraber gereken yineleme sayısı oldukça fazla olabilmektedir. Basitliği sebebi ile tercih edilebilecek bir algoritmadır. Diferansiyel yöneyi artış yönünü gösterdiğinden en küçükleme probleminde azalma yönü için yöneyin tersi yönde hareket edilmektedir.

2.1.2. Newton yöntemi

Newton yöntemi pratikte oldukça fazla kullanım alanı bulmuş olan bir algoritmadır. Yöntem, birinci mertebeden diferansiyel bilgisi olan eğim yöneyinin yanı sıra ikinci mertebeden diferansiyel bilgisi olan hessian dizeyini de kullanmaktadır. Genel halde newton yöntemi Eşitlik (2.7) ile ifade edilmektedir.

$$x_{k+1} = x_k + \alpha_k \left(-\nabla^2 f_0(x_k)^{-1} \nabla f_0(x_k) \right) \tag{2.7}$$

Yöntemde kullanılan hessian dizeyinin tersi, hesaplaması karmaşıklık teorisi açısından Gauss-Jordan eliminasyon gibi basit bir yöntem ile $O(n^3)$ olmaktadır. Dizeyin ayrık veya üçköşegen(tridiagonal) olması durumunda ise dizeyin tersinin hesaplanması işlemi için karmaşıklık teorisi açısından daha hızlı algoritmalar geliştirilmiştir.

2.2. Kısıtlı En Uygunlama

Kısıtlı en uygunlama, gerçek hayatta karşılaşılan problemlerde sistemin gerektirdiği kısıtların modellenmesinde ve çözülmesinde sıklıkla kullanılmaktadır. Kısıtlar birçok durumda farklı matematiksel eşitlik veya eşitsizlikler ile ifade edilebilmektedirler. Bu sebeple farklı matematiksel modellerin çözülmesi konusunda farklı algoritmalar geliştirilmiştir. Modellerin türleri etkin çözüm elde etmek için önem taşımakla beraber kısıtlı en uygunlama için genel çözüm yöntemleri, model türünden bağımsız olarak kullanılabilmektedir. Kısıtlı en uygunlama problemi, bir dizi kısıtsız en uygunlama problemi biçiminde ifade edilerek kısıtsız en uygunlama algoritmaları ile çözülebilmektedir. Kısıtların, kısıtsız en uygunlama problemi olarak ifade edilmesinde genellikle kısıtların cezalandırılmasına yönelik bir yaklaşım kullanılmaktadır. Ceza fonksiyonu ile amaç fonksiyonu Eşitlik (2.8) ile ifade edilmektedir.

$$\text{En Küçükle}_x \left(\text{En Büyükle}_x\right) f_0(x) + \sum_{i=1}^{m} \varphi\left(f_i(x)\right) \tag{2.8}$$

Ceza fonksiyonu $\varphi(x)$ ile ifade edilmektedir. Uygulamada ceza fonksiyonu olarak kullanılan fonksiyonların temel amacı kısıtsız en uygunlama problemlerinde kullanılan algoritmaların daha iyi bir en uygun değeri bulamamasını sağlamaktır. Teorik olarak en uygun ceza fonksiyonu, kısıtların sağlanmadığı durumda sonsuz, kısıtların sağlandığı durumlarda ise sıfır değeri alan bir fonksiyondur. Ancak uygulamada, eşitlik kısıtları için fonksiyon değerinin karesi, eşitsizlik kısıtları için negatif logaritma fonksiyonunun ceza fonksiyonu olarak kullanımı yaygındır. Bu durumun sebebi ise ceza fonksiyonunun diferansiyelinin kolay hesaplanabilmesidir.

2.2.1. Dışbükey programlama

Dışbükey(convex) programlama modellerinin çözümü teorik ve pratik olarak diğer matematiksel modellere göre kolay olduğundan üzerinde oldukça çalışılmıştır. Ayrıca dışbükey programlamada bulunan bir yerel en uygun çözüm, mutlak en uygun çözüm olduğundan dolayı uygulamada ayrı bir yere sahiptir. Dışbükey programlama modeli Sistem (2.9) ile ifade edilmektedir.

$$\text{En Küçükle}_x \left(\text{En Büyükle}_x\right) f_0(x) \tag{2.9a}$$

Aşağıdaki kısıtlar altında:

$$f_i(x) \leq 0 \ \forall i \in \{1, .., m\} \tag{2.9b}$$

$$Ax = b \tag{2.9c}$$

$$x \in R^n \tag{2.9d}$$

Sistem (2.9)'da bulunan tüm $f_i(x)$ fonksiyonlarının dışbükey olması, eşitlik kısıtlarının ise sadece afin(affine) fonksiyonlar olması gerekmektedir (Boyd ve Vandenberghe, 2009).

Dışbükey programlama birçok durumda tümler gevşeklik, zayıf ve güçlü ikillik(duality) ile ilgili kuramları(theorem) sağlamaktadır. Bu kuramların kullanılması ile birincil-ikil algoritmalar geliştirilmiştir. Ayrıca birçok problemde - zayıf ikillik kuramı ile ilişkili olarak - alt/üst sınırın hesaplanması için bir araç olarak kullanılmaktadır.

2.2.1.1.Doğrusal programlama

Doğrusal programlama dışbükey en uygunlama modellerinin en çok bilineni ve en sık kullanılanıdır. Teorik ve pratik olarak sıkça çalışılmış olan doğrusal programlama, sonlu adımda çözümün bulunabildiği bir matematiksel programlama alanıdır. Doğrusal programlama modeli Sistem (2.10) ile ifade edilmektedir.

$$\text{En Küçükle}_x \left(\text{En Büyükle}_x\right) c^T x \tag{2.10a}$$

Aşağıdaki kısıtlar altında:

$$Ax = b \tag{2.10b}$$

$$x \geqslant 0 \tag{2.10c}$$

$$x \in R^n \tag{2.10d}$$

Sistem (2.10)'da amaç fonksiyonu iki yöneyin iç çarpımları biçiminde ifade edilmekte olup, eşitlik kısıtlayıcıların tamamı dizey eşitliği biçiminde yazılmıştır. Karar değişkenlerinin negatif olmaması kısıtı modeldeki tek eşitsizlik kısıtıdır.

Doğrusal programlamada kısıtların sınırladığı alan bir dışbükey çokyüzlü olup, en uygun çözüm yöneyi kısıtların belirlediği alanın köşe noktalarından birisinde bulunmak durumundadır. Bu durumda tüm köşe noktaları test edilerek çözümün bulunabileceği açık olmakla beraber köşe noktaları üzerinde ilerleyen algoritmaların pratikte oldukça başarılı oldukları bilinmektedir. Ancak bu algoritmaların bazı durumlar için teorik olarak polinom zamanlı olmayan karmaşıklığa sahip olduğu gösterilmiştir. Bu durumun yanı sıra çokyüzlünün köşeleri yerine iç bölgesinden hareket edilerek çözüm yapan ve karmaşıklığı polinom zamanlı olan bazı algoritmalar bulunmuştur.

2.2.1.2.İkinci derece programlama

İkinci derece programlama, en uygun çözümde olması gereken koşulların özel bir duruma getirilebilmesi ile çözülmektedir. İkinci derece programlama modeli P dizeyinin bakışımlı(symmetric) artı yarı tanımlı olduğu durumda Sistem (2.11) ile ifade edilmektedir.

$$\text{En Küçükle}_x\ \left(\text{En Büyükle}_x\right)\ (1/2)x^TPx+q^Tx + r \tag{2.11a}$$

Aşağıdaki kısıtlar altında:

$$Gx \preccurlyeq h \tag{2.11b}$$

$$Ax= b \tag{2.11c}$$

$$x \in R^n \tag{2.11d}$$

Sistem (2.11)'de amaç fonksiyonu ikinci dereceden bir polinomun dizey ifadesidir. Kısıt kümesinin çokyüzlü olduğu kolaylıkla görülmekte, ancak doğrusal programlamadaki çözüm yöneyinin çokyüzlünün köşesinde olma durumu oluşmamaktadır. İkinci derece programlama modeli, doğrusal programlama modelini de özel bir durum olarak kapsamaktadır. Şöyle ki ikinci derece programlama modelinde P dizeyinin 0 dizeyi olduğu durumda model, doğrusal programlama modeline dönüşmektedir. Ayrıca gerektiği durumda x yöneyinin 0 yöneyinden büyük olma zorunluluğu eşitsizlik kısıtı ile sağlanmalıdır.

İkinci derece programlama modellerini de kapsayan ve kısıt kümesi elipsoidlerin kesişimi olan ikinci derece kısıtlı ikinci derece programlama modelleri P_i dizeylerinin bakışımlı artı yarı tanımlı olduğu durumda Sistem (2.12) ile ifade edilmektedir.

$$\text{En Küçükle}_x\ \left(\text{En Büyükle}_x\right)\ (1/2)x^TP_0x+q_0^Tx + r_0 \tag{2.12a}$$

Aşağıdaki kısıtlar altında:

$$\left(\frac{1}{2}\right)x^TP_ix+q_i^Tx + r_i \leq 0\ \forall i \in \{1, .., m\} \tag{2.12b}$$

$$Ax= b \tag{2.12c}$$

$$x \in R^n \tag{2.12d}$$

İkinci derece programlama modelleri birçok alanda öncülük etmektedir. En çok bilinen uygulamaları doğrusal en küçük kareler yöntemi, Markowitz portföy en uygunlaması ve rastgele programlamadır.

2.2.1.3.Doğrusal kesirli programlama

Doğrusal kesirli programlama modeli, bir dışbükey çokyüzlü kümesinde amaç fonksiyonu olarak iki afin fonksiyonun oranlandığı matematiksel programlama modelidir. Bu modelin tanım kümesinde paydanın sıfırdan büyük olduğu kabulü yapılmaktadır. Doğrusal kesirli programlama modelleri Sistem (2.13) ile ifade edilmektedir.

$$\text{En Küçükle}_x \left(\text{En Büyükle}_x\right) \frac{c^T x + d}{e^T x + f} \tag{2.13a}$$

Aşağıdaki kısıtlar altında:

$$Gx \preccurlyeq h \tag{2.13b}$$

$$Ax = b \tag{2.13c}$$

$$x \in R^n \tag{2.13d}$$

Modelin çözümü için bir dönüşüm kullanılmaktadır. Dönüşüm, Eşitlik (2.14) ve Eşitlik (2.15) ile ifade edilmektedir.

$$y = \frac{x}{e^T x + f} \tag{2.14}$$

$$z = \frac{1}{e^T x + f} \tag{2.15}$$

Dönüşüm yapıldıktan sonra eşdeğer model Sistem (2.16) ile ifade edilmektedir.

$$\text{En Küçükle}_{y,\,z} \left(\text{En Büyükle}_{y,\,z}\right) c^T y + dz \tag{2.16a}$$

Aşağıdaki kısıtlar altında:

$$Gy - hz \preccurlyeq 0 \tag{2.16b}$$

$$Ay - bz = 0 \tag{2.16c}$$

$$e^T y + fz = 1 \tag{2.16d}$$

$$z \geqslant 0 \tag{2.16e}$$

$$y, z \in R^n \tag{2.16f}$$

Dönüşüm sonunda oluşan denklem sisteminin doğrusal programlama modeli olduğu görülmektedir ve doğrusal programlama için geliştirilen algoritmalar yardımı ile çözülebilmektedir.

2.2.1.4.Geometrik programlama

Geometrik programlama modeli, doğrusal kesirli programlama modeli gibi doğrudan dışbükey programlama modeli olmayıp dönüşüm yardımı ile kolayca çözülmekte olan bir matematiksel programlama modelidir. Bir geometrik programlama modelinin açıklanması için öncelikle birterimli(monomial) fonksiyonun tanımlanması gerekmektedir. Birterimli fonksiyon Eşitlik (2.17) ile ifade edilmektedir.

$$f(x) = cx_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n} \tag{2.17}$$

Burada c terimi sıfırdan büyük bir katsayıdır. Birterimli fonksiyonların toplamı bir çokterimli(posynomial) fonksiyonu oluşturmaktadır. Bir çokterimli fonksiyon Eşitlik (2.18) ile ifade edilmektedir.

$$f(x) = \sum_{k=1}^{K} c_k x_1^{\alpha_{1k}} x_2^{\alpha_{2k}} \ldots x_n^{\alpha_{nk}} \tag{2.18}$$

Bu tanımların yapılmasından sonra bir geometrik programlama modeli Sistem (2.19) ile ifade edilmektedir.

$$\text{En Küçükle}_x \left(\text{En Büyükle}_x\right) f_0(x) \tag{2.19a}$$

Aşağıdaki kısıtlar altında:

$$f_i(x) \leq 1 \ \forall i \in \{1, .., m\} \tag{2.19b}$$

$$h_i(x) = 1 \ \forall i \in \{1, .., p\} \tag{2.19c}$$

Modelde $f_i(x)$ fonksiyonları çokterimli fonksiyonlar, $h_i(x)$ fonksiyonları birterimli fonksiyonlardır. Modelin çözümü için yapılan dönüşüme ait denklem Eşitlik (2.20) ile ifade edilmektedir.

$$y_i = \log(x_i) \tag{2.20}$$

Dönüşümün yapılması ile model dışbükey programlama modeli haline gelmektedir. Kısıtların ve amaç fonksiyonunun birterimli fonksiyonlar olması durumunda ise model dışbükey programlama modellerinden doğrusal programlama modeline dönüşmektedir.

2.2.1.5.Yarı tanımlı programlama

Yarı tanımlı programlama, doğrusal programlamanın genelleştirilmiş halidir. Doğrusal programlamadaki karar yöneylerinin genelleştirilerek dizeylere uygulanması ile kurulmaktadır. Genelleştirilmiş eşitsizlikler ile tanımlanabileceği gibi doğrusal programlama modeline benzer biçimde dizeyler ile de tanımlanabilmektedir. Yarı belirli programlama modeli Sistem (2.21) ile ifade edilmektedir.

$$\text{En Küçükle}_x \left(\text{En Büyükle}_x\right) \text{İz}(CX) \tag{2.21a}$$

Aşağıdaki kısıtlar altında:

$$\text{İz}(A_i X) = b_i \ \forall i \in \{1, 2, \ldots, p\} \tag{2.21b}$$

$$X \succcurlyeq 0 \tag{2.21c}$$

X dizeyi yarı artı tanımlı bir dizeydir ve tüm özdeğerleri sıfır veya artı değerlidir. Özel bir durum olan dizeyin köşegeni hariç tüm değerlerinin sıfır olması durumunda model doğrusal programlama modelini kapsamaktadır.

Yarı tanımlı programlama için günümüzde oldukça hızlı çözücüler geliştirilmiştir. Çözücülerin kullandıkları algoritmalar farklı olmakla beraber problemin çözümü polinom zamanlıdır.

Yarı tanımlı programlama polinom zamanda çözülebildiğinden dolayı tamsayılı programlama problemleri için tamsayı gevşetmesine alternatif kullanışlı bir yaklaşım sunabilmektedir. Bazı problem türleri için yarı tanımlı programlama gevşetmesinin üst sınırdan en kötü durumda ne kadar uzak olduğu matematiksel olarak hesaplanabilmektedir.

Bir 0-1 tamsayılı doğrusal programlama problemi Sistem (2.22) ile verilmiş olsun.

$$\text{En Küçükle}_x \; c^T x \tag{2.22a}$$

Aşağıdaki Kısıtlar Altında:

$$Ax \leq b \qquad \forall i \in \{1, 2, \ldots, p\} \tag{2.22b}$$

$$x \in \{0, 1\} \tag{2.22c}$$

Problemde Formül (2.22c) kısıtının olmaması durumunda problem doğrusal programlama problemi biçiminde ifade edilebilecek ve polinom zamanda çözülebilecektir. Bu problem için doğrusal programlama gevşetmesi Formül (2.22c) kısıtının kaldırılarak Sistem (2.23)'e Eşitsizlik (2.23c) ve Eşitsizlik (2.23d) kısıtlarının eklenmesi ile elde edilir.

$$\text{En Küçükle}_x \; c^T x \tag{2.23a}$$

Aşağıdaki Koşullar Altında:

$$Ax \leq b \qquad \forall i \in \{1, 2, \ldots, p\} \tag{2.23b}$$

$$x \geq 0 \tag{2.23c}$$

$$x \leq 1 \tag{2.23d}$$

Eşitsizlik (2.23c) ve Eşitsizlik (2.23d) ile değişken yöneyinin sürekli değerler alabilmesi ve bu değerlerin $[0, 1]$ kapalı aralığında olması gerektiği ifade edilmektedir. Modelin çözümü polinom zamanda yapılabilmekte ve problem için bir üst sınır tanımlamaktadır. Hesaplanan bu üst sınır, kesme düzlemi algoritmalarında kesme düzlemlerinin hesaplanması için de kullanılmaktadır.

Yarı tanımlı programlama problemlerinde X değişken dizeyinin artı tanımlı olması gerekmektedir. Bir x değişken yöneyi için X dizeyi Eşitlik (2.24) ile tanımlanabilir.

$$X = \begin{bmatrix} 1 & x^T \\ x & xx^T \end{bmatrix} \tag{2.24}$$

Eşitlik (2.24) ile gösterilen dizeyde bulunan xx^T dizeyinin artı tanımlı olması zorunluluktur. xx^T dizeyi rank 1 bir dizey olmakta ve problemi çözülmesi zor bir probleme dönüştürmektedir. Bu xx^T dizeyinin Y dizeyi olarak ifade edilmesi durumunda Eşitlik (2.25) elde edilmektedir.

$$Y = xx^T \tag{2.25}$$

Y dizeyinin rank 1 olma koşulu kaldırılır ise problem daha basit bir çözüme sahip olmaktadır. Bunun için Eşitsizlik (2.26) kabul edilirse yarı tanımlı programlama gevşetmesi elde edilmektedir.

$$Y - xx^T \succcurlyeq 0 \tag{2.26}$$

Daha katı koşullar elde etmek için, x yöneyinin $\{0, 1\}$ kümesinden değer alması gerektiğinden dolayı Y dizeyinin köşegenindeki elemanlarının x yöneyindeki elemanlara eşit olması gerektiği düşünülebilir. Bu koşul yarı tanımlı programlama modelinde Eşitlik (2.27) ile verilen kısıtlar biçiminde ifade edilmektedir.

$$Y_{ii} = x_i \tag{2.27}$$

Yarı tanımlı programlama gevşetme modeli, ek kısıtlar olmadan gevşetilmiş doğrusal programlama modeli ile aynı üst sınırı bulmaktadır. Bu durum Eşitsizlik (2.26)'nın, Eşitsizlik (2.23c) ve Eşitsizlik (2.23d) ile verilen ve değişkenlerin $[0, 1]$ kapalı aralığında sürekli değerler alabilmelerini ifade etmesinden kaynaklanmaktadır. Daha katı kısıtlar elde etmek için X dizeyinin özellikleri incelendiğinde, $x_i x_j$ değişken ifadesinin modellenebildiği görülmektedir. Herhangi bir $a^T x \leq b$ biçimindeki doğrusal kısıtta, her iki taraf x_i ve $(1 - x_i)$ ile çarpılırsa Eşitsizlik (2.28) ve Eşitsizlik (2.29) eşitsizlikleri elde edilmektedir (Laurent ve Rendl, 2002).

$$\sum_{j=1}^{n} a_j X_{ij} \leq b X_{ii} \qquad \forall i \in \{1, 2, \ldots, n\} \tag{2.28}$$

$$\sum_{j=1}^{n} a_j\left(X_{jj} - X_{ij}\right) \leq b(1 - X_{ii}) \qquad \forall i \in \{1, 2, \ldots, n\} \qquad (2.29)$$

Eşitsizlik (2.28) ve Eşitsizlik (2.29), yarı tanımlı programlama problemine kısıt olarak eklendiğinde doğrusal programlama gevşetmesinden daha iyi veya aynı değerde bir sınır elde edilmektedir.

3.3.1.1.Ayrılabilir programlama

Ayrılabilir programlama, amaç ve kısıt fonksiyonlarının tek değişkene bağlı fonksiyonların toplamı olarak yazılabildiği problemlerdir. Ayrılabilir programlamada fonksiyonların doğrusal olması zorunlu değildir. Genel olarak ayrılabilir programlama modeli Sistem (2.30) ile ifade edilmektedir.

$$\text{En Küçükle}_x \left(\text{En Büyükle}_x\right) \sum_{j=1}^{n} f_{0j}\left(x_j\right) \qquad (2.30a)$$

Aşağıdaki kısıtlar altında:

$$\sum_{j=1}^{n} f_{ij}\left(x_j\right) \leq 0 \ \forall i \in \{1, .., m\} \qquad (2.30b)$$

Ayrılabilir programlamada doğrusal olmayan fonksiyonlar, küçük aralıklarda doğrusal kabul edilerek değişkenlerin doğrusallaştırılması ile çok sayıda doğrusal programlama modeline dönüştürülmektedir. Elde edilen doğrusal modellerin tamamının çözülmesi ile ayrılabilir programlama probleminin çözümü elde edilir.

2.2.2. Dışbükey olmayan programlama

Dışbükey olmayan programlama, amaç fonksiyonunun veya kısıtların oluşturduğu olurlu bölgenin dışbükey olmadığı modelleri çözmek için kullanılan yöntemleri içermektedir. Birçok durumda güçlü ikillik kuramı bu tür modeller için geçerli olmamaktadır. Güçlü ikillik kuramının geçerli olduğu durumlarda en uygun çözümün bulunması mümkün olmaktadır. Kapsadığı model türlerinden en önemli olanı tamsayılı programlama modelleridir. Doğrusal tamsayılı programlama modelleri, doğrusal karma tamsayılı programlama modelleri, 0-1 tamsayılı programlama modelleri, doğrusal olmayan tamsayılı programlama modelleri bu kapsamda değerlendirilmektedir.

Doğrusal olmayan programlama modellerini çözmek için kullanılan farklı yaklaşımlar bulunmaktadır. Bu yaklaşımların birçoğu newton yönteminin değiştirilmesi ile geliştirilmiştir, bazıları ise ikil bilgisi kullanmaktadır. Bunlardan bazıları kuasi(quasi) newton yöntemleri, eşlenik eğim yöntemleri, Levenberg-Marquardt yöntemi, güvenli bölge yöntemleridir (Bazaraa, 2006).

3. SEZGİSEL VE METASEZGİSEL YÖNTEMLER

Sezgisel ve metasezgisel yöntemler, gerçek hayatta karşılaşılan ve analitik ve/veya matematiksel yinelemeler yardımı ile çözümü oldukça uzun süreler alabilen problemler için geliştirilmişlerdir. Sezgisel ve metasezgisel yöntemler en uygun çözümü sunma garantisi vermemekle birlikte uygulamada oldukça iyi sonuçlar verebildiklerinden dolayı tercih edilmektedirler. Sezgisel yöntemlerin genel özelliği, geliştirilmiş olduğu problem türü için problemin girdi verisine bağlı olarak her zaman aynı çıktıyı üretmeleridir. Metasezgisel yöntemler ise rastgele sayılar yardımı ile mevcut çözümü veya çözümleri geliştirmektedirler. Rastgele sayıların aynı sayılar geldikleri geliştirme durumları hariç her çalıştırmalarında farklı adımlar işletmeleri ve değişik çözümler bulmaları beklenmektedir.

Literatüre bakıldığında metasezgisel yöntemlerde genel olarak iki temel sınıflandırmaya gidildiği görülmektedir. Birinci sınıflandırma geliştirilen çözüm sayısı ile ilgili ayrımdır. Tek bir aday çözüm geliştirilebildiği gibi birden çok bireyi geliştiren popülasyon(population) tabanlı algoritmalar da bulunmaktadır. Tek bir aday çözümün geliştirildiği algoritmalardan en bilinenleri tabu arama ve tavlama benzetimi(simulated annealing) algoritmalarıdır. Popülasyon tabanlı algoritmalardan en bilinenleri ise genetik algoritmalar, karınca algoritması, parçacık sürü optimizasyonu algoritmasıdır. İkinci sınıflamaya göre ise metasezgisel yöntemler, yapıcı algoritmalar ve çözüm geliştirici algoritmalar olarak ayrılmaktadırlar. Yapıcı algoritmalar temel veriler ile yeni bir çözüm üretmeye odaklanmaktadırlar. Temel veriler genellikle probleme dair veriler ve daha önce üretilmiş olan çözümlerden çıkarılan verilerdir. Çözüm geliştirici algoritmalar ise mevcut bir veya birkaç çözümü yeni bir çözüm elde etmek için kullanırlar. Yapıcı algoritmalardan en bilineni karınca algoritması, çözüm geliştirici algoritmalardan en bilinenleri tabu arama algoritması ve genetik algoritmalardır.

Sıklıkla kullanılan metasezgisel algoritmalar şunlardır:

- Tabu arama algoritması(Tabu search algorithm)

- Tavlama benzetimi algoritması(Simulated annealing algorithm)
- Genetik algoritmalar(Genetic algorithms)
- Yapay bağışıklık sistemi algoritması(Artificial immune system algorithm)
- Karınca algoritması(Ant algorithm)
- Parçacık sürü algoritması(Particle swarm algorithm)
- Yapay arı koloni algoritması(Artificial bee colony algorithm)
- Diferansiyel gelişim algoritması(Differential evolution algorithm)

Bu algoritmalar ayrı alt bölümlerde açıklanmaktadır. Bu algoritmaların yanı sıra yapay kanguru algoritması, büyük patlama büyük çatırtı algoritması, uyum arama algoritması gibi algoritmalar da literatürde bulunmaktadır. Ayrıca son dönemlerde ortaya çıkan memetik algoritmalar ve hiper sezgisel yöntemler de literatürde mevcuttur.

3.1. Tabu Arama Algoritması

Tabu arama algoritması F. Glover tarafından geliştirilmiş yinelemeli bir araştırma algoritmasıdır (Karaboğa, 2014). Algoritma tek bir çözüm üzerinden ilerlemektedir. Mevcut çözüme komşu çözümlerin üretilerek değerlendirilmesi ve değerlendirme sonucunda bir komşu çözüme geçilmesi prensibine dayanmaktadır. Bazı durumlarda seçilen komşu çözüm bir sonraki yinelemede çözümü bir önceki duruma geri getirebilmektedir. Bu gibi durumlar algoritma işleyişinde çözümler arasında döngü oluşmasına sebep olmakta, kullanılan işlem gücünün boşa harcanması ve mevcut çözümün gelişememesi ile sonuçlanmaktadır. Bu sorunu ortadan kaldırmak amacı ile tabu arama algoritması tabu listesi adı verilen bir yasaklamalar listesi tutmakta, bu yasaklamalar listesinde bulunan komşuluk hareketlerini mevcut çözüme uygulamamaktadır. Tabu listesinde bulunan bir hareketin, mevcut çözümü oldukça geliştirebilecek olduğu durumlarda tabu listesindeki hareketin yapılamaması bir problem oluşturmaktadır. Bu problemi ortadan kaldırmak için nefes alma kriteri adı verilen bir kriter geliştirilmesi düşünülmüştür. Eğer mevcut çözüme uygulanacak ancak tabu listesinde bulunan bir komşuluk hareketi, çözümü nefes alma kriterini sağlayacak biçimde geliştirebilecek ise bu komşuluk hareketine izin verilmektedir.

Bir diğer durum da tabu listesinin sınırsız olamayacağı ile ilgili durumdur. Gerek hesaplama sistemlerinin depolama miktarlarındaki sınırlamalar gerekse üretilebilecek ve değerlendirilebilecek komşu çözüm sayısının sınırlı olması dolayısı ile tabu listeleri belirli uzunlukta tutulmaktadır.

Farklı problemlerde farklı komşu çözüm üretme yöntemleri bulunmaktadır. Bu komşu çözüm üretme yöntemlerinden biri veya birkaçı bir arada kullanılabilmektedir. Tabu arama algoritması tek bir başlangıç çözümü ile çalıştığından, tüm çözüm uzayını dolaşabilmeyi sağlayacak komşu çözüm üretme yönteminin seçilmesi önem taşımaktadır.

Tabu arama algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: Bir başlangıç çözümü x_0 al ve gerekli parametreleri ayarla.
Adım 2: Komşuluk fonksiyonu $\varepsilon(x)$ ile komşu çözümler üret.
Adım 3: Üretilen komşu çözümlerden tabu listesinde olmayan ve aspirasyon (aspiration) kriterleri tarafından yapılması mümkün kılınmayan en uygun komşu çözümü seç ve yeni çözüm olarak ata.
Adım 4: Durdurma kriteri sağlanmadı ise Adım 2'ye git.

Ayrıca farklı çözümler ile tekrar başlama, uzun dönemli hafıza stratejileri gibi yöntemler de literatürde probleme özgü olarak kullanılmaktadır.

3.2. Tavlama Benzetimi Algoritması

Tavlama benzetimi algoritması aynı zamanda ısıl işlem algoritması adı ile de bilinmektedir. Algoritma, metal malzemelerde tavlama işleminin yapısından esinlenerek geliştirilmiştir. Tavlama işlemi, metallerin daha sağlam olması için belirli bir sıcaklığa kadar ısıtılması, daha sonrasında ise aniden soğutulması ve bu işlemin sıcaklık düşürülerek tekrar tekrar yapılmasıdır. Yöntemin matematiksel en küçükleme problemi ile bağlantısı Pincus tarafından vurgulanmış, en uygunlama tekniği olarak kullanılması fikri ise Kirkpatrick ve diğerleri tarafından ortaya atılmıştır (Cura, 2008).

Tavlama benzetimi algoritması da tek bir çözüm üzerinden ilerlemektedir. Bu çözüme komşu yeni çözümler üretilerek değerlendirilmekte ve o anki sıcaklık değerine göre daha kötü bir çözümün kabul edilip edilmeyeceğine karar verilmektedir. Sıcaklık değeri belirli kurallara göre değiştirilmekte, aday komşu çözümün uygunluk değerinin o anki sıcaklıkta esas çözümün yerine geçip geçmeyeceği, uygunluk değerlerine bağlı olarak rastgele bir süreç ile belirlenmektedir. Sıcaklık değerinin değiştirilmesi monoton azalma ile yapılabildiği gibi çözümlerdeki iyileşme seviyesi ve benzeri yöntemler ile de yapılmaktadır.

Tavlama benzetimi algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: Bir başlangıç çözümü x_0 al ve gerekli parametreleri ayarla.
Adım 2: Komşuluk fonksiyonu $\varepsilon(x)$ ile komşu çözüm üret.
Adım 3: Üretilen komşu çözümün uygunluk değeri ile mevcut çözümün uygunluk değeri arasındaki farkı (Δ) hesapla.
Adım 4: Eğer üretilen komşu çözümün uygunluk değeri mevcut çözümün uygunluk değerinden küçük ($\Delta <0$) ise mevcut çözümü güncelle.
Adım 5: Eğer üretilen komşu çözümün uygunluk değeri mevcut çözümün uygunluk değerinden büyük veya eşit ($\Delta \geq 0$) ise; $(0, 1)$ aralığında bir θ rastgele sayısı üret ve anlık sıcaklık T için $e^{\frac{-\Delta}{T}} > \theta$ koşulu sağlanıyor ise mevcut çözümü güncelle.
Adım 6: Sıcaklığı kullanılan kurallara göre güncelle.
Adım 7: Durdurma kriteri sağlanmadı ise Adım 2'ye git.

Farklı çözümler ile tekrar başlama, farklı soğutma stratejileri gibi yöntemler de literatürde probleme özgü olarak kullanılmaktadır.

3.3. Genetik Algoritmalar

Genetik algoritma, metasezgisel yöntemler alanında en çok bilinen ve üzerinde çalışılmış olan yöntemdir. Genetik algoritma genellikle genetik algoritmalar olarak ifade edilmektedir ve John Holland tarafından 1970'lerde keşfedilmiştir (Luke, 2009). Genetik algoritmalar evrime dayalı algoritmaların bir türüdür (Karaboğa, 2014). Çok farklı uygulama alanları bulmuş olan genetik algoritmalar temel olarak popülasyondaki bireylerin birbiri ile etkileşimini de kullanan algoritmalardır.

Bireyin temsil edilmesi probleme özgü niteliklerin gen ile ifade edilmesi ile başlar. Bir gen kodlama yapısına göre tek bir değişkene karşılık gelebileceği gibi birkaç gen birlikte tek bir değişkene de karşılık gelebilir. Bazı kodlama yapılarında ise genler değişkenlere karşılık gelmemekte, genlerin sıralaması değişkenlere karşılık gelmektedir. Bu, tamamen problem türü ve problemin nasıl modelleneceği ile ilgili bir durumdur. Genler birleşerek kromozom dizisi oluşturmaktadırlar. Genel olarak kullanılan kromozom kodlama yapıları ikil, reel ve devşirim kodlama yapılarıdır.

Genetik algoritmalar temel olarak seçim(selection), çaprazlama(crossover), değişinim(mutation) operatörleri ile çalışmaktadırlar. Bu operatörlerin en çok kullanılanları alt bölümlerde açıklanmaktadır.

3.3.1. Seçim operatörü

Seçim operatörü, bir popülasyonda diğer operatörlerin uygulanacağı bireylerin nasıl seçileceğini belirleyen operatördür. Bu operatör, yaşamak için daha uygun olan bireylerin daha çok seçilmesini, yaşamaya uygun olmayan bireylerin ise süreç içerisinde mevcut popülasyonda bulunmamalarını amaçlamaktadır. Farklı seçim yöntemlerinden bazıları rulet çemberi, turnuva seçimi, kesme seçimi, rastgele seçim yöntemleridir.

Rulet çemberi tekniği, en uygun bireye en yüksek seçilme olasılığını, en uygun ikinci bireye en yüksek ikinci seçilme olasılığını, en uygun son bireye de en düşük seçilme olasılığını vermeyi amaçlamaktadır. En büyükleme problemlerinde birey uygunluklarının - genellemeyi bozmadan sıfırdan büyük oldukları varsayımı ile - seçilme ihtimallerini belirtmesi gerektiği düşünülerek her bir birey uygunluğunun uygunluklar toplamına bölünmesi ile seçilme olasılıkları hesaplanmaktadır. En büyükleme problemlerinde her bir birey için seçilme olasılığı Eşitlik (3.1) ile verilmektedir.

$$p_i = \frac{C(x_i)}{\sum_{j=1}^{N} C(x_j)} \qquad \forall i \tag{3.1}$$

Eşitlikte i kaçıncı bireyin olasılığının hesaplandığını, $C(x)$ fonksiyonu probleme özgü olan uygunluk fonksiyonunu, N ise popülasyondaki toplam birey sayısını ifade etmektedir. Yöntem uygunluk değeri düşük olan bireylere de bir sonraki popülasyonda var olma fırsatı vermektedir (Chudasama ve diğerleri, 2011).

En küçükleme problemleri için ise uygulamada farklı yaklaşımlar kullanılabilmektedir. Olasılık hesabı için ölçeklemeyi bozmadan kullanılan bir yöntem diğer uygunluk değerlerinin çarpımının uygunluk değeri olarak en büyükleme olasılık hesabı eşitliğindeki gibi kullanılmasıdır. Yöntemin dezavantajı, hesaplama sistemlerindeki sayıların çarpımlarının büyük sayılar çıkması sonucunda belirli bir büyüklüğün üzerindeki sayıların taşma durumlarına yol açabilmesidir. En küçükleme problemlerinde her bir birey için seçilme olasılığı Eşitlik (3.2) ile verilmektedir.

$$p_i = \frac{\prod_{j=1,\, j\neq i}^{N} C(x_j)}{\sum_{k=1}^{N} \prod_{j=1,\, j\neq k}^{N} C(x_j)} \qquad \forall i \tag{3.2}$$

Diğer bir yöntem ise en küçükleme probleminde uygunluk değerlerinin çarpmaya göre tersini alarak çalışmaktadır. Çarpmaya göre tersi alınan uygunluk değerleri en büyükleme olasılık hesabındaki gibi kullanılabilmektedir. Bu yöntemin dezavantajı, matematiksel olarak Eşitlik (3.2) ile aynı olmalarına rağmen çarpmaya göre ters işleminde hesaplama sistemlerinin hassasiyetine bağlı olarak kayıpların olabilmesidir. Yöntem ile her bir birey için seçilme olasılığı Eşitlik (3.3) ile verilmektedir.

$$p_i = \frac{\left(\frac{1}{C(x_i)}\right)}{\sum_{j=1}^{N} \left(\frac{1}{C(x_j)}\right)} \qquad \forall i \tag{3.3}$$

Kullanılabilecek bir başka yöntem ise ölçeklemeyi bozan bir yöntemdir. Yöntem, en büyük uygunluk değerine uzaklığa göre çalışmakta ancak en büyük maliyete sahip bireyin olasılığının sıfır olmaması için bir eşikleme yapmaktadır. Yöntem ile her bir birey için seçilme olasılığı Eşitlik (3.4) ile verilmektedir. Eşitlikte δ eşik miktarını göstermektedir.

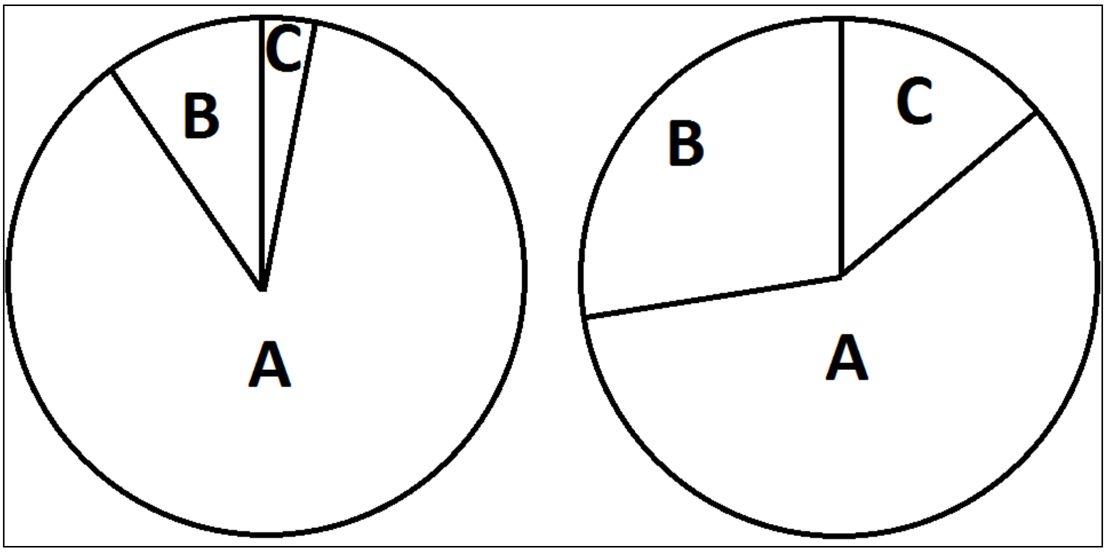
$$p_i = \frac{\left(\max_k C(x_k)\right) - C(x_i) + \delta}{\sum_{j=1}^{N} \left(\left(\max_k C(x_k)\right) - C(x_j) + \delta\right)} \qquad \forall i \tag{3.4}$$

Rulet çemberi tekniği uygunluk fonksiyonu değeri ile kullanılabildiği gibi sıralama ile de kullanılabilmektedir. Sıralama tabanlı rulet çemberi tekniği ilk olarak uygunluk değerlerine göre bireyleri sıralamakta ve bu bireylerin sıralamada kaçıncı olduğunu belirlemektedir. Sıralama tabanlı rulet çemberi tekniği, bireylerin sıralamadaki sıralarını bireylerin seçim olasılık değerlerine çevirmek için bir fonksiyon kullanmaktadır. Kullanılan fonksiyon doğrusal veya doğrusal olmayan formda olabilmektedir. Yöntemin performansı seçilen fonksiyona oldukça bağımlıdır. Doğrusal sıralama tabanlı rulet çemberi tekniği için en uygun olmayan bireyin 1'inci, en uygun bireyin n'inci olduğu durumda sıralama doğrusal olarak Eşitlik (3.5) ile ölçeklenebilmektedir.

$$\text{Rank(Pos)} = 2\text{-SP} + \left(2\,(\text{SP-1})\ \frac{(\text{Pos-1})}{(\text{n-1})}\right) \tag{3.5}$$

Sıralama tabanlı seçim şemaları erken yakınsamadan kaçınmakta, uygunluk değerlerinin ölçeklemesine ihtiyacı ortadan kaldırmakta, ancak sıralama gereksiniminden dolayı hesapsal olarak yüksek karmaşıklığa sahip olmaktadırlar (Razali ve Geraghty, 2011).

Uygunluk değerine uygulanan rulet çemberi yöntemi ile sıralama tabanlı rulet çemberi yöntemleri Şekil 3.1 ile kıyaslanmaktadır.



Şekil 3.1. Rulet çemberi ve sıralama tabanlı rulet çemberi olasılıkları

Kullanılan bir diğer yöntem ise turnuva seçimi yöntemidir. Turnuva seçimi yönteminde rastgele olarak seçilen belirli bir sayıdaki bireyden en uygun olanı seçilmektedir (Schmitt, 2001). Bu yöntem uygunluk fonksiyonu özelliklerine hassasiyet göstermemesi, basitliği, ön hesaplamaya gereksinim duymaması, paralelleştirilebilmeye oldukça yatkın olması ve seçicilik hususunda uyarlanabilmesi sebebi ile sıklıkla kullanılmaktadır (Luke, 2011).

Turnuva seçim yönteminin temel adımları şu şekilde verilmektedir:

Adım 1: Turnuva büyüklüğü t belirle.
Adım 2: Eşit olasılıkla bir bireyi(X_b) rastgele seç.
Adım 3: 2'den t ye kadar adım 4 ve adım 5 adımlarını uygula:
Adım 4: Eşit olasılıkla bir bireyi(X_c) rastgele seç.
Adım 5: Eğer $C(X_c) < C(X_b)$ ise $X_b = X_c$.
Adım 6: X_b bireyini geri dön.

Yöntemin kritik parametresi turnuva büyüklüğüdür. Turnuva büyüklüğünün 1 seçilmesi durumunda popülasyondan rastgele bir seçim yapılırken turnuva büyüklüğünün oldukça büyük seçilmesi durumunda en uygun bireyin turnuva seçimi yöntemi tarafından seçilmesi ihtimali artmakta ve limit durumda 1 olmaktadır. Bu durum yöntemin esnekliğini ve uyarlanabilirliğini göstermektedir (Luke, 2011). Yöntem uygulamada sıklıkla turnuva büyüklüğü iki değeri ($t = 2$) alınarak kullanılmaktadır ve bu duruma ikili turnuva yöntemi adı verilmektedir (Razali ve Geraghty, 2011).

Diğer bir yöntem de seçkinler seçimdir. Bu seçim yönteminde uygunluk değeri en uygun olan bireylerin belirli bir yüzdesi ebeveyn olarak seçilmektedir (Chudasama ve diğerleri, 2011). Uygunluk değerlerine göre sıralanan kromozomlar, düzenlenmiş kümede kromozomların ikişerli olarak seçilmektedir. Bu yöntemde diğer operatörler güçlü kromozomlar yine güçlü olanlar ile birlikte, zayıf olan kromozomlar zayıf olan kromozomlar ile birlikte olacak biçimde uygulanmaktadır. Bu yöntemde zayıf ve güçlü kromozomlara diğer genetik operatörlerin uygulanma şansı bulunmamaktadır (Alabsi ve Naoum, 2012).

Seçim yöntemlerinin kıyaslanması açısından hesapsal karmaşıklıkları rulet çemberi için $O(n^2)$, sıralama seçimi için $O(n \ln n)$ + seçim süresi, turnuva seçimi için $O(n)$, seçkinler seçimi için $O(m\, n^2)$ olmaktadır (Sharma ve diğerleri, 2014).

3.3.2. Çaprazlama operatörü

Çaprazlama operatörleri kromozomların problemi temsil yöntemi ile doğrudan alakalıdır. Farklı temsil yöntemlerine farklı çaprazlama yöntemleri uygulanabilmektedir. Genel olarak kullanılan çaprazlama yöntemleri şunlardır:

- Bir noktalı çaprazlama
- İki noktalı çaprazlama
- Tekdüze çaprazlama
- Düzgün(Mendes, 2013) çaprazlama
- Çevrim(Otman ve Jaafar, 2011) çaprazlama
- Parçalı haritalamalı(Otman ve Jaafar, 2011) çaprazlama
- Tekdüze parçalı haritalamalı(Otman ve Jaafar, 2011) çaprazlama
- Sıralı(Otman ve Jaafar, 2011) çaprazlama
- Sarılı olmayan sıralı(Otman ve Jaafar, 2011) çaprazlama

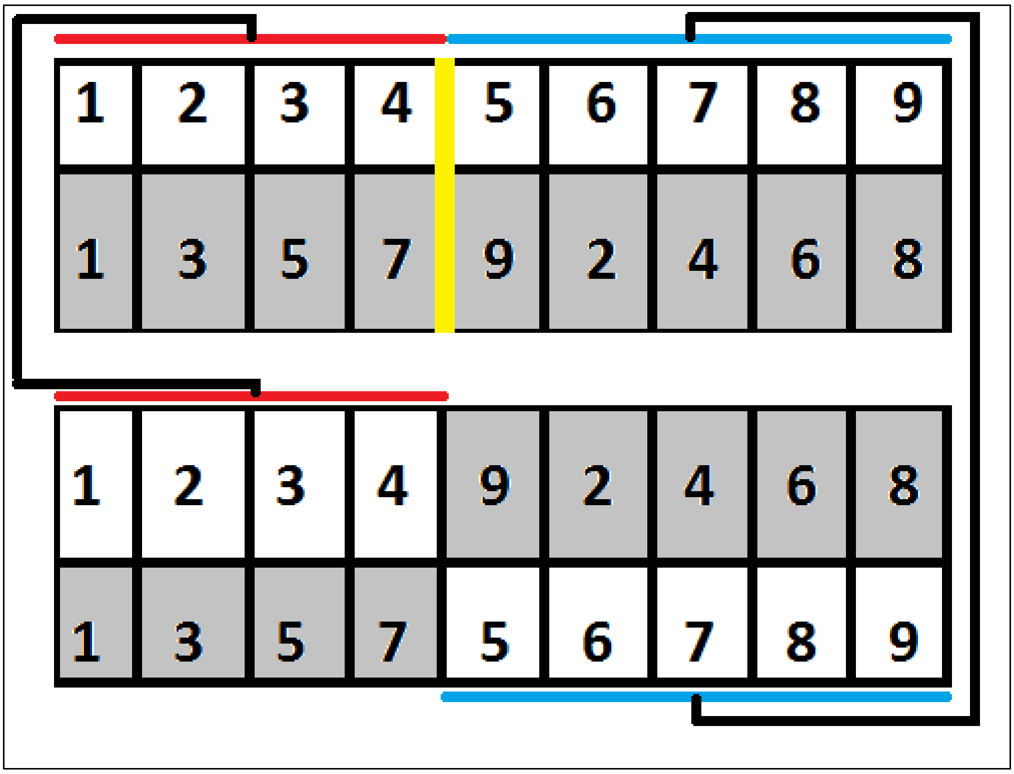
Bunların yanı sıra literatürde görülen diğer çaprazlama yöntemleri şu şekildedir:

- Üç ebeveynli çaprazlama
- N-Noktalı çaprazlama
- Kes ve birleştir çaprazlama
- Yarı tekdüze çaprazlama
- Aritmetik çaprazlama
- Sezgisel çaprazlama
- Öz adaptif çaprazlama
- Bilgi tabanlı çaprazlama
- His çaprazlama
- Orta nokta çaprazlama
- Rehberli çaprazlama
- Simpleks çaprazlama
- Tarama çaprazlama
- Köşegen çaprazlama
- Tekmodlu normal dağıtılmış çaprazlama
- Benzetilmiş 0-1 çaprazlama
- Laplace çaprazlama

- Ebeveyn merkezli çaprazlama
- Orta seviye çaprazlama
- Halka çaprazlama

Çaprazlama yöntemlerinden bazıları sadece ikil ve reel kodlama ile temsil edilen kromozom yapılarına uygulanabilirken, bazıları da sadece devşirim ile temsil edilen kromozom yapılarına uygulanabilmektedir.

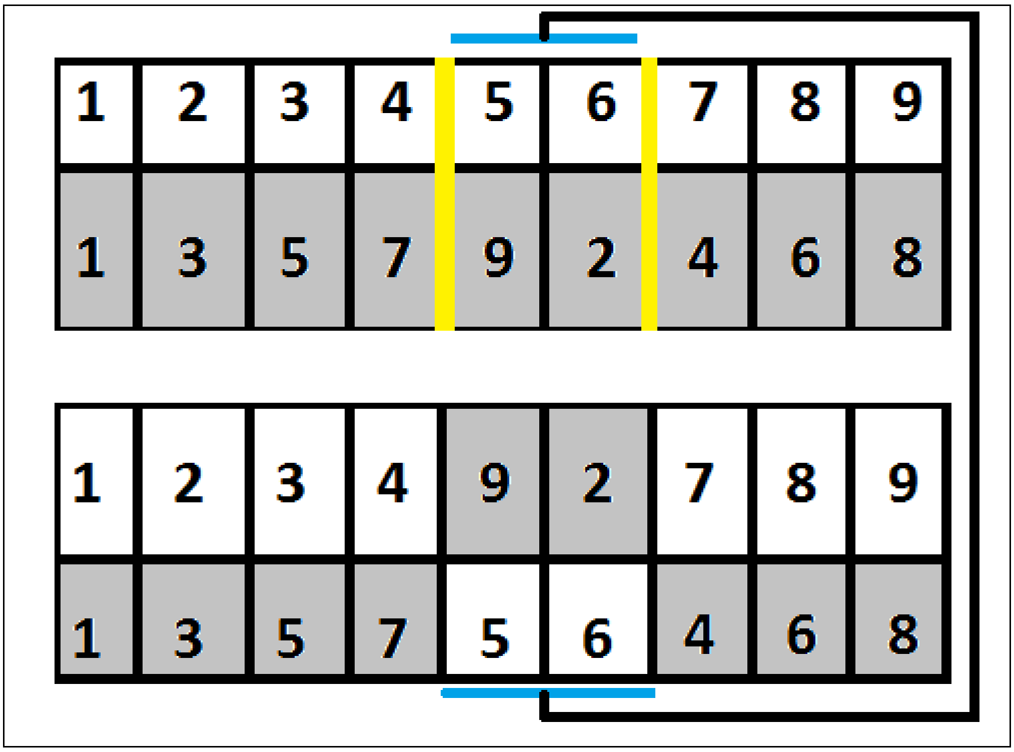
Bir noktalı çaprazlama, ebeveynlerin kromozom dizisi üzerinde bir nokta seçilerek birinci çocuk bireyin seçilen noktaya kadar olan genlerinin ilk ebeveynden, seçilen noktadan sonraki genlerin ikinci ebeveynden alınarak; ikinci çocuk bireyin seçilen aynı noktaya kadar olan genlerinin ikinci ebeveynden, seçilen noktadan sonraki genlerin birinci ebeveynden alınarak oluşturulması işlemidir. Bir noktalı çaprazlama işlemi Şekil 3.2 ile gösterilmektedir.



Şekil 3.2. Tek noktalı çaprazlama

Bir noktalı çaprazlama genellikle devşirim tipi kromozom yapılarında bozulmaya ve dolayısıyla düzeltme için ek işlem yüküne yol açtığından dolayı tercih edilmemektedir.

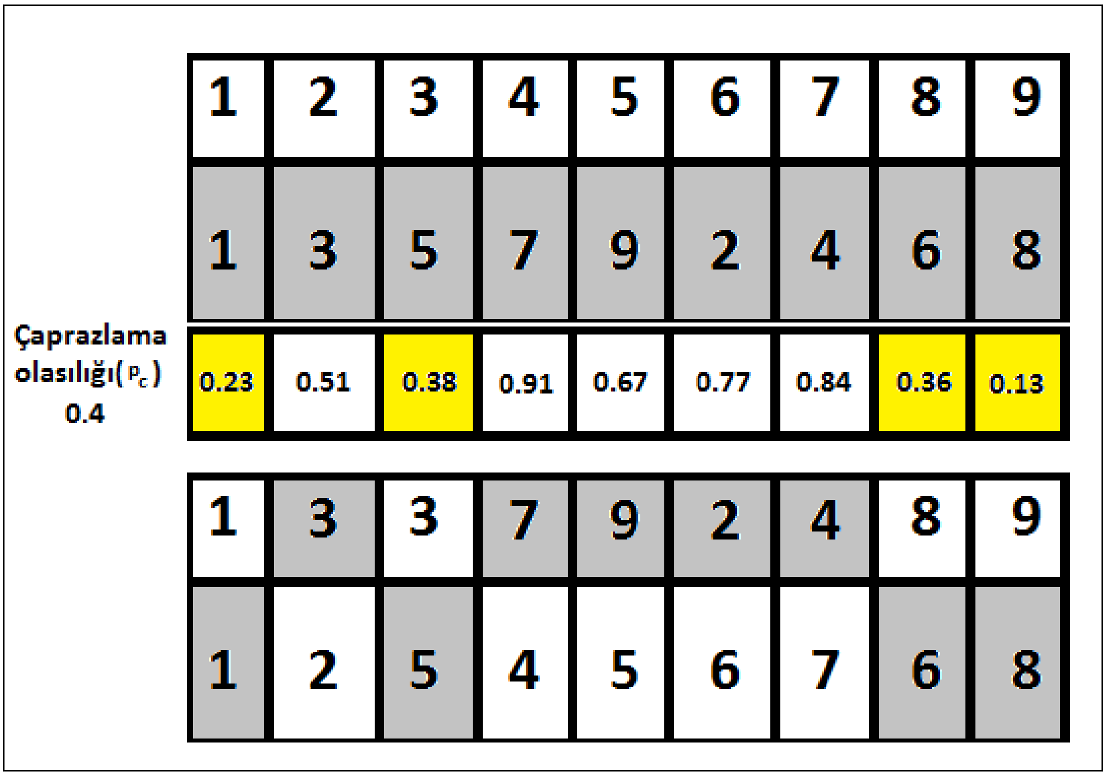
İki noktalı çaprazlama, rastgele iki farklı nokta belirlemek haricinde bir noktalı çaprazlamaya oldukça benzemektedir (Mendes, 2013). İki noktalı çaprazlama, ebeveynlerin kromozom dizisi üzerinde iki farklı nokta seçilerek birinci çocuk bireyin seçilen iki nokta arası hariç olan genlerinin ilk ebeveynden, seçilen iki nokta arasındaki genlerin ise ikinci ebeveynden alınarak; ikinci çocuk bireyin seçilen iki nokta arası hariç genlerinin ikinci ebeveynden, seçilen iki nokta arasındaki genlerin ise birinci ebeveynden alınarak oluşturulması işlemidir. İki noktalı çaprazlama işlemi Şekil 3.3 ile gösterilmektedir.



Şekil 3.3. İki noktalı çaprazlama

İki noktalı çaprazlama işlemi de genellikle devşirim tipi kromozom yapılarında bozulmaya ve dolayısıyla düzeltme için ek işlem yüküne yol açtığından dolayı tercih edilmemektedir.

Tekdüze çaprazlama, ebeveynlerin kromozom dizisi üzerinden genlerin, üretilen bir rastgele sayıya göre ilk çocuğa p_c olasılıkla birinci ebeveynden, $(1-p_c)$ olasılıkla ikinci ebeveynden; ikinci çocuğa üretilmiş olan aynı rastgele sayıya göre $(1-p_c)$ olasılıkla birinci ebeveynden, p_c olasılıkla ikinci ebeveynden alınarak çocukların oluşturulması işlemidir. Tekdüze çaprazlama işlemi Şekil 3.4 ile gösterilmektedir.



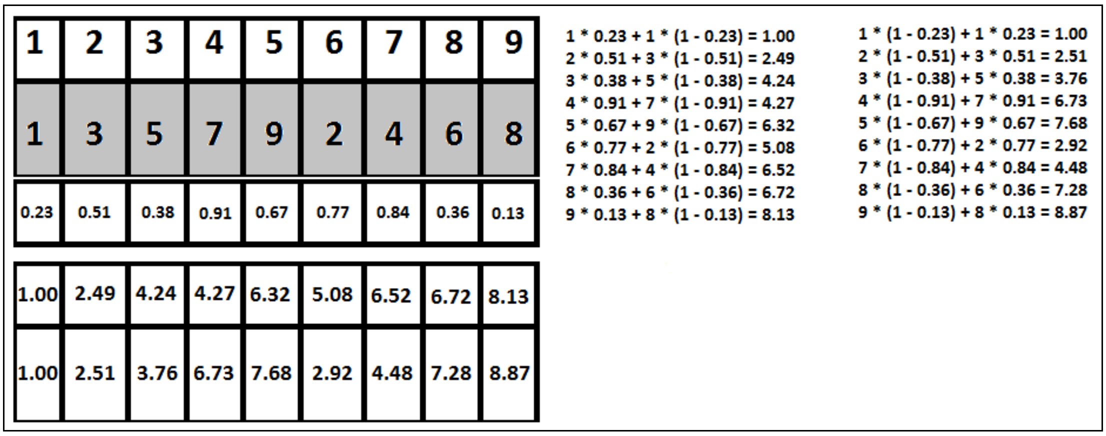
Şekil 3.4. Tekdüze çaprazlama

Tekdüze çaprazlama işlemi de genellikle devşirim tipi kromozom yapılarında bozulmaya ve dolayısıyla düzeltme için ek işlem yüküne yol açtığından dolayı tercih edilmemektedir.

Düzgün çaprazlama, ebeveynlerin kromozom dizisi üzerinden her gen için üretilen bir rastgele sayı ile çocuk bireyin genlerinin doğrusal birleşimi(combination) biçiminde oluşturulması işlemidir (Mendes, 2013). Çocuk bireyin her geni Eşitlik (3.6) ile hesaplanmaktadır.

$$\text{Çocuk}_i = r_i \,\text{Ebeveyn1}_i + (1 - r_i)\,\text{Ebeveyn2}_i \tag{3.6}$$

Eşitlikte r_i değeri i geni için üretilen rastgele sayıyı ifade etmektedir. Düzgün çaprazlama işlemi Şekil 3.5 ile gösterilmektedir.



Şekil 3.5. Düzgün çaprazlama

Düzgün çaprazlama işlemi devşirim tipi ve tamsayılı kromozom yapılarında bozulmaya ve dolayısıyla düzeltme için ek işlem yüküne yol açtığından dolayı tercih edilmemektedir.

Çevrim çaprazlama, devşirim tipi kromozom yapılarında çalışmak üzere geliştirilmiş bir çaprazlama türüdür. Bu çaprazlamada çocuk genleri ve gen konumu, ebeveynlerden bir tanesinden gelmektedir (Otman ve Jaafar, 2011). Yöntem bir tur oluşana kadar çocuk genlerinin ebeveynlerin birisinden, tur oluştuktan sonra boşta kalan genlerin ise diğer ebeveynden alınması üzerine kurulmuştur. Ebeveynlerden birincisinin ilk geni çocuğa aynen aktarılmakta ve bu gen konumuna diğer ebeveynde bakılmaktadır. Diğer ebeveyndeki gen değeri, ilk ebeveynde bulunmakta ve bu adımlar bir çevrim oluşturana kadar devam etmektedir. Çevrim oluştuğunda çocuk bireyde bazı genler boş kalabilmektedir. Bu durumda boş kalan genler sırasıyla ikinci ebeveynden alınmaktadır.

Çevrim çaprazlama algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: Birinci ebeveyn p, ikinci ebeveyn q, oluşturulacak çocuk birey c, kullanılacak geçici değişken g olsun.

Adım 2: $c_1 = p_1$, $g = q_1$

Adım 3: $p_j = g$ eşitliğini sağlayan j indisini bul.

Adım 4: $c_j = p_j$ ata

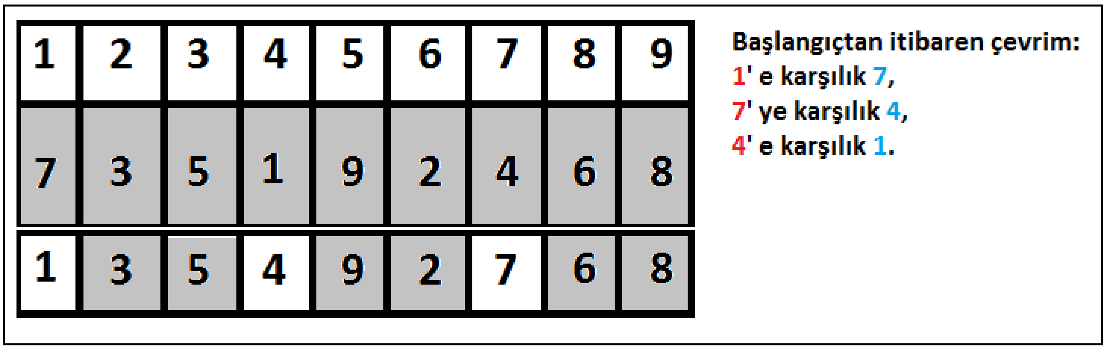
Adım 5: Eğer q_j geni, c'de bulunuyorsa Adım 7'ye git.

Adım 6: $g = q_j$ ve Adım 3'e git.

Adım 7: c'de bulunan genleri q'dan sil.

Adım 8: j sırasıyla c'deki atanmamış genleri gösterecek biçimde; i sırasıyla q'da bulunan genleri gösterecek biçimde $c_j = q_i$ ata.

Çevrim çaprazlama işlemi Şekil 3.6 ile gösterilmektedir.



Şekil 3.6. Çevrim çaprazlama

Parçalı haritalamalı çaprazlama da devşirim tipi kromozom yapılarında çalışmak üzere geliştirilmiş bir diğer çaprazlama türüdür. Bu çaprazlamada ebeveynleri üç parçaya ayıran rastgele iki nokta seçilmektedir (Otman ve Jaafar, 2011). İki noktalı çaprazlamada olduğu gibi ortada kalan parçalar değiştirilerek çocuk bireyler oluşturulmaktadır. Ancak oluşan çocuk bireyler devşirim kuralına uymadığından dolayı yöntem bir düzeltme işlemi kullanmaktadır. Çocuklarda, yer değiştiren genlerden bazıları tekrar etmektedir. Yer değiştiren ve tekrar eden parçaların oluşturdukları çevrimler haritalanabilmektedir. Bu haritalama işleminin başlangıç ve bitiş noktaları önem taşımakta, çocuk bireyde, yer değiştirmemiş genler haritanın başlangıç ve bitiş noktalarına bakılarak değiştirilmektedir.

Parçalı haritalamalı çaprazlama algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: Birinci ebeveyn p, ikinci ebeveyn q, oluşturulacak çocuk birey c, kullanılacak geçici değişkenler g ve h olsun.

Adım 2: Rastgele kromozom kesim noktaları r_1 ve r_2 ata.

Adım 3: c = p ata.

Adım 4: $i = r_1$

Adım 5: $c_i = q_i$

Adım 6: Eğer i = r_2 değil ise i = i + 1 ve Adım 5'e git.

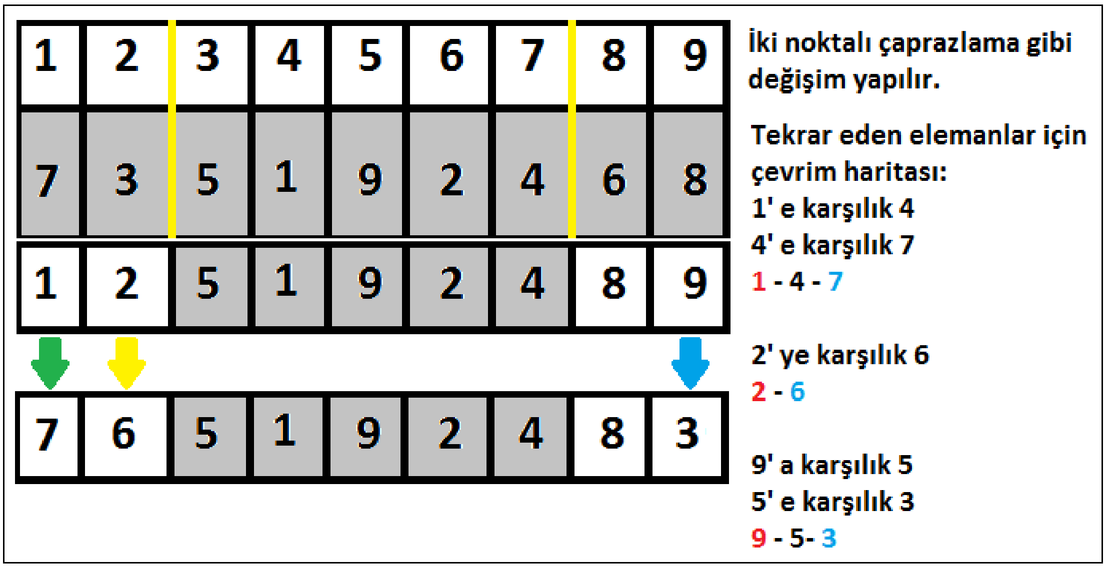
Adım 7: c'de tekrar eden gen varsa bul ve g'ye ata, yoksa algoritmayı bitir.

Adım 8: g'yi r_1 ile r_2 aralığında haritala ve haritanın bitiş genini bul ve h'ye ata.

Adım 9: c'de r_1 ile r_2 aralığı dışında g'yi bul ve bu geni h ile değiştir.

Adım 10: Adım 7'ye git.

Parçalı haritalamalı çaprazlama işlemi Şekil 3.7 ile gösterilmektedir.



Şekil 3.7. Parçalı haritalamalı çaprazlama

Tekdüze parçalı haritalamalı çaprazlama da devşirim tipi kromozom yapılarında çalışmak üzere geliştirilmiş parçalı haritalamalı çaprazlamaya oldukça benzeyen bir çaprazlama türüdür. Bu çaprazlama, parçalı haritalamalı çaprazlamayı kullanmakta, çaprazlanacak genlerin seçiminde iki nokta belirlemek yerine her gen için - tekdüze çaprazlamada olduğu gibi - bir olasılık değerine göre belirlemektedir (Otman ve Jaafar, 2011). Tekdüze çaprazlamada olduğu gibi bir genin hangi ebeveynden alınacağı belirlenerek çocuk bireyler oluşturulmaktadır. Ancak oluşan çocuk bireyler devşirim kuralına uymadığından dolayı yöntem bir düzeltme işlemi kullanmaktadır. Çocuklarda, yer değiştiren genlerden bazıları tekrar etmektedir. Bu noktadan sonra yapılan işlem, parçalı haritalamalı çaprazlamada olduğu gibi ebeveynlerden bir tanesi referans alınarak değiştirilen genlerin korunması, değiştirilmemiş ancak devşirim yapısını bozan genlerin ise haritalama ile değiştirilecek karşılıklarının bulunarak değiştirilmesidir.

Tekdüze parçalı haritalamalı çaprazlama algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: Birinci ebeveyn p, ikinci ebeveyn q, oluşturulacak çocuk birey c, rastgele değişim yöneyi r, kromozom uzunluğu n, kullanılacak geçici değişkenler g ve h olsun.

Adım 2: Rastgele değiştirilecek gen dizisi r belirle.

Adım 3: c = p ata.

Adım 4: i = 1

Adım 5: Eğer $r_i = 1$ ise $c_i = q_i$

Adım 6: Eğer i = n değil ise i = i + 1 ve Adım 5'e git.

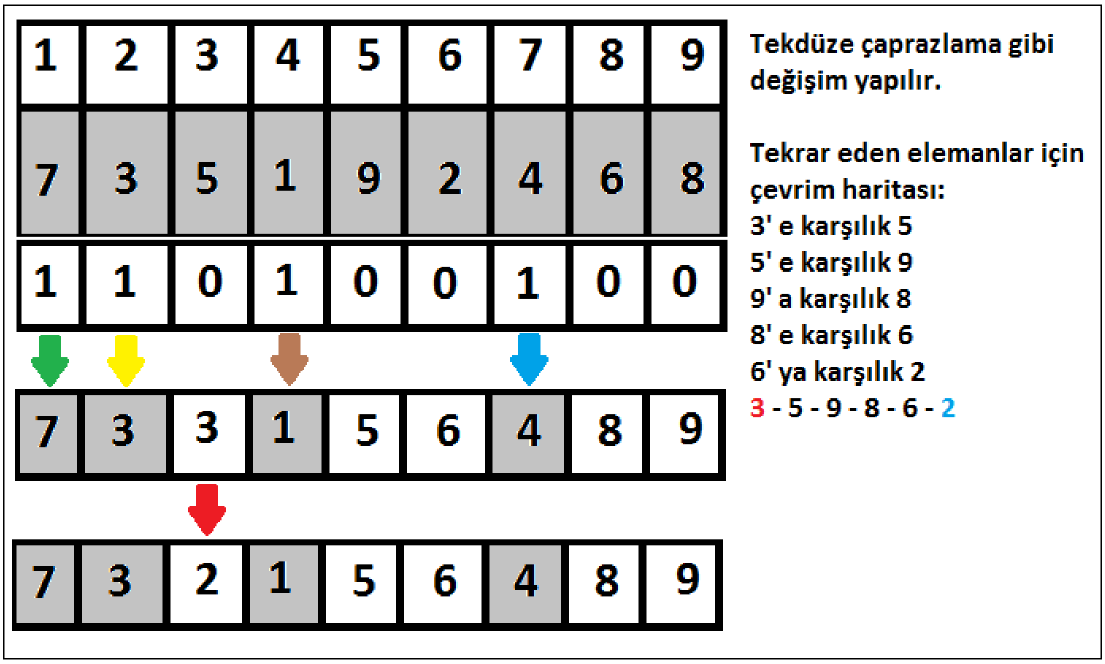
Adım 7: c'de tekrar eden gen varsa bul ve g'ye ata, yoksa algoritmayı bitir.

Adım 8: g'yi r yöneyindeki elemanlara göre haritala ve haritanın bitiş genini bularak h'ye ata.

Adım 9: c'de r'nin 1 olmadığı durumda g'yi bul ve bu geni h ile değiştir.

Adım 10: Adım 7'ye git.

Tekdüze parçalı haritalamalı çaprazlama işlemi Şekil 3.8 ile gösterilmektedir.



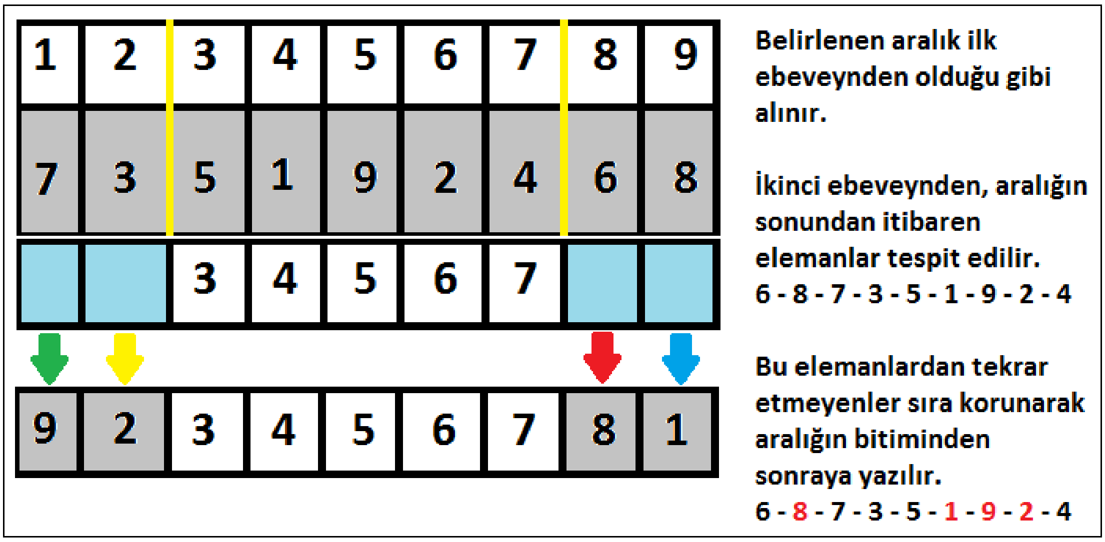
Şekil 3.8. Tekdüze parçalı haritalamalı çaprazlama

Devşirim tipi kromozom yapılarında çalışmak üzere geliştirilmiş bir diğer çaprazlama türü de sıralı çaprazlamadır. Bu çaprazlama problem yapısı sıralama tabanlı ise -örneğin U biçimli montaj hattı dengeleme problemi v.b.- kullanılmaktadır (Otman ve Jaafar, 2011). Yöntem rastgele iki nokta belirleyerek ebeveynlerde bu noktalar aralığında kalan genlerin diğer ebeveynde silinmesi ile işleme başlamaktadır. Bu durumda oluşan boşluklar, ikinci adımda her ebeveynin kendisine ait silinmemiş genler ile aralığın sonundan başlanarak sıra bozulmadan doldurulmakta, üçüncü adımda aralıktaki genler karşılıklı olarak değiştirilerek devşirim yapısının korunması sağlanmaktadır.

Sıralı çaprazlama algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: Birinci ebeveyn p, ikinci ebeveyn q, oluşturulacak çocuk birey c, kullanılacak geçici değişken g olsun.
Adım 2: Rastgele kromozom kesim noktaları r_1 ve r_2 ata.
Adım 3: c = p ata.
Adım 4: $i = r_1$
Adım 5: c'de q_i bul ve ilgili gen boş kalacak biçimde sil.
Adım 6: Eğer $i = r_2$ değil ise $i = i + 1$ ve Adım 5'e git.
Adım 7: c'de kalan genleri r_2+ 1'den itibaren sırala.
Adım 8: c'de r_1 ila r_2 aralığına q'da r_1 ila r_2 aralığını ata.

Sıralı çaprazlama işlemi Şekil 3.9 ile gösterilmektedir.



Şekil 3.9. Sıralı çaprazlama

Sarılı olmayan sıralı çaprazlama da devşirim tipi kromozom yapılarında çalışmak üzere geliştirilmiş, sıralı çaprazlamanın neredeyse aynısı sayılabilecek olan diğer bir çaprazlama türüdür. Bu çaprazlamada çocuk bireyin oluşturulması için prensip, ebeveyn genlerinin mutlak düzenlerinin korunarak boşlukların oluşturulması ve doldurulmasıdır (Otman ve Jaafar, 2011). Yöntem rastgele iki nokta belirleyerek ebeveynlerde bu noktalar aralığında kalan genlerin diğer ebeveynde silinmesi ile işleme başlamaktadır. Bu durumda oluşan boşluklar, ikinci adımda her ebeveynin kendisine ait silinmemiş genler ile baştan başlanarak, belirlenmiş olan aralık boş bırakılmak kaydıyla sıra bozulmadan doldurulmakta, üçüncü adımda aralıktaki genler karşılıklı olarak değiştirilerek devşirim yapısının korunması sağlanmaktadır.

Sarılı olmayan sıralı çaprazlama algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: Birinci ebeveyn p, ikinci ebeveyn q, oluşturulacak çocuk birey c, kullanılacak geçici değişken g olsun.

Adım 2: Rastgele kromozom kesim noktaları r_1 ve r_2 ata.

Adım 3: c = p ata.

Adım 4: $i = r_1$

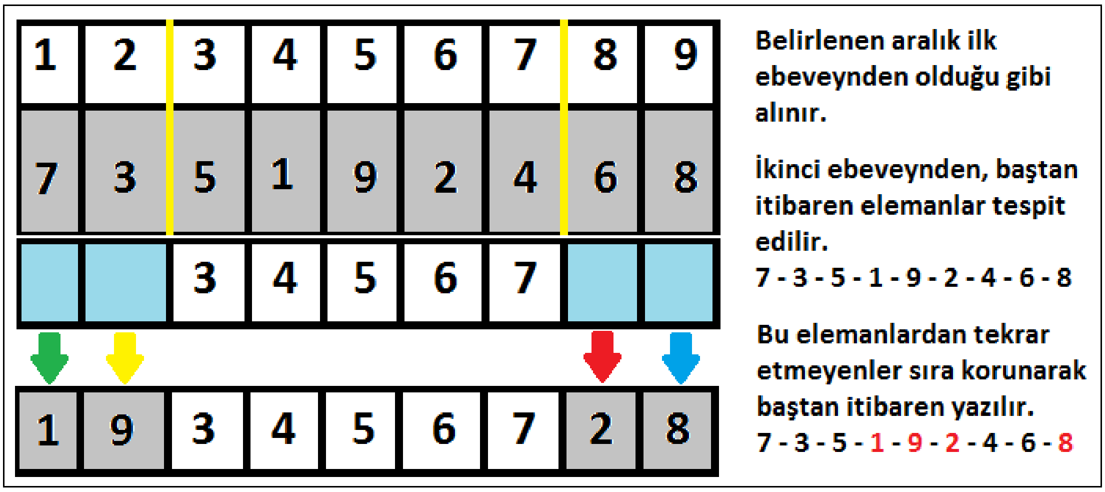
Adım 5: c'de q_i bul ve ilgili gen boş kalacak biçimde sil.

Adım 6: Eğer $i = r_2$ değil ise $i = i + 1$ ve Adım 5'e git.

Adım 7: c'de kalan genleri baştan itibaren r_1 ila r_2 aralığı boş kalacak şekilde sırala.

Adım 8: c'de r_1 ila r_2 aralığına q'da r_1 ila r_2 aralığını ata.

Sarılı olmayan sıralı çaprazlama işlemi Şekil 3.10 ile gösterilmektedir.



Şekil 3.10. Sarılı olmayan sıralı çaprazlama

Diğer çaprazlama yöntemleri ile ilgili bilgi için (Geetha ve Muthukumaran, 2013) çalışmasına bakılabilir.

3.3.3. Değişinim operatörü

Değişinim operatörü de çaprazlama operatörü gibi kromozomların problemi temsil yöntemi ile doğrudan alakalıdır. Değişinim operatörünün temel amacı popülasyondaki bireylerin genetik olarak birbirlerinden farklı olmasını sağlamaktır. Değişinim operatörü kromozom üzerinde ufak değişiklikler yapmakta ve değişinime uğramadan önceki bireye çok benzeyen yeni bir birey üretmektedir. Benzerlik gen bazında olabileceği gibi görünüş bazında da olabilir.

Genel olarak kullanılan değişinim yöntemleri şunlardır:

- Bit değiştirme değişinim
- Rastgele değişinim
- Sınır değişinim
- Tekdüze değişinim
- Tekdüze olmayan değişinim
- Gauss değişinim
- Novel değişinim

- Bit dizisi değişinim
- Tek noktalı değişinim
- İki noktalı değişinim
- Komşu değişinim
- Öteleme değişinim
- Değiş tokuş değişinim
- Alt liste karıştırma değişinim
- Devşirim için ekleme değişinim
- Devşirim için değiştirme değişinim
- Devşirim için tersleme değişinim
- Devşirim için karıştırma değişinim

Kullanılacak değişinim operatörü genellikle kromozom yapılarında olursuz çözümlere yol açmayacak biçimde seçilmektedir. Bazı değişinim türleri sadece devşirim türü kodlama ile temsil edilen kromozom yapılarında kullanılabilirken, bazıları ikil kodlama ile temsil edilen kromozom yapılarında, bazı değişinim türleri de reel veya tamsayılı kromozom yapılarında kullanılabilmektedir.

Bit değiştirme değişinim birey üzerinde seçilen genin değerinin değillenmesi işlemidir. Bu değişinim yöntemi 0 olan bir gen değerini 1, 1 olan bir gen değerini de 0 yapmaktadır. Bu operatör sadece ikil kodlama ile temsil edilen kromozom yapılarında kullanılmaktadır (Geetha ve Muthukumaran, 2013).

Rastgele değişinim birey üzerinde seçilen genin değerinin rastgele değiştirilmesi işlemidir (Geetha ve Muthukumaran, 2013).

Sınır değişinim birey üzerinde seçilen genin değerinin, genin alabileceği en alt veya en üst sınır değer ile değiştirilmesi işlemidir. Bu operatör sadece reel veya tamsayılı kromozom yapılarında kullanılmaktadır (Geetha ve Muthukumaran, 2013).

Tekdüze değişinim birey üzerinde seçilen genin değerinin, genin alabileceği en alt ve en üst sınır değerler arasında veya kullanıcının belirlediği bir aralıkta tekdüze dağılıma uygun olarak değiştirilmesi işlemidir. Bu operatör sadece reel veya tamsayılı kromozom yapılarında kullanılmaktadır (Geetha ve Muthukumaran, 2013).

Tekdüze olmayan değişinim yineleme sayısı arttıkça değişinim miktarı 0'a yakın olacak biçimde olasılığı arttırma işlemidir. Bu operatör sadece reel veya tamsayılı kromozom yapılarında kullanılmaktadır (Geetha ve Muthukumaran, 2013).

Gauss değişinim birey üzerinde seçilen genin değerine gauss dağılıma uygun rastgele değerin eklenmesi işlemidir. Yeni gen değeri eğer genin alabileceği en alt ve en üst sınır değerleri geçmiş ise gen değeri bu sınıra çekilir. Bu operatör sadece reel veya tamsayılı kromozom yapılarında kullanılmaktadır (Geetha ve Muthukumaran, 2013).

Novel değişinim, geleneksel değişinim operatörlerinin etkin olamadığı küresel en uygunlama problemlerinin çözümü için önerilmiştir. Değişinim operatörlerinin öngörememe durumlarını bertaraf etmek için küresel işlemde araya girerek var olan bilgiden faydalanmaktadır. Beş adım ile çalışmaktadır. Bu adımlar hazırlık, bilgi çıkarımı, değişinim, seçim ve tekrarlama adımlarıdır (Geetha ve Muthukumaran, 2013).

Bit dizisi değişinimi rastgele genlerin değiştirilmesi yoluyla sağlanır. İkil yöney uzunluğu l olmak üzere bir bitin değişinim olasılığı 1/l'dir (Geetha ve Muthukumaran, 2013). Bu operatör ikil kromozom yapılarında kullanılmaktadır.

Tek noktalı değişinim birey üzerinde seçilen genin değerinin, 1'den 0'a değiştirilmesidir (Geetha ve Muthukumaran, 2013). Bu operatör ikil kromozom yapılarında kullanılmaktadır.

İki noktalı değişinim birey üzerinde seçilen iki genin değerlerinin, 1'den 0'a veya 0'dan 1'e değiştirilmesidir (Geetha ve Muthukumaran, 2013). Bu operatör ikil kromozom yapılarında kullanılmaktadır.

Komşu değişinim birey üzerinde komşu genlerin değiştirilmesidir (Geetha ve Muthukumaran, 2013). Bu operatör hemen hemen tüm kromozom yapılarında kullanılmaktadır.

Öteleme değişinim ebeveynde rastgele bir değer seçmekte ve bu değeri rastgele seçilen bir noktaya ekleyerek ekleme noktasından sonraki genleri sağa ötelemektedir (Geetha ve Muthukumaran, 2013). Bu operatör hemen hemen tüm kromozom yapılarında kullanılmaktadır.

Değiş tokuş değişinim devşirimden iki rastgele geni değiştirme işlemidir. Kromozomda kalan diğer genlere dokunulmamaktadır (Geetha ve Muthukumaran, 2013). Bu operatör hemen hemen tüm kromozom yapılarında kullanılmaktadır.

Alt liste karıştırma değişinim kromozomdan rastgele iki nokta seçmekte ve iki nokta arasında kalan genleri rastgele devşirmektedir (Geetha ve Muthukumaran, 2013). Bu operatör hemen hemen tüm kromozom yapılarında kullanılmaktadır.

Devşirim için ekleme değişinim kromozomdan rastgele iki gen seçmekte ve seçilen ikinci geni seçilen birinci geni izleyecek biçimde hareket ettirip diğer genleri ötelemektedir. Bu işlem sıralama ve komşuluk bilgisini büyük oranda muhafaza etmektedir (Geetha ve Muthukumaran, 2013).

Devşirim için değiştirme değişinim kromozomdan rastgele iki gen seçmekte ve seçilen genlerin konumlarını değiştirmektedir. Bu işlem komşuluk bilgisini büyük oranda korur ve sıralamayı sekteye uğratır. Komşuluk bilgisinde dört bağlantıyı bozarak dört yeni bağlantı oluşturmaktadır (Geetha ve Muthukumaran, 2013).

Devşirim için tersleme değişinim kromozomdan rastgele iki gen seçmekte ve seçilen genler arasında kalan alt diziyi ters çevirmektedir. Bu işlem komşuluk bilgisini büyük oranda korur ve sıralamayı sekteye uğratır. Komşuluk bilgisinde iki bağlantıyı bozarak iki yeni bağlantı oluşturmaktadır (Geetha ve Muthukumaran, 2013).

Devşirim için karıştırma değişinim genlerin bir alt kümesini rastgele seçmekte ve bu konumlara seçilen genleri tekrar düzenlemektedir (Geetha ve Muthukumaran, 2013).

3.4. Yapay Bağışıklık Algoritması

Yapay bağışıklık klonal seçme algoritması doğal bağışıklık sistemini taklit etme prensibine dayanmaktadır. Doğal bağışıklık sistemi temel olarak tekillik, sapaklık tespiti, dağıtılmış tespit, mükemmel olmayan tespit, takviyeli öğrenme, farklılaşma, dış çevreye uyum sağlama ve hafızaya sahip olma özelliklerine sahiptir (Karaboğa, 2014). Doğal bağışıklık sisteminde antijenler sisteme girdiğinde antikorların üretimi ile antijenler etkisizleştirilmektedir. Üretilen antikorların sisteme giren antijene uygun olması gerekmekte, dolayısı ile antijeni etkisiz kılabilecek antikorlar daha fazla üretilmektedir. Antijene benzeme durumu uygunluk fonksiyonu ile, antikorlar ise çözümler ile bağdaştırılmaktadır.

Yapay bağışıklık klonal seçme algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: Eğer varsa hafıza hücrelerinin bir kısmını(M) popülasyona(P_r) dahil et($P = P_r + M$) ve elde edilen hücreler kümesinden oluşmuş aday çözümler kümesi(P) üret.
Adım 2: Popülasyondan en iyi n adet bireyi belirle.
Adım 3: Popülasyonda bulunan n adet bireyin kopyaları kümesini(K) antijene benzerlikle orantılı olarak oluştur.
Adım 4: Klonları hiperdeğişinim işlemine tabi tut ve olgunlaşmış antikor popülasyonu(K^*) üret.
Adım 5: Hafıza hücre kümesini(M) güncelle.
Adım 6: Popülasyonun düşük bezerlikli antikorlarını yeni antikorlar ile değiştir.
Adım 7: Durdurma kriteri sağlanmadı ise Adım 1'e git (Karaboğa, 2014).

Doğal bağışıklık sisteminin oldukça karışık, geri beslemeli yapıları da mevcuttur. Bu yapıların modellenmesi ile farklı algoritmalar geliştirilerek kullanılabilir (Karaboğa, 2014).

3.5. Karınca Algoritması

Karınca algoritması görme yetileri olmayan karıncaların kendi kolonileri ile yiyecek kaynağı arasındaki en kısa ağ tabanlı yolu aramak için kullandıkları davranışı modellemeyi amaçlamaktadır (Shekhawat, 2009). Feromon izlerini kullanan karıncalar, başlangıçta hiç feromon bulunmayan yollarda rastgele bir davranış sergilemektedirler. Çok sayıda karınca her yol üzerinde farklı miktarlarda feromon birikmesine sebep olmaktadır. Karıncaların feromon seviyesinin yüksek olduğu yolu seçme ihtimalleri artmaktadır. Bir süre sonra karıncaların büyük bölümü kendi koloni merkezleri ile yiyecek kaynağı arasındaki en kısa yolu kullanmaya başlamaktadır. Algoritma, karınca popülasyonu ile çalıştığından dolayı popülasyon tabanlı, her bir karınca yeni bir çözüm oluşturduğundan dolayı da yapıcı algoritmalar sınıfında yer almaktadır. İlk ortaya çıkan karınca algoritması, karınca sistemi adıyla bilinen algoritmadır ve 1991 senesinde Dorigo ve diğerleri tarafından ortaya atılmıştır. Orijinal karınca sisteminin önemi, daha sonra geliştirilmiş olan karınca algoritmalarına model oluşturmasıdır (Cura, 2008). Karınca algoritmaları ağırlıklı olarak kesikli en uygunlama türündeki problemlerde kullanılmaktadır. Sayısal en uygunlama problemlerine yönelik ancak birkaç çalışma mevcuttur (Karaboğa, 2014). Farklı feromon güncelleme kuralları kullanılarak farklı isimler ile adlandırılmakta olan karınca sistemleri ortaya çıkmıştır. Bunlardan en çok bilinenleri karınca sistemi, max-min karınca sistemi, karınca koloni algoritması, karınca yoğunluk algoritması, karınca miktar algoritması ve karınca çevrim algoritmasıdır (Cura, 2008; Karaboğa 2014).

Literatürde karınca algoritmasının temelde gezgin satıcı problemlerinde kullanıldığı görülmekte olup (Dorigo ve Gambardella, 1997; Hlaing ve Khine, 2011) iş sıralama problemlerinde (Flórez ve diğerleri, 2013;Yağmahan ve Yenisey, 2010; Rajendran ve Ziegler, 2004), karesel atama problemlerinde (Özkale ve Fığlalı, 2011; Sahu ve Pandey, 2014) de kullanıldığı görülmektedir.

Karınca sistemi algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: α, β ve ρ parametrelerini kullanıcıdan al, noktalar arasındaki başlangıç feromon miktarını sıfırla ($\tau_{ij}=0$).

Adım 2: Her karıncayı(m adet) başlangıç noktalarına ata.

Adım 3: Her k karıncası($\forall k \in \{1, 2, \ldots, m\}$) için adım 4, 5 ve 6 adımlarını uygula.

Adım 4: Tam çözüm elde edene kadar adım 5 ve adım 6 adımlarını uygula.

Adım 5: k karıncasının bulunduğu i noktasından j noktasına geçiş olasılığını(p_{ij}^{k}) hesapla.

Adım 6: Rastgele sayı üreterek k karıncasının i noktasından hangi noktaya gideceğini k karıncasının olasılıklarına göre belirle.

Adım 7: Tam çözüm elde edildikten sonra k karıncasının feromon değişim dizeyini($\Delta\tau_{ij}^{k}$) güncelle.

Adım 8: Genel feromon değişim dizeyini($\Delta\tau_{ij}$) güncelle.

Adım 9: Genel feromon dizeyini(τ_{ij}) güncelle.

Adım 10: Durdurma kriteri sağlanmadı ise Adım 2'ye git (Cura, 2008).

Algoritmada k karıncasının i noktasından j noktasına geçiş olasılığı hesabı için Eşitlik (3.7) kullanılmaktadır. Eşitlikte α ve β kullanıcının belirlediği ve [0, 1] aralığında bulunan parametrelerdir. Ayrıca η_{ij} değişkeni, i noktasından j noktasına geçişin ne kadar cazip olacağını gösteren bir değişkendir ve bu bilginin kullanımı, bilginin elde edilebilmesi durumunda mümkün olmaktadır. Özellikle gezgin satıcı problemi gibi problemlerde mesafe bilgisi kullanılabilmekte, ancak benzer bir yapıda çözülen iş sıralama problemleri için cazip olma bilgisi kolayca elde edilemeyebilmektedir.

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\alpha} + \eta_{ij}^{\beta}}{\sum_{\forall il \notin \text{Yasaklılar Listesi}_k} \tau_{il}^{\alpha} + \eta_{il}^{\beta}}, & \text{Eğer } (ij) \notin \text{Yasaklılar Listesi}_k \\ 0, & \text{Diğer durumlarda} \end{cases} \tag{3.7}$$

Algoritmada k karıncasının feromon değişim dizeyinin($\Delta\tau_{ij}^{k}$) güncellenmesi adımında probleme özel hesaplama yapılması gerekmektedir. Gezgin satıcı problemi için Eşitlik (3.8) kullanılmaktadır (Cura, 2008).

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k}, & \text{Eğer k karıncası, } (i, j) \text{ yolunu kullandıysa} \\ 0, & \text{Diğer durumlarda} \end{cases} \tag{3.8}$$

Eşitlikte L_k değişkeni k karıncasının tur uzunluğunu temsil etmektedir. Q ise sabit bir sayıdır.

Algoritmanın bir sonraki adımında genel feromon değişim dizeyi hesaplanmaktadır. Bu değer, her karıncanın feromon değişim dizeyleri toplamına eşittir ve Eşitlik (3.9) ile ifade edilmektedir.

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{3.9}$$

Feromon dizeyinin ne kadar değişeceği hesaplandıktan sonra feromon dizeyinin değiştirilmesi adımı bulunmaktadır. Feromon dizeyinin değiştirilmesi, o anda yollarda bulunan mevcut feromon miktarını da göz önüne almakta, mevcut feromon miktarının bir kısmının buharlaşması ile güncelleme yapılmaktadır ve Eşitlik (3.10) ile verilmektedir.

$$\tau_{ij} = \rho\, \tau_{ij} + \Delta\tau_{ij} \tag{3.10}$$

Eşitlikte ρ, buharlaşma parametresi olarak bilinmektedir ve $[0, 1]$ aralığında kullanıcı tarafından belirlenmektedir. Bu parametrenin yüksek tutulması mevcut feromon miktarının çok az bir miktarının buharlaşmasına sebep olmakta, düşük tutulması ise daha fazla feromonun azalması ile sonuçlanmaktadır. Bu parametrenin adaptif olarak kullanılması da mümkündür (Luke, 2011).

3.6. Parçacık Sürü Algoritması

Parçacık sürü algoritması, sürü algoritmalarının üzerinde en çok çalışılmış olanlarından birisidir. Parçacık sürü optimizasyonu ilk olarak J.Kennedy ve R.C.Eberhart tarafından geliştirilmiş, popülasyon tabanlı bir tekniktir (Özsağlam ve Çunkaş, 2008). Yöntem, parçacıkların hem birbirleri ile bilgi alışverişi yapmalarını hem de kendi geçmiş bilgilerini kullanmalarını sağlamaktadır. Algoritmada her bir parçacığın belirli bir hızı ve konumu bulunmaktadır. Parçacıklar, hızları ve konumlarına göre bir sonraki konumlarına doğru hareket ederler ve diğer parçacıklar ile haberleşerek hızlarını güncellerler. Bütün parçacıklar geçmişte gitmiş oldukları çözümler ile ilgili bilgilere sahiptirler ve birbirleri ile haberleşerek olası uygun çözümlere doğru yönelmektedirler. Popülasyon tabanlı olduğundan, birden çok sayıda parçacık aynı uygun çözümün etrafında yeni uygun çözümler aramakta ve daha uygun bir çözümün bulunması durumunda ise diğer parçacıklar ile haberleşilerek bilinen en uygun çözümün değiştiğini bildirmektedirler. Bunların yanı sıra kendi geçmiş bilgileri ile de hareket eden parçacıklar, kendi buldukları en uygun çözüm etrafında da arama yapmaktadır. Böylece hem sürünün davranışına ayak uydurabilmekte hem de kendi bölgelerinde uygun çözüm arayabilmektedirler.

Yöntemin farklı uygulamaları olmasına rağmen genellikle sürekli değişkenli problemlere uygulanan yöntemin, schaffer fonksiyonu, shubert fonksiyonu, hansen fonksiyonu, camel fonksiyonu gibi sürekli fonksiyonlarda başarılı sonuçlar verdiği bilinmektedir (Yan ve diğerleri, 2012). Algoritma robot navigasyonu, insansız hava araçlarının kontrolü, gibi mühendislik uygulamalarında da kullanılmıştır (Ast ve diğerleri, 2008). Literatürde ayrıca çok amaçlı en uygunlama modelleri için pareto eğrisi elde etmek amacı ile de kullanımı görülmektedir (Coello ve diğerleri, 2004; Reddy ve Kumar, 2007). Kesikli problemlere uygulamaları ile de karşılaşılan yöntemin gezgin satıcı probleminde (Yan ve diğerleri, 2012), sensör yerleşimi probleminde (Rapaic ve diğerleri, 2008), akış tipi iş çizelgeleme probleminde (Ponnambalam ve diğerleri, 2009), iki kriterli depo yerleşim probleminde (Özsoydan ve Saraç, 2011) kullanıldığı görülmektedir.

Parçacık sürü algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: Gerekli parametreleri(α, c_1, c_2, c_3, β) kullanıcıdan al.
Adım 2: Her parçacık için rastgele hız($\overrightarrow{v_k}$) ve konum($\overrightarrow{x_k}$) yöneyleri üret ve parçacık uyguluklarını($C(\overrightarrow{x_k})$) hesapla.
Adım 3: $\overrightarrow{x_k^+}$, $\overrightarrow{x_k^*}$, $\overrightarrow{x^*}$ değerlerini güncelle.
Adım 4: Hız yöneylerini güncelle.
Adım 5: Her bir parçacık için yeni konumları($\overrightarrow{x_k}$) ve uygunluklarını($C(\overrightarrow{x_k})$) hesapla.
Adım 6: Durdurma kriteri sağlanmadı ise adım 3'e git.

Algoritmada 3. adımda yapılan güncellemede; $\overrightarrow{x_k^+}$ konum yöneyi, k parçacığının bilgi alış verişi yaptığı parçacıklardan o yinelemeye kadar bulunmuş olan en uygun parçacık konumunu, $\overrightarrow{x_k^*}$ konum yöneyi k parçacığının kendisinin o yinelemeye kadar bulmuş olduğu en uygun parçacık konumunu, $\overrightarrow{x^*}$ konum yöneyi de tüm parçacıkların o yinelemeye kadar bulmuş olduğu en uygun parçacık konumunu ifade etmektedir. Hız yöneylerinin güncellenmesi adımında Eşitlik (3.11) ile verilen yöney farklarının toplamı ifadesi kullanılmaktadır. r_1, r_2 ve r_3 sayıları(veya yöneyleri) sırasıyla $[0, c_1]$, $[0, c_2]$, $[0, c_3]$ kapalı aralıklarında rastgele sayıl(veya yöney) olarak seçilmektedir. α sayısı mevcut hız yöneyinin etkisini ayarlamakta kullanılmaktadır.

$$\overrightarrow{v_k} = \alpha\ \overrightarrow{v_k} + r_1 \,.\, \left(\overrightarrow{x_k^+} - \overrightarrow{x_k}\right) + r_2 \,.\, \left(\overrightarrow{x_k^*} - \overrightarrow{x_k}\right) + r_3 \,.\, \left(\overrightarrow{x^*} - \overrightarrow{x_k}\right) \tag{3.11}$$

Her bir parçacığın hız yöneylerinin güncellenmesinden sonra konumları güncellenmektedir. Konum güncellemesi için de Eşitlik (3.12) kullanılmaktadır.

$$\overrightarrow{x_k} = \overrightarrow{x_k} + \beta\, \overrightarrow{v_k} \tag{3.12}$$

β parametresi parçacıkların ne kadar hızlı hareket edeceğini ifade etmektedir. Eğer büyük bir sayı seçilirse parçacıklar bir çözüm bölgesinden diğer çözüm bölgesine hızlıca hareket edecektir. Uygulamada genellikle 1 olarak seçilmektedir (Luke, 2008).

3.7. Yapay Arı Koloni Algoritması

Yapay arı koloni algoritması, doğadaki arıların davranışlarından esinlenerek geliştirilmiş olan bir diğer sürü algoritmasıdır. Doğadaki arılar, kendi kendilerine iş bölümü yaparak yiyecek aramaktadırlar. Yiyecek kaynağı bulunduğunda bir değerlendirme yaparak kaynağın değerini belirlerler. Yeni kaynaklar bulmak için kaşif arılar rastgele davranış içerisindedirler. Kaşif arıların yeni bir kaynak bulması durumunda bulunan kaynağın nektar miktarı gibi faktörlere göre kaşif arı dans etmekte ve bu yolla kovandaki diğer arılara haber vermektedir. Kaşif arının dansını gören gözcü arılar işçi arı biçiminde davranmaya başlayarak kaynağa gidebilir veya gözcü arı olarak kalmayı tercih edebilir. İşçi arı yiyeceği topladıktan sonra kovana geri döner ve aynı kaynağa gitmeye veya gitmemeye karar verebilir. İşçi arılar, görevli oldukları kaynak ile ilgili bilgileri kovandaki diğer arılar ile de paylaşmaktadırlar (Karaboğa, 2014).

Yapay arı koloni algoritmasında, arıların konumları yiyecek kaynaklarını temsil etmektedir. Bir kaşif arı bulunduğu konumdaki uygunluk değerini kovandaki gözcü arılara bildirmekte ve işçi arıları bulunduğu kaynağa çekmeye, kaynağın ve kaynak bölgesinin yeterince araştırılmasını sağlamaya çalışmaktadır. İşçi arılar, yeterince araştırma yapılması ile ilgili kriterin sağlandığı durumda kaynak bölgelerini terk etmekte ve kaşif arı olarak yeni bir çözüm oluşturmaktadırlar.

Yapay arı koloni algoritmasının griewank fonksiyonu, rastrigin fonksiyonu, rosenbrock fonksiyonu, ackley fonksiyonu, schwefel fonksiyonu en uygunlamasında başarılı sonuçlar verdiği bilinmektedir (Karaboğa ve Baştürk, 2007). Tıbbi verilerde bulanık kümeleme ile karşılaştırıldığı ve başarılı olduğu görülmektedir (Karaboğa ve Öztürk, 2010). Bir diğer tıbbi uygulamasına ise manyetik rezonans görüntülerinin sınıflandırılması konusunda karşılaşılmakta ve başarılı olduğu bilinmektedir (Zhang ve diğerleri, 2011). Kesikli problemler açısından ise 0-1 tamsayılı çok boyutlu sırt çantası problemine (Sundar ve diğerleri, 2010), akış tipi iş sıralama problemine (Guanlong ve diğerleri, 2012), gezgin satıcı problemine (Pandey ve Kumar, 2013) uygulamaları mevcuttur.

Yapay arı koloni algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: Gerekli parametreleri(limit) kullanıcıdan al ve çözüm geliştirememe sayaçlarını($failure_i$) sıfırla.
Adım 2: Başlangıç çözümleri($\vec{x_i}$) üret ve nektar miktarlarını($C(\vec{x_i})$) hespala.
Adım 3: Her arı için komşu çözüm($\vec{v_i}$) üret ve nektar miktarlarını($C(\vec{v_i})$) hespala.
Adım 4: Açgözlü seçim işlemi ile uygun olan çözümü seç.
Adım 5: İlgili çözüm geliştirememe sayaçlarını($failure_i$) güncelle.
Adım 6: Çözüm geliştirememe sayaçlarının limit parametresi ile kıyasla ve yeni bir çözüm üretip üretmeme kararı ver.
Adım 7: Durdurma kriteri sağlanmadı ise adım 3'e git.

Algoritmanın 3. adımında yapılan komşu çözüm üretme işlemi yöneyin belirli bir uzaklıkta değiştirilmesi işlemidir. Bu işlem için Eşitlik (3.13) kullanılmaktadır. Eşitlikte ϕ_{ij} değeri [0, 1] aralığında rastgele bir sayıdır.

$$v_{ij}=x_{ij}+\phi_{ij}\left(x_{ij}-x_{kj}\right) \tag{3.13}$$

Algoritmada 4. adımda yapılan seçim işlemi, en küçükleme problemleri için daha küçük uygunluk değerine sahip olan arının seçilmesi, en büyükleme problemleri için ise daha büyük uygunluk değerine sahip olan arının seçilmesi anlamına gelmektedir.

Algoritmada 5. adımda çözüm geliştirememe sayaçlarının($failure_i$) güncellenmesi aşaması, algoritmanın 4. adımı ile bağlantılıdır. Eğer 4. adımda oluşturulan komşu çözüm($\vec{v_i}$) ile mevcut çözümün($\vec{x_i}$) karşılaştırılması işlemi sonucunda mevcut çözüm değişmedi ise ilgili çözüm geliştirememe sayacı($failure_i$) bir artırılır, değişmesi durumunda ise ilgili çözüm geliştirememe sayacı($failure_i$) sıfırlanır.

Algoritmanın 6. adımında ise her arının bulunduğu kaynağı terk edip etmeyeceği limit parametresine ve $failure_i$ değerine bakılarak belirlenmektedir. Bulunduğu kaynağı terk edecek olan arılar kaşif arı olup, yeni çözüm($\vec{x_i}$) üretmektedirler (Karaboğa, 2014).

3.8. Diferansiyel Gelişim Algoritması

Diferansiyel gelişim algoritması, sürekli çözüm uzaylarında küresel en uygun çözümün bulunması için geliştirilmiş popülasyon tabanlı basit ve etkili bir algoritmadır (Storn ve Price, 1997). Algoritma temel olarak değişinim, tekrar birleşim, değerlendirme ve seçim adımları ile çalışmaktadır. Algoritmanın, genellikle fonksiyon en uygunlama problemlerinde kullanıldığı görülmektedir (Karaboğa ve Ökdem, 2004; Shamekhi, 2013; Storn ve Price, 1997). Kayan noktalı sayılar ile yöneyler üzerinde basit matematiksel işlemler ile çalışmakta olan algoritma en az dört yöney gerektirmektedir.

Diferansiyel gelişim algoritmasının temel adımları şu şekilde verilmektedir.

Adım 1: Gerekli parametreleri(D, G_{max}, NP, F, CR) kullanıcıdan al ve çözüm yöneyi alt ve üst sınır değerlerini($\overline{x}^{(lo)}$, $\overline{x}^{(hi)}$) ata.
Adım 2: Başlangıç çözüm yöneylerini($\overrightarrow{x_i}$) üret ve uygunluk değerlerini($C(\overrightarrow{x_i})$) hespala.
Adım 3: Her çözüm yöneyi için aday çözüm yöneyi($\overrightarrow{u_i}$) oluştur ve uygunluk değerlerini($C(\overrightarrow{u_i})$) hespala.
Adım 4: Her bir çözüm yöneyini, karşılık gelen aday çözüm yöneyi ile kıyaslayarak daha uygun olanı seç.
Adım 5: Durdurma kriteri sağlanmadı ise adım 3'e git.

Algoritmanın 3. adımında yapılan aday çözüm yöneyi oluşturma işleminde rastgele seçilmiş birbirinden ve ilgili yöneyden(i) farklı üç yöney(r_1, r_2, r_3) kullanılmaktadır. Ayrıca oluşturulan aday yöney üzerindeki en azından bir parametrenin farklı olmasını garanti altına almak için rastgele bir yöney parametresi(j_{rand}) seçilmektedir. Yöneyler üzerindeki parametrelerin değiştirilmesi amacı ile her bir parametre için bir rastgele sayı üretilmekte ve bu sayı CR kontrol parametresi ile kıyaslanarak ilgili parametrenin ilgili yöneyden(i) mi alınacağını yoksa değişinim ve tekrar birleşim ile mi oluşturulacağını belirlemektedir. Eğer ilgili parametrenin değişinim ve tekrar birleşim ile oluşturulması kararı verilir ise rastgele seçilmiş olan yöneyler devreye girmektedir. İlgili parametre için rastgele seçilmiş olan üç yöneyden ikisinin farkı alınarak bir ölçekleme faktörü ile çarpılmakta ve üçüncü yöneye eklenmektedir. Değişinim ve tekrar birleşim işlemi Eşitlik (3.14) ile ifade edilmektedir.

$$u_{j,\,i}=\begin{cases} x_{j,\,r_3}+F\left(x_{j,\,r_1}-x_{j,\,r_2}\right),\ \text{Eğer}\left(\text{rand}_j[0,\,1]\leq CR \text{ veya } j=j_{rand}\right) \\ \qquad x_{j,\,i},\ \text{Diğer durumlarda} \end{cases} \tag{3.14}$$

Algoritmanın 4. adımında aday çözüm ile mevcut çözüm kıyaslanmakta, daha uygun olan çözümün bir sonraki yinelemede kullanılması amacıyla güncelleme yapılmaktadır.

4. METASEZGİSEL YÖNTEM PARAMETRELERİNİN BELİRLENMESİNE YÖNELİK ALGORİTMALAR

Metasezgisel yöntemlerin hızlı çözüm üretmelerinin yanı sıra makul kalitede çözümler üretmeleri de beklenmektedir. Metasezgisel yöntemlerde kullanıcı tarafından değiştirilebilen ve çözüm kalitesini etkileyen algoritma parametreleri bulunmaktadır. Algoritmaya özgü bu parametrelerden başlıcaları tabu arama algoritması için tabu listesi uzunluğu, tavlama benzetimi algoritması için başlangıç sıcaklığı, popülasyon temelli algoritmaların tamamı için popülasyon büyüklüğü, genetik algoritmalar için çaprazlama oranı, mutasyon oranı, çaprazlama yöntemi, mutasyon yöntemi, karınca algoritmasında buharlaşma oranıdır.

Birçok durumda algoritmaların parametreleri bir defa belirlenmekte ve çalışma zamanında belirlenmiş olan parametrelere müdahale edilememektedir. Bu durum, çözülen problem büyüklüğüne göre farklı parametre değerlerinin belirlenmesi gereksinimini ortaya çıkarmakta ancak parametre değerleri metasezgisel algoritma çalışması esnasında sabit olacağından dolayı parametre değerleri küçük boyutlu problemler için belirlenerek büyük boyutlu problemlerde de aynı parametreler kullanılmaktadır. Parametre değerlerinin belirlenmesi için deneme yanılma, kullanılan metasezgisel algoritmaya benzer diğer algoritmalarda kullanılan parametre değerlerini kullanma, deney tasarımı gibi yöntemler kullanılmaktadır. Bu parametrelerin ne temsil ettiklerinden bağımsız olarak belirlenmeleri ile ilgili algoritmik yöntemler de önerilmiştir. Bu yöntemlerden başlıcaları F-Race (Birattari ve diğerleri, 2002) algoritması ve REVAC (Nannen ve Eiben, 2007) algoritmasıdır. F-Race algoritması hem sürekli hem de kesikli parametre değerlerine uygulanabilirken REVAC algoritması sadece sürekli parametre değerlerine uygulanabilmektedir. Ayrıca algoritmalar, problem kümesini bütün olarak ele alarak küçük boyutlu problem ve büyük boyutlu problem ayrımını ortadan kaldırmaya çalışmaktadırlar.

4.1. F-Race Algoritması

F-Race algoritması kullanılan metasezgisel yöntemin parametrelerin belirlenmesi amacı ile istatistiksel hipotez testleri kullanan bir algoritmadır. Algoritma hem sürekli hem kesikli parametre değerleri ile çalışabilmekte ancak sürekli parametrelerin örnekleme ile kesikli formda ifade edilmesine ve çözülecek olan problemler kümesine ihtiyaç duymaktadır.

Aday parametreler kümesi Θ, çözülecek problemler kümesi I, çözülecek problemlerin oluşma olasılıkları $P_I(i)$, bir i probleminin çözülmesi için kullanılabilecek süre $t(i)$, i probleminin θ parametre yöneyi ile $t(i)$ sürede çözümü ile elde edilen en uygun maliyet değeri $c(\theta, i)$ ile ifade edilsin. $c(\theta, i)$ fonksiyonunun $t(i)$ bağımlılığının dahili olduğu varsayılmaktadır. $C(\theta|\Theta, I, P_I, P_C, t)$ kriter fonksiyonunun en küçük yapacak olan en uygun θ^* parametre yöneyinin bulunması nihai problemdir. Kriter fonksiyonu olarak F-Race algoritması maliyet fonksiyonunun beklenen değerini ele almaktadır. F-Race algoritmasının en küçük yapmaya çalıştığı fonksiyon Eşitlik (4.1) ile verilmektedir.

$$C(\theta|\Theta, I, P_I, P_C, t) = E_{I,C}[c(\theta, i)] \tag{4.1}$$

Ayrıca bulunması amaçlanan θ^* değeri de Eşitlik (4.2) ile verilmektedir.

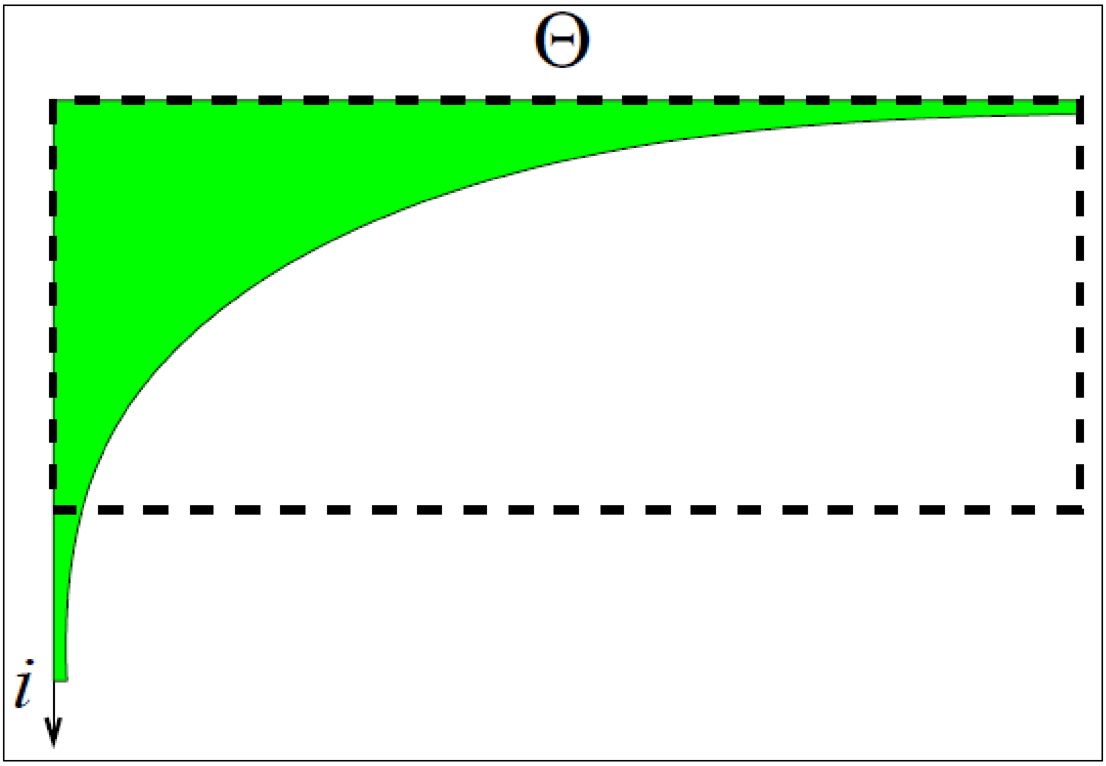
$$\theta^* = \operatorname{argmin}_\theta C(\theta|\Theta, I, P_I, P_C, t) \tag{4.2}$$

F-Race algoritması, problemlerin oluşma olasılıklarına göre rastgele olarak seçilen bir problemi metasezgisel yöntem parametresi olarak Θ kümesinde bulunan bütün θ yöneylerini ayrı ayrı kullanarak çözmektedir. Çözüm sonuçlarını sıralama yaparak oluşturduğu sıralama dizeyine kaydetmektedir. Sıralama dizeyinde bir farklılığın bulunup bulunmadığını veya her bloğun eşit olup olmadığını araştırmak için Friedman test istatistiğini kullanmaktadır. Eğer test istatistiği bloklar arasında farklılık olmadığına dair hipotezi ret ediyorsa en azından bir aday parametre yöneyinin diğer yöneylerden daha uygun bir çözüm üretmesi gerekmektedir. İkinci bir test ile en uygun değeri veren parametre yöneyinin diğer yöneyler ile kıyaslanması Student's t testi ile ikili kıyaslamalar biçiminde yapılmaktadır. Elde edilen sonuçlara göre parametreler arasından istatistiksel olarak elenebilecek olduğu belirlenen parametre yöneyleri parametre yöneyleri kümesinden çıkartılır ve tek bir parametre yöneyi kalana kadar veya çalışma süresi dolana kadar yeni bir problem belirlenerek devam eder.

F-Race algoritmasının temel adımları şu şekilde verilmektedir:

Adım 1: Algoritmanın çalışacağı süre, parametre kümesi Θ, problemler kümesi I, problem seçilme olasılıkları $P_I(i)$, problem çözüm süreleri $t(i)$ ile ilgili gerekli bilgileri al.
Adım 2: Rastgele bir i problemi seç. Parametre kümesi Θ'da bulunan bütün θ yöneyleri ile problemi $t(i)$ süresinde çöz.
Adım 3: Çözüm değerlerini sıralayarak R dizeyine ekle.
Adım 4: R dizeyinde farklı bloklar olup olmadığını Friedman testi ile test et. Eğer farklılık bulunamadı veya algoritma çalışma süresi bitmedi ise Adım 2'ye git.
Adım 5: En küçük sıralama değerlerine sahip θ yöneyinin R dizeyindeki karşılığını bul ve R dizeyindeki diğer bütün yöneyler ile Student's t testine tabi tut. Test sonucuna göre Θ kümesinden silinebilecek θ yöneylerini belirle.
Adım 6: Silinebilecek θ yöneylerine göre Θ kümesini ve R dizeyini güncelle.
Adım 7: Birden fazla θ yöneyi kaldı ve algoritma çalışma süresi bitmedi ise Adım 2'ye git.
Adım 8: En küçük sıralama değerlerine sahip θ yöneyini bul ve bitir.

Algoritma önerildiği hali ile deney tasarımının birçok dezavantajını ortadan kaldırmakta, tüm parametre yöneylerinden istatistiksel olarak anlamlı fark bulması durumunda parametre yöneylerini eleyerek test edilmesi gereken yöneyleri azaltmaktadır. Bu durum Şekil 4.1 ile gösterilmektedir (Birattari ve diğerleri, 2002).



Şekil 4.1. Parametre uzayının F-Race ve deney tasarımı ile indirgeme hızı kıyaslaması(Birattari ve diğerleri, 2002)

4.2. REVAC Algoritması

REVAC algoritması, parametre uygunluğunu ölçmek için bilgi teorisini temel almaktadır. Farklı parametre değerleri veya aralıkları için metasezgisel yöntemin başarısını kestirmek yerine parametre değerleri en büyüklenmiş Shannon dağıntısı(entropy) ile oluşturulan C olasılık yoğunluk dağılımından seçilen parametre değerlerinin beklenen başarımını kestirmektedir. Başlangıç X parametre çözüm uzayında tekdüze dağılım ile başlayan algoritma ilerleyen aşamalarda metasezgisel yöntemin beklenen performansını arttıran X çözüm uzayındaki parametre bölgelerine daha yüksek seçim olaslığı vermektedir. Algoritma iki farklı bakış açısı ile incelenebilmektedir. Bu bakış açılarından ilki algoritmanın bir evrimsel yöntem olarak düşünülmesidir. Evrimsel yöntemde yanıt uzayında m adet parametre yöneyi ile çalışıldığı düşünülmektedir. Bir sonraki popülasyona n adet($n < m$) parametre yöneyi ebeveyn olarak seçilerek tekrar birleşim ve değişinim operatörleri uygulanarak tek bir çocuk yöney üretilmektedir. Yöntemde her seferinde sadece bir yöney değiştirilmektedir. Tekrar birleşim operatörü olarak çok ebeveynli çaprazlama operatörü, değişinim operatörü olarak ise iki aşamalı bir değişinim operatörü kullanılmaktadır. İkinci bakış açısında yöntem, önceki olasılık yoğunluk fonksiyonları ile örneklenen parametre değerleri yanıt yüzeyinin kestirimini kullanarak yeni bir dağılım oluşturmaktadır. Her yinelemede daha yüksek yanıt seviyelerine sahip yanıt yüzeyi bölgelerine daha yüksek olasılık değeri verilmektedir. Birinci bakış açısında bulunan iki aşamalı değişinim operatörü yoğunluk fonksiyonunu yumuşatmaktadır. Yumuşatma işlemi çok gürültülü durumlarla çalışmayı ve oluşturulan dağılımın dağıntısını en büyüklemeyi sağlamaktadır (Nannen ve Eiben, 2007).

5. YAYILMA ZAMANLI OPERASYONEL SABİT İŞ ÇİZELGELEME PROBLEMİ

Yayılma zamanlı operasyonel sabit iş çizelgeleme problemleri, sabit iş çizelgeleme problemleri altında incelenmektedir. Sabit iş çizelgeleme problemleri, bir j işinin başlama zamanı r_j, tamamlanma veya teslim zamanı d_j olarak tanımlandığında işin işlenmesi için gerekli p_j süresinin $d_j - r_j$ farkına eşit olduğu durumlarda ortaya çıkmaktadırlar. Dolayısı ile sabit iş çizelgeleme problemlerinde işin hareket edebileceği bir zaman penceresi bulunmamaktadır.

Sabit iş çizelgeleme problemlerinde işlerin işlenmesi için tabi oldukları bir işleme zaman penceresi bulunmadığından dolayı işin yapılıp yapılmayacağına veya bütün işlerin yapılma zorunluluğu olduğu durumlarda ise en az kaç makine gerektiğine karar verilebilmektedir. Hangi işlerin yapılıp yapılmayacağına karar verilmesi problemi operasyonel sabit iş çizelgeleme problemi olarak, en az kaç adet makine ile işlerin tamamının yapılabileceği problemi taktik sabit iş çizelgeleme problemi olarak adlandırılmaktadır.

Makineler ile ilgili kısıtlar düşünülerek problemlere yayılma zamanlı ve çalışma zamanlı makine kısıtları eklenmiştir. Yayılma zamanı, makine üzerinde yapılacak işler arasında en erken hazır olma zamanı ve en son tamamlanma zamanı arasındaki fark olarak tanımlanır. Bir makinede iki zaman arasında boş süreler olabilir, yaygınlık zamanı boş süreleri de kapsar. Çalışma zamanı kısıtlarında ise makinelerin müsaade edilen toplam çalışma süresinden daha fazla çalışamayacağı kabul edilir ve makinenin çalışma süresi, o makineye atanan işlerin toplam işlem süresi olarak ifade edilir. Bir makinenin çalışma zamanı, boş zamanları içermez. Çalışma zamanı kısıtları, farklı makineler için aynı veya farklı olabilir (Kaya ve Engin, 2009).

Literatürde yayılma zamanlı sabit iş çizelgeleme probleminin tanımlanması için yapılan bazı varsayımlar mevcuttur. n adet iş, m adet makine bulunan bir problem olduğu varsayıldığında, mevcut işler içerisinden bir j işinin başlama zamanı r_j, tamamlanma zamanı d_j, işin getireceği kazanç miktarı w_j biçiminde ifade edilmiş olsun. m adet makinenin de her birinin sadece aynı anda bir tek işi işleyebildiği, işleme sırasında işin yarıda bırakılarak başka bir işin yapılmasına geçmenin mümkün olmadığı bir durum tanımlansın. Her bir k makinesi için yayılma zamanı S, $d_i - r_j \leq S$ olacak biçimde her bir i ve j işleri için ifade edilsin. Genel ifadeyi korumak amacı ile tüm r_j, d_j ve w_j değerlerinin artı birer tamsayı olduğu ve her bir j işi için $d_j - r_j \leq S$ koşulunun geçerli olduğu kabul edilsin. İşlerin başlangıç ve bitiş zamanları, artı tamsayılar ile ifade edildiğinden, zamanı kesikli bir küme biçiminde düşünmek mümkün olmaktadır. İşlerin başlangıç ve bitiş zamanlarını içeren A kümesi Eşitlik (5.1) ile tanımlanmaktadır.

$$A = \bigcup_{j=\{1, 2, \ldots, n\}} \{rj, dj\} \tag{5.1}$$

Bu A kümesinin elemanlarının azalmayan(non decreasing) sıra ile sıralanarak, tekrar eden elemanların kümeden çıkartılması ile A kümesi yeniden oluşturulmakta ve Eşitlik (5.2) ile h tanımlanmaktadır.

$$h = |A| - 2 \tag{5.2}$$

Kümenin oluşturulmasındaki amaç, kümenin belirttiği aralıklardan her bir $t \in \{0, 1, ..., h\}$ elemanı için, $]A_t, A_{t+1}[$ zaman aralığında herhangi bir işin başlama veya bitiş durumunun olmamasının sağlanmasıdır. Sonraki adımda n adet iş ve m adet makine için bir j işinin k makinesine atanıp atanmayacağını tanımlayan - j işinin k makinesine atanması durumunda 1, j işinin k makinesine atanmaması durumunda ise 0 değerini alan - bir $x_{jk} \in \{0, 1\}$ değişkeni tanımlanmaktadır.

Değişkenlerin ve zamanları ifade eden A kümesinin yanı sıra, işleri ve aralarındaki ilişkileri tanımlayan kümeler de belirlenmelidir. J_t kümesi her $t \in \{0, 1, ..., h\}$ olmak kaydı ile $]A_t, A_{t+1}[$ zaman aralığında işlenebilecek bütün işlerin bir kümesini, I_j kümesi de her $j \in \{1, 2, ..., n\}$ olmak kaydı ile birbirleri ile uyumsuz bütün işlerin bir kümesini tanımlamaktadır. İşlerin uyumsuzluğu ile iki durumda karşılaşılmaktadır. Bu durumlar:

- İki iş(i ve j) birbiri ile örtüşüyor ise, yani ($r_j \leq r_i$ ve $d_j > r_i$) veya ($r_i \leq r_j$ ve $d_i > r_j$) durumunda,

- Yayılma zamanı kısıtı ihlal edildiğinde, yani $d_i - r_j > S$ veya $d_j - r_i > S$ durumunda,

biçiminde ifade edilmektedir. Bu tanımlamalar doğrultusunda 0-1 tamsayılı doğrusal programlama modeli Sistem (5.3) ile verilmektedir:

$$\text{En Büyükle}_x \sum_{j=1}^{n}\sum_{k=1}^{m} w_j\, x_{jk} \tag{5.3a}$$

Aşağıdaki Koşullar Altında:

$$\sum_{k=1}^{m} x_{jk} \leq 1 \qquad \forall j \in \{1, 2, \ldots, n\} \tag{5.3b}$$

$$\sum_{j \in J_t} x_{jk} \leq 1 \qquad \forall k \in \{1, 2, \ldots, m\},\ \forall t \in \{1, 2, \ldots, h\} \tag{5.3c}$$

$$x_{ik} + x_{jk} \leq 1 \qquad \forall k \in \{1, 2, \ldots, m\},\ \forall j \in \{1, 2, \ldots, n\},\ \forall i \in I_j \tag{5.3d}$$

$$x_{jk} \in \{0, 1\} \qquad \forall k \in \{1, 2, \ldots, m\},\ \forall j \in \{1, 2, \ldots, n\} \tag{5.3e}$$

Modelin amaç fonksiyonu Formül (5.3a) ile verilmekte olup seçilen işlerden elde edilecek gelir miktarının en büyük olmasını amaçlamaktadır. Eşitsizlik (5.3b), her bir işin en fazla bir makineye atanması gerektiğini belirtmektedir. Eşitsizlik (5.3c), her bir makinenin aynı anda sadece bir iş üzerinde işlem yapabileceğini belirtmektedir. Eşitsizlik (5.3d), işlerin uyumsuz olduğu işler ile birlikte aynı makineye atanamayacağını belirtmektedir. Formül (5.3e), karar değişkenlerinin tamamının 0 veya 1 değerlerinden birisini alması gerektiğini belirtmektedir (Eliiyi ve Azizoğlu, 2006; Rossi ve diğerleri, 2010).

Problemin NP-zor sınıfında olduğu,. problemin bazı durumlarının polinom zamanda çözülebildiği bilinmektedir. Problemin polinom zamanda çözülebildiği durumlar; tek bir makinenin bulunduğu problem, işlerin parçalı işlenmesi ve makinelerde öne alımlara izin verilen problem ve tüm makinelerin ortak bir zamanda çalışmaya başladığı problemdir (Eliiyi ve Azizoğlu, 2006). Polinom zamanda çözülebilen çeşitlenmeleri(variant) haricinde problemin çözümü için tamsayılı programlamadan daha etkili yöntemlere ihtiyaç duyulmuştur. Yayılma zamanı kısıtlı operasyonel sabit iş çizelgeleme problemi için sezgisel ve metasezgisel yöntemler (Rossi ve diğerleri, 2010; Eliiyi ve Azizoğlu, 2011), dal-sınır algoritması (Eliiyi ve Azizoğlu, 2006) ve dal-değerleme algoritması (Solyalı ve Özpeynirci, 2009) önerilmiştir.

En uygun çözümde bulunması gereken bazı özellikler olduğu bilinmektedir. Bu özellikler kısaca şunlardır:

- Eğer en uygun çözümde $w_j > w_i$, $r_i \leq r_j$, $d_i \geq d_j$ koşullarını sağlayan bir i işi herhangi bir makinede işlenmekte ise j işi, i işine göre baskındır ve j işi de çözümde olmalıdır.
- Bir i işini basılgın yapan işlerin kümesi D_i olsun. Eğer $|D_i| \geq m$ ise i işi en uygun çözümde bulunamaz.
- En uygun çözümde eğer i işi işlenmiş ve $r_i \leq r_j$, $d_i \geq d_j$ koşullarını sağlayan tüm j işlerinin getirebileceği en büyük getiri için $\sum_{j \in LP_i} w_j > w_i$ ise LP_i kümesindeki en az bir iş çözümde olmalıdır.
- $r_i \leq r_j < d_i < d_j$ ve $B=\{l | d_i \leq r_l < d_j\}$ olsun. Eğer en uygun çözümde i işi herhangi bir makinede ilk iş olarak işeniyorsa ve $\sum_{l \in B} w_l + w_i < w_j$ ise çözümde j işi de olmalıdır.
- $r_j < r_i$ ve $C=\{l | r_j + S < d_l \leq r_i + S\}$ olsun. Eğer en uygun çözümde $r_i < d_j$ olan bir i işi herhangi bir makinede ilk iş olarak işleniyorsa ve $\sum_{l \in C} w_l + w_i < w_j$ ise j işi de çözümde olmalıdır. $r_j < r_i$ ve $C=\{l | r_j + S < d_l \leq r_i + S\}$ olsun. Eğer en uygun çözümde $r_i \geq d_j$ olan bir i işi herhangi bir makinede ilk iş olarak işleniyorsa ve $\sum_{l \in C} w_l < w_j$ ise j işi de çözümde olmalıdır. Eğer bu iki koşulu sağlayan işlerin sayısı makine sayısından fazla ise i işi en uygun çözümde ilk iş olarak bulunamaz.
- ST_k, k makinesinde ilk işin başlangıç zamanı olsun. Eğer i işi k makinesinde işleniyorsa ve $ST_k \leq r_j \leq r_i$, $d_j \leq r_i$ yanı sıra $[r_j, d_j]$ aralığında k makinesinde hiçbir iş işlenmemişse en uygun çözümde j işi olmalıdır.
- Eğer bir j işi l adetten daha az sayıda iş ile örtüşüyorsa ve l adet makine için $ST_k \leq r_j \leq d_j \leq ST_k + S$ ise en uygun çözümde j işi olmalıdır.
- $ST_k < r_i < r_j < d_i < d_j < ST_k + S$ olması durumunda; eğer $w_i > w_j$ ve $[r_i, r_j]$ aralığında hiçbir iş bulunmamakta ise j işinin çözümde olması durumunda çözümde i işi de

olmalıdır, eğer $w_j > w_i$ ve $[\mathrm{d_i}, \mathrm{d_j}]$ aralığında hiçbir iş bulunmamakta ise i işinin çözümde olması durumunda çözümde j işi de olmalıdır (Eliiyi ve Azizoğlu, 2006).

5.1. Yayılma Zamanlı Operasyonel Sabit İş Çizelgeleme Problemi için Açgözlü Sezgisel Algoritma

Bu bölümde, Rossi ve diğerleri 2010 çalışmasında önerilen açgözlü sezgisel algoritmanın temelleri verilmekte, ilerleyen bölümlerde genelleştirilmiş problem için açgözlü sezgisel algoritmada yapılan değişiklikler aktarılmaktadır.

Yayılma zamanlı operasyonel sabit iş çizelgeleme için açgözlü sezgisel algoritma, yinelemeli olarak makinelere atanmış olan işlerin toplam kazancını en büyüklemeye çalışırken, bu yinelemeler sırasında atamaları makinelerin yayılma zamanı kısıtlarını ihlal etmeden yapmaktadır. Açgözlü sezgisel algoritma üç adımdan oluşmaktadır. Bu adımlar şu şekilde ifade edilebilir:

1- $\forall t \in \{0, 1, \dots, h\}$ için $[\mathrm{A_t}, \mathrm{A_t} + \mathrm{S}]$ zaman penceresi kazanç üst limiti $\mathrm{g_t}$ olarak tanımlanır. Kazanç hesabı $[\mathrm{A_t}, \mathrm{A_t} + \mathrm{S}]$ zaman penceresinde oluşan bir iş çakışması durumunda, hangi işlerin tercih edileceğinin belirlenmesini ele alır. Zaman penceresinde işlerde çakışma olması durumunda birim zaman diliminde en yüksek getiri miktarına sahip işin o zaman diliminde alındığı hesaplanmakta, çakışma olmadığı durumda birim zaman diliminde alınabilecek işin o zaman diliminde alındığı hesaplanarak kazanç bulunmaktadır. Tüm zaman dilimlerinde çakışma olmaması durumunda alınabilecek işlerin tamamı bitirilebileceğinden dolayı zaman penceresinde yapılabilecek işlere ait getiriler toplamı hesaplanır. Basit bir ifade ile kazanç, işlerin parçalı olarak işlenebildiği durumda tek bir makinenin yayılma zamanı içinde elde edebileceği en büyük iş getirileri toplamıdır. Tüm zaman aralıkları için hesaplanan kazançlar artmayan biçimde sıralanır. Artmayan sırada $\mathrm{p_h}$ adet en yüksek kazanç ile bir B kümesi oluşturulur. $\mathrm{p_h}$ parametresi etkinlik için ayarlanmalıdır.

2- B kümesindeki her bir zaman aralığı başlangıcı t için, ataması yapılmamış işler ile $\mathrm{M_t}$ kümeleri oluşturulur. Bu kümenin oluşturulması sırasında $[\mathrm{A_t}, \mathrm{A_t} + \mathrm{S}]$ aralığında işlenebilecek işler ile bir ağ yapısı oluşturulur. Başlangıç ve bitiş düğümleri dışındaki her bir düğüm ile bir iş temsil edilir. Birbiri arkasından işlenebilecek işler arasında yönlü ark oluşturulur ve bu arkların değerleri, işlerin getirileri olarak belirlenir. Oluşturulan bu ağ modeli en uzun yol problemi biçiminde modellenerek çözüldüğünde hangi işlerin $[A_t, A_t + S]$ aralığında işlenmesi durumunda w_j toplamının en büyük olacağı tek bir makine için hesaplanmış olunur. M_t kümesi bu işleri içermektedir.

3- B kümesindeki her bir zaman aralığı başlangıcı için hesaplanan kümelerden, en büyük w_j toplamına sahip olan M_t kümesindeki işler, o anda mevcut olan makineye atanır. Atanması yapılmamış makine var ise ve atanacak işler bitmemiş ise birinci adıma dönülür, bütün makinelere atama yapılmış ise algoritma bitirilir (Rossi ve diğerleri, 2010).

Algoritmanın temelinde tek makine probleminin en uzun yol problemi biçiminde modellenerek polinom zamanda çözülebilmesi fikri bulunmaktadır. Tek makine için en uzun yol problemi, ağ çevrimsiz olduğundan dolayı $O(n^2)$ karmaşıklık ile çözülebilmektedir (Eliiyi ve Azizoğlu, 2006). Aynı amaca yönelik olarak kullanılabilen ve karmaşıklığı $O(n)$ olan bir dinamik programlama algoritması bulunmaktadır (Solyalı ve Özpeynirci, 2009).

5.2. Yayılma Zamanlı Operasyonel Sabit İş Çizelgeleme Problemi için Metasezgisel Algoritma

Bu bölümde, Rossi ve diğerleri 2010 çalışmasında önerilen gruplama genetik metasezgisel algoritmasının temelleri verilmekte, ilerleyen bölümlerde genelleştirilmiş problem için metasezgisel algoritmada yapılan değişiklikler aktarılmaktadır.

Rossi ve diğerleri tarafından önerilen algoritma adımları şu şekildedir:

- Başlangıç çözümlerin oluşturulması
- Seçim
- Çaprazlama veya değişinim
- Yerel arama
- Değerlendirme
- Değiştirme

Yöntemde makinelerin yayılma zamanlarının ve makinelerdeki getirilerinin eşit olduğu varsayılmakta, dolayısı ile algoritmada makinelerin birbiri yerine geçebiliyor olması durumu düşünülmektedir. Algoritmanın kromozom yapısında, m makine n iş içeren bir problem için makinelerin birbiri yerine geçebiliyor olmasından dolayı m adet makine çizelgesi tutulmaktadır. Makinelere atanmış olan işlerin getirileri toplamı uygunluk fonksiyonu olarak kullanılmaktadır.

Algoritmanın ilk adımı başlangıç çözümlerinin oluşturulmasıdır. Başlangıç çözümlerin oluşturulmasında iki farklı yöntemden her biri eşit olasılığa sahip olacak biçimde kullanılmaktadır. İlk yöntem, sırası ile bir makinenin çizelgesini oluşturduktan sonra bir diğer makinenin çizelgesini oluşturmaktadır. Bir makinenin çizelgesini oluşturmak için çizelgelenmemiş işlerden rastgele bir tanesi seçilmekte ve makineye atanmakta, sonrasında ise henüz makinelere atanmamış işlerden atanmış işlere en yakın iş seçilerek devam edilmektedir. Birden çok sayıda işin atanmış işlere en yakın olması durumunda ise kararsızlığı gidermek için işlerden hangisi daha yüksek kazanç/işlem süresi oranına sahip ise o iş seçilerek makine çizelgesine atanmaktadır. Her işin popülasyonda yer alabilmesi için en erken başlama ve en geç teslim zamanı olan işlerin ilk iş olarak seçildiği iki ayrı çözüm oluşturulmaktadır. İkinci yöntem her makineye rastgele bir iş atayarak başlamaktadır. Bir p_{hgt} olasılığına göre en yüksek kazanç/işlem süresi oranına sahip iş ya da tamamen rastgele bir iş seçilmektedir. Seçilen iş, atama sonrasında en az boş kalacak olan makineye atanmaktadır. Yeni iş seçimine, makineye atanabilecek iş kalmayana kadar devam edilmektedir.

Algoritmada çaprazlamaya girecek bireylerin seçimi için olasılıksal ikili turnuva seçim yöntemi kullanılmaktadır. Daha iyi olan aday birey p_{better} olasılık ile seçilmektedir. Değişinim operatörü için seçim işleminde, daha iyi aday çözüm seçilmekte, daha kötü olan bireye şans verilmemektedir. Bir ebeveyne, çaprazlama veya değişinim operatörlerinden sadece birisi uygulanmaktadır. Rastgele sayı üretilerek p_c çaprazlama olasılığına göre çaprazlamaya girip girilmeyeceğine karar verilmektedir.

Algoritmada kullanılan çaprazlama operatörü yinelemeli olarak çalışmaktadır. Seçilmiş olan iki ebeveyn bireyden birisini rastgele seçerek, seçilen bireyin en yüksek getiriye sahip makine çizelgesi doğrudan çocuk bireye aktarılmaktadır. İlk aktarımdan sonra çocuk bireyin çizelgesinde bazı işler olduğundan dolayı çocuk bireye aktarılmış olan işler her iki ebeveynden de kaldırılarak makine getirileri güncellenmektedir. Çocuk bireyde mevcut makinelerin tamamı doldurulması için tüm makineler için bu işlemler tekrar etmektedir.

Değişinim operatörü olarak iki farklı operatör kullanılmaktadır. İlk değişinim operatörü en düşük kazanca sahip makine çizelgesini ve rastgele bir makine çizelgesini çözümden kaldırmaktadır. Çizelgeye alınmamış iş sayısı arttığından dolayı ikinci aşamada boş kalan makine çizelgelerini oluşturmak için sezgisel yerel arama algoritması kullanmaktadır. İkinci değişinim operatörü ise makinelere atanmış olan işleri rastgele bir sayı üreterek belirli bir p_{rem} olasılığına göre çizelgeden çıkartmaktadır. Yine aynı biçimde çizelgeye alınmamış iş sayısı artmakta ve sezgisel yerel arama algoritması çalıştırılmaktadır.

Sezgisel yerel arama; algoritmanın temel noktasını oluşturmakta ve hem çaprazlama operatörü hem de değişinim operatörü sonrasında uygulanmaktadır. Sezgisel yerel arama yinelemeli olarak çalışmakta, her yinelemede belirli kriterlere göre atanmamış bir işi seçmekte ve seçilen işi makinelerden birisine atamaya çalışmaktadır. İş seçimi, atanmamış işlerden ya rastgele ya da en yüksek kazanç/işlem süresi oranına sahip iş seçilerek yapılmaktadır. Hangi iş seçimi yöntemin kullanılacağı hakkında karar vermek için p_s parametresi kullanılmaktadır. Seçilmiş olan iş, tüm olası makinelerde boşluklara atanmaya çalışılmakta, birden fazla makineye atanabilecek ise atama sonunda en az boşluk kalacak makineye atama yapılmaktadır. Eğer seçilen iş, hiçbir makinede mevcut boşluklara alınamıyor ise, seçilen işin makinelere atanabilmesi için bazı işlerin makinelerden çıkartılması gerekmektedir. Her makine için makineden çıkartılması gereken işler, makinede mevcut olup ele alınan iş ile uyumsuz olan işlerdir. Çıkartılması için belirlenen bu işlerin makineye alınmaması durumunda makinedeki getirilerinden vazgeçilmekte, ele alınan işin getirisi kazanılmaktadır. Eğer makinedeki bazı işleri çıkartarak seçilen işi atama hareketi getiri ile sonuçlanıyor ise bu atama hareketi yapılmaktadır. Aksi durumda makineden çıkartılacak işlerin getirileri toplamı, ele alınan işin getirisinden büyük olduğunda ele alınan işi makineye almamak daha iyi olmaktadır. Seçim yöntemine uygun olarak seçilebilecek tüm işler denenmesine rağmen hiçbir iş iyileştirme sağlamamakta ise sezgisel yerel arama sonlandırılmaktadır. Eğer bir işin herhangi bir makineye alınması getiri artışı ile sonuçlanmakta ise, iş makineye atanmakta, bu iş ile uyumsuz olan işler ise diğer makinelere atanabilecek olduğundan çizelgelenmemiş işler listesi güncellenmekte ve sezgisel yerel arama süreci tekrar başlatılmaktadır.

Yeni oluşturulan çocuk birey değerlendirilerek, aynı bireyden popülasyonda bulunmaması durumunda, mevcut popülasyondaki daha kötü uygunluk değerine sahip bir bireyin yerine geçirilmektedir.

Algoritmanın etkinliği, küçük problemlerde tamsayılı programlama modeli çözümünden elde edilen en uygun çözüm değeri ile kıyaslanarak, büyük problemlerde açgözlü sezgisel algoritma ile elde edilen çözüm değerinin ne kadar iyileştirdiği ile kıyaslanarak gösterilmiştir (Rossi ve diğerleri, 2010).

5.3. Yayılma Zamanlı Operasyonel Sabit İş Çizelgeleme Problemi için Önerilen Yarı Tanımlı Programlama Gevşetmesi

Yarı tanımlı programlama, doğrusal programlamanın genelleştirilmiş biçimidir. En büyükleme türünde amaç fonksiyonuna sahip tamsayılı doğrusal programlama modellerinde üst sınır hesaplamada sıklıkla modelin doğrusal programlama gevşetmesi kullanılmaktadır. En uygun çözüm değeri, üst sınır değeri ile herhangi bir olurlu tamsayı çözümden elde edilen alt sınır değeri arasında kalmaktadır. Bu durum Eşitsizlik (5.4) ile ifade edilmektedir.

$$z_{\text{Alt Sınır}} \leq z^{*} \leq z_{\text{Üst Sınır}} \tag{5.4}$$

Üst sınır değeri hesaplamak için kullanılabilecek bir diğer yöntem ise yarı tanımlı programlama gevşetmesidir. Bu gevşetme yöntemi doğrusal programlama gevşetmesini de içerdiğinden daha katı bir üst sınır değeri vermektedir. Bazı problemler için en kötü durumda yarı tanımlı programlama gevşetmesinin en uygun çözüm değerinden belirli bir sabit uzaklıkta olduğu gösterilmiştir (Goemans ve Williamson, 1995).

Yarı tanımlı programlama gevşetmesi ile ikinci dereceden değişkenleri gevşetme modeline ekleyebilmek mümkün olmaktadır. Bu özellik ikinci dereceden atama problemi gibi problemler için oldukça önemlidir. Yayılma zamanlı operasyonel sabit iş çizelgeleme probleminde ise birlikte alınamayacak olan i ve j işleri için x_i ve x_j değişkenlerin çarpımının 0 olması gerekliliği önem taşımaktadır. Yarı tanımlı programlama modelinde bu çarpım doğrusallaştırılarak yeni bir değişken ile ifade edilmekte ve bu değişkenin 0 olması koşulu modele eklenmektedir. Ayrıca doğrusal tamsayılı modeldeki bütün doğrusal kısıtlar birer x değişkeni ile çarpılarak yeni kısıtlar elde edilmektedir.

Modelde x yöneyi, k makinesinde işlenip işlenmeyeceğine karar verilecek olan j işini temsil eden x_{jk} karar değişkenlerini sıralı biçimde tutmaktadır. Bu durum m makine n iş için Eşitlik (5.5) ile gösterilmektedir.

$$x = [x_{11}\ x_{12} \ldots x_{1n}\ x_{21}\ x_{22} \ldots x_{2n} \ldots x_{m1}\ x_{m2} \ldots x_{mn}]^T \tag{5.5}$$

Yarı tanımlı dizey oluşturulduğunda, aynı k makinesinde birlikte işlenemeyecek i ve j işleri için $x_{ik}x_{jk}$ değerinin 0 olması gerekmektedir. Benzer biçimde bir j işi için iki farklı k ve l makinelerinde işlenme durumunu ifade eden $x_{jk}x_{jl}$ değerinin de 0 olması gerekmektedir. Bu durumun sonucunda, değişken dizeyinin bir bölümü blok olarak sıfırlardan oluşan dizeyler haline gelmektedir.

İki makine üç işlik bir problemde ilk işin ikinci iş ile aynı zaman içerisinde işlenmesi gerektiği($r_1 < r_2$ ve $d_1 > r_2$), birinci iş ile üçüncü iş arasında ise yayılma zamanı kısıtının($r_1 < r_3$ ve $r_1 + S < d_3$) ihlal edildiği problem için örnek bir dizey Eşitlik (5.6) ile verilmektedir.

$$X = \begin{bmatrix} 1 & & & & x^T & & \\ & x_{11}^2 & 0 & 0 & 0 & 0 & 0 \\ & 0 & x_{12}^2 & 0 & 0 & 0 & 0 \\ x & 0 & 0 & x_{21}^2 & 0 & x_{21}x_{31} & x_{21}x_{32} \\ & 0 & 0 & 0 & x_{22}^2 & x_{22}x_{31} & x_{22}x_{32} \\ & 0 & 0 & x_{21}x_{31} & x_{22}x_{31} & x_{31}^2 & 0 \\ & 0 & 0 & x_{21}x_{32} & x_{22}x_{32} & 0 & x_{32}^2 \end{bmatrix} \tag{5.6}$$

Örnek dizeyin elemanlarından 0 değeri almak zorunda olmayan $x_{21}x_{31}$ gibi iki değişkenin çarpımı ifadeleri yeni y değişkenleri ile ifade edilerek doğrusallaştırma yapıldığında ortaya çıkan örnek dizey Eşitlik (5.7) ile verilmektedir.

$$
Y = \begin{bmatrix}
1 & & & & \mathrm{x}^T & & \\
 & \mathrm{x}_{11}^2 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & \mathrm{x}_{12}^2 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & \mathrm{x}_{21}^2 & 0 & \mathrm{y}_{21\,31} & \mathrm{y}_{21\,32} \\
\mathrm{x} & 0 & 0 & 0 & \mathrm{x}_{22}^2 & \mathrm{y}_{22\,31} & \mathrm{y}_{22\,32} \\
 & 0 & 0 & \mathrm{y}_{21\,31} & \mathrm{y}_{22\,31} & \mathrm{x}_{31}^2 & 0 \\
 & 0 & 0 & \mathrm{y}_{21\,32} & \mathrm{y}_{22\,32} & 0 & \mathrm{x}_{32}^2
\end{bmatrix} \tag{5.7}
$$

Dizeyde çok fazla sayıda 0 değeri bulunmakta ve dizey ayrıklık seviyesi artmaktadır. Birçok sayısal hesaplama yönteminde özel bir durum olan ayrık veriler için hesaplama gereksinimi bulunmadığından, bu elemanlar işlem yüküne sebep olmamaktadırlar. Birbiri ile uyumsuz olan işlerin artışı ile dizeydeki ayrıklık daha da artmaktadır.

Yarı tanımı programlama, doğrusal programlama gibi dışbükey programlamanın özel bir alt dalı olduğundan dolayı yarı tanımlı programlama modellerini polinom zaman karmaşıklığı ile çözebilen algoritmalar mevcuttur.

6. YAYILMA ZAMANLI GENELLEŞTİRİLMİŞ OPERASYONEL SABİT İŞ ÇİZELGELEME PROBLEMİ

Literatürde bulunan yayılma zamanlı operasyonel sabit iş çizelgeleme probleminin tanımlanmasında, gerçek hayata uygun olmayan bazı kısıtları bulunmaktadır. Bu bölümde kısıtların bir bölümünün kaldırılması ile elde edilen problem tanımlanmaktadır. Tanımlanan probleme ait tamsayılı programlama modeli verilmekte, tanımlanan problem için bir hibrit metasezgisel algoritma önerilmektedir.

Gerçek hayatta birçok durumda makinelerin işleme maliyetleri veya teknolojik hassasiyet dolayısı ile müşteri tercihleri gibi sebeplerden dolayı makine getirileri değişkenlik gösterebilmektedir. Bu durumda makinelerin birbirinin aynısı olma kabulü geçerliliğini kaybetmekte ve iş getirileri olan w_j parametresi her makine için farklı olmaktadır. Yayılma zamanlı operasyonel sabit iş çizelgeleme problemine ait Sistem (5.3) ile verilen modelde getirileri temsil eden w_j parametresine, yeni bir makine indisi eklenerek w_{jk} biçiminde ifade edilmesi ile farklı makine getirileri genelleştirilmiş problem için modellenmektedir.

Makine getirilerinin farklı olması yanı sıra gerçek hayatta benzer biçimde makinelerin yayılma zamanlarını temsil eden S parametresi farklılık gösterebilmektedir. S parametresine yeni bir indis eklenerek S_k biçiminde ifade edilmesi ile her bir makine için farklı yayılma zamanları modellenmektedir. Bu durumda, aynı olan iş uyumsuzlukları kümesi I_j de her makine için farklı olmaktadır. Farklı makinelerin farklı yayılma zamanı kısıtları olabildiğinden, I_j kümesine her makine için yeni bir indis eklenerek I_{jk} biçiminde genelleştirilmiş problem için modellenmektedir.

Gerçek hayatta her makine için bir maliyet söz konusudur ve farklı makineler için farklı maliyetler söz konusu olabilmektedir. Bu maliyet kalemi sabit ve değişken maliyet kalemlerinden oluşmaktadır. Değişken maliyetler, birçok durumda alınacak iş başına geliri etkileyebilecek şekilde modellenebildiğinden w_{jk} parametresinin içinde düşünülmektedir. Ancak sabit maliyet kalemi bu kapsam dışında kalmakta, makinenin çalıştırılarak tek bir iş yapması durumunda bile bir sabit maliyet kalemi oluşmaktadır. Bu maliyet kalemi, gerçek hayatta makinenin kirası gibi düşünülebilir ve her makine için farklılık gösterebilmektedir. Makinenin çalıştırılması durumunda karşılanması gereken bu maliyet c_k parametresi ile modellenmektedir. Ayrıca modele makinenin çalıştırılıp çalıştırılmayacağını belirten bir 0-1 tamsayılı y_k değişkeni eklenmektedir. Makinenin çalışması durumunda y_k değişkeninin 1 değerini, çalışmaması durumunda da 0 değerini alması için model kısıtları ve amaç fonksiyonu genelleştirilmiş problem için modellenmektedir.

Makine yaygınlık zamanlarının farklı olmasının yanı sıra, bir makinenin kiralanması gibi durumlarda kiralama yapılan birimler için maliyetler değişkenlik gösterebilmektedir. Özellikle işletmelerin makine kiralama bedellerini sadece kira süresine göre belirlemediği bilinmektedir. Bu kiralama süresinin artması ile ek süre için birim maliyetin artabildiği durumlar olabildiği gibi - hizmet sektöründe fazla mesai - birim maliyetin azalabildiği durumlar da - üretim sektöründe makine üretim maliyetinin üretim miktarı ile azalması - bulunmaktadır. Bu iki durumda bulunabilecek olan makine türleri için aynı makine, farklı makineler biçiminde modellenmektedir. Bu durum için aynı makinenin birden çok defa kullanılamamasının sağlanması gerektiğinden makineler arası uyumsuzluk kümesi Im_k tanımlanmaktadır. Referans olarak seçilen bir k makinesi için Im_k kümesinde bulunan makinelerde, makine kullanımını temsil eden 0-1 tamsayılı y_k değişkeninin sadece bir tanesinin kullanılması gerekmektedir. Bu durum da bir kısıt olarak genelleştirilmiş problem için modele dahil edilmektedir.

m adet makinenin, n adet işin, alınabilecek işlerin getirilerinin makine bağımlı olduğu, mevcut makinelerin yayılma zamanlarının, kendi aralarında uyumsuzluklarının ve maliyetlerinin bulunduğu, işlerin alınması ile elde edilecek kar miktarının en büyük yapılmaya çalışıldığı ve tüm parametrelerin artı kabul edildiği sabit iş çizelgeleme problemlerine yayılma zamanlı genelleştirilmiş operasyonel sabit iş çizelgeleme problemi adı verilmektedir.

Problemin matematiksel modeli Sistem (6.1) ile ifade edilmektedir:

En büyükle: $\sum_{j=1}^{n}\sum_{k=1}^{m} w_{jk}\, x_{jk} - \sum_{k=1}^{m} c_k\, y_k$ (6.1a)

Aşağıdaki Koşullar Altında:

$\sum_{k=1}^{m} x_{jk} \leq 1 \qquad \forall j \in \{1, 2, \ldots, n\}$ (6.1b)

$\sum_{j \in J_t} x_{jk} \leq y_k \qquad \forall k \in \{1, 2, \ldots, m\}, \forall t \in \{1, 2, \ldots, h\}$ (6.1c)

$x_{ik} + x_{jk} \leq 1 \qquad \forall k \in \{1, 2, \ldots, m\}, \forall j \in \{1, 2, \ldots, n\}, \forall i \in I_{jk}$ (6.1d)

$y_k + y_l \leq 1 \qquad \forall k \in \{1, 2, \ldots, m\}, \forall l \in Im_k$ (6.1e)

$x_{jk} \in \{0, 1\} \qquad \forall k \in \{1, 2, \ldots, m\}, \forall j \in \{1, 2, \ldots, n\}$ (6.1f)

$y_k \in \{0, 1\} \qquad \forall k \in \{1, 2, \ldots, m\}$ (6.1g)

Amaç fonksiyonu Formül (6.1a) ile verilmekte olup j işinin k makinesine atanması durumunda makinelere ataması yapılan işlerin getirilerini toplamakta, k makinesinin çalıştırıldığı durumlarda çalıştırılan makinelerin maliyetlerinin toplamını çıkartmakta ve böylece elde edilecek olan net geliri hesaplamaktadır. Nihai hedef net gelir değerinin en büyük yapılmasıdır.

Eşitsizlik (6.1b), her bir işin en fazla bir makineye atanması gerektiğini ifade etmektedir. Eğer j işi bir k makinesine atandı ise başka bir l makinesine alınamamalıdır. Bir j işi herhangi bir makineye atandı ise eşitsizliğin sol tarafı 1 olmakta ve eşitlik sağlanmaktadır. Diğer durumda eğer j işi hiçbir makineye atanmadı ise iş yapılmayacak anlamına gelmektedir ve eşitsizlik sol taraf değeri 0 olarak sağlanmaktadır.

Eşitsizlik (6.1c), bir k makinesinin aynı anda sadece tek bir j işini işleyebileceği ifade etmektedir. Ayrıca sağ taraf değişkeni olan y_k, k makinesinin kullanılıp kullanılmadığını gösteren değişkedir. Bu kısıt amaç fonksiyonu ile birlikte düşünüldüğünde y_k değişkeni sadece makinenin kullanılması durumunda 1 değerini almaktadır. Makinenin kullanılmadığı durumlarda değişkenin 1 değerini alması amaç fonksiyonunda bir azalmaya yol açtığı için değişkenin, amaç fonksiyonu tarafından alabileceği en düşük değer olan 0 değerine doğru zorlanacaktır.

Eşitsizlik (6.1d), bir j işinin k makinesine atanması durumunda aynı makinede işlenemeyecek olan bir i işinin k makinesine atanamayacağını, i ve j işlerinden en fazla bir tanesinin k makinesine atanabileceğini ifade etmektedir. Bir k makinesine işlenemeyecek olan i ve j işlerinin ikisinin birden atanması durumunda sol taraf değeri 2 olacaktır ve koşul sağlanmayacaktır. I_{jk} iş uyumsuzluk kümesi k makinesi için j işi ile birlikte işlenemeyecek işleri içermektedir. Yayılma zamanları farklı olan makineler için kümeler de farklı olmakta ve yayılma zamanına göre kümeler belirlenmektedir.

Eşitsizlik (6.1e), k ve l makinelerinden en fazla bir tanesinin çalıştırılabileceğini ifade etmektedir. Birlikte çalıştırılamayacak iki makinenin çalıştırılması durumunda sol taraf değeri 2 olacaktır ve koşul sağlanmayacaktır. Im_k makine uyumsuzluk kümesi k makinesi ile birlikte çalıştırılamayacak olan makineleri içermektedir. Farklı durumlar için farklı makine uyumsuzluk kümeleri belirlenmelidir.

Formül (6.1f) ve Formül (6.1g), x_{jk} ve y_k değişkenlerinin ya 0 ya da 1 değeri alması gerektiğini belirtmektedir.

Yayılma zamanlı genelleştirilmiş operasyonel sabit iş çizelgeleme probleminin özel bir durumu yayılma zamanlı operasyonel sabit iş çizelgeleme problemidir. Problemin çözümü için matematiksel modellerin yetersiz kaldığı durumlarda sezgisel ve yapay zeka temelli tekniklerin kullanıldığı bilinmektedir. Yayılma zamanlı operasyonel sabit iş çizelgeleme problemi için oldukça hızlı çözüm elde etmek için önerilen açgözlü sezgisel algoritma ve büyük problemler için önerilen hibrit metasezgisel algoritma alt bölümlerde açıklanmaktadır.

6.1. Yayılma Zamanlı Genelleştirilmiş Operasyonel Sabit İş Çizelgeleme Problemi için Önerilen Açgözlü Sezgisel Algoritma

Önerilen açgözlü sezgisel algoritma, yinelemeli olarak elde edilecek olan net karı, makinelere atanmış olan işlerin toplam getirisinden makine maliyetlerini çıkartarak hesaplamakta ve en büyüklemeye çalışmaktadır. Algoritma yinelemeler sırasında, atamaları makinelerin yayılma zamanı kısıtlarını ihlal etmeden yapmaktadır.

Önerilen açgözlü sezgisel algoritma dört adımdan oluşmaktadır. Bu adımlar şu şekilde ifade edilebilir:

1- Ataması yapılacak makine seçimi için bir sıralama belirle. Bu sıralama rastgele olabileceği gibi makinelere ait en düşük maliyet/yayılma zamanı oranına göre de olabilmektedir. Makine sıralamasından seçilen makine için, bu makine ile birlikte uyumsuz olan diğer tüm makineler için 2, 3 ve 4 adımlarını uygula.
2- $\forall t \in \{0, 1, \dots, h\}$ için $[A_t, A_t+S]$ zaman penceresi kazanç üst limiti g_t olarak tanımlanır. Kazanç hesabı $[A_t, A_t+S]$ zaman penceresinde oluşan bir iş çakışması durumunda, hangi işlerin tercih edileceğinin belirlenmesini ele alır. Zaman penceresinde işlerde çakışma olması durumunda birim zaman diliminde en yüksek getiri miktarına sahip işin o zaman diliminde alındığı hesaplanmakta, çakışma olmadığı durumda birim zaman diliminde alınabilecek işin o zaman diliminde alındığı hesaplanarak kazanç bulunmaktadır. Tüm zaman dilimlerinde çakışma olmaması durumunda alınabilecek işlerin tamamı bitirilebileceğinden dolayı zaman penceresinde yapılabilecek işlere ait getiriler toplamı hesaplanır. Basit bir ifade ile kazanç, işlerin parçalı olarak işlenebildiği durumda tek bir makinenin yayılma zamanı içinde elde edebileceği en büyük iş getirileri toplamıdır. Tüm zaman aralıkları için hesaplanan kazançlar artmayan biçimde sıralanır. Artmayan sırada p_h adet en yüksek kazanç ile bir B kümesi oluşturulur. p_h parametresi etkinlik için ayarlanmalıdır.
3- B kümesindeki her bir zaman aralığı başlangıcı t için, ataması yapılmamış işler ile M_t kümeleri oluşturulur. Bu kümenin oluşturulması sırasında $[A_t, A_t+S]$ aralığında işlenebilecek işler ile bir ağ yapısı oluşturulur. Başlangıç ve bitiş düğümleri dışındaki her bir düğüm ile bir iş temsil edilir. Birbiri arkasından işlenebilecek işler arasında yönlü ark oluşturulur ve bu arkların değerleri, işlerin alınması durumunda işlerin o makinedeki getirileri, alınmaması durumunda ise 0 olarak belirlenir. Oluşturulan bu ağ modeli dinamik programlama problemi biçiminde modellenerek çözüldüğünde hangi işlerin $[A_t, A_t+S]$ aralığında işlenmesi durumunda w_{jk} toplamının en büyük olacağı tek bir makine için hesaplanmış olunur. M_t kümesi bu işleri içermektedir İş

getirilerinden makinenin maliyeti olan c_k çıkartılarak makinenin çalıştırılması durumunda elde edilecek net kar hesaplanır ve net kar miktarının artı çıkmaması durumunda M_t kümesine hiçbir iş alınmaz.

4- B kümesindeki her bir zaman aralığı başlangıcı için hesaplanan kümelerden, en büyük kar toplamına sahip olan M_t kümesindeki işler, karşılık gelen makineye atanır. Makine sıralamasında makine var ise ve atanacak işler bitmemiş ise ikinci adıma dönülür, diğer durumda ise algoritma bitirilir.

6.2. Yayılma Zamanlı Genelleştirilmiş Operasyonel Sabit İş Çizelgeleme Problemi için Önerilen Hibrit Metasezgisel Algoritma

Önerilen metasezgisel algoritma, başlangıç çözümlerin oluşturulması, seçim, çaprazlama, değişinim, yerel arama, değerlendirme ve değiştirme adımlarını içermektedir. Kromozom temsilinde hangi makinenin hangi işi aldığı veya almadığı bilgisi 0 - 1 tamsayılı değişkenler ile ifade edilmektedir. İşlerin makinelerdeki getirileri aynı olmadığından dolayı uygunluk değeri hesaplanırken işin atandığı makineye ait getirisi dikkate alınmaktadır. Ayrıca makine maliyetleri de işlerin toplam getirisinden düşülerek, elde edilecek net kar miktarı uygunluk değeri olarak hesaplanmaktadır.

Algoritmanın ilk adımı başlangıç çözümlerinin oluşturulmasıdır. Önerilen algoritmada dört farklı başlangıç çözüm oluşturma yöntemi kullanılmaktadır.

İlk yöntemde açgözlü sezgisel algoritma çözümü başlangıç çözümünde kullanılmaktadır. Açgözlü sezgisel algoritma ile üretilen çözümler birbirine oldukça benzer olduğundan dolayı başlangıç çözümüne sadece bir tane çözüm alınmaktadır.

Başlangıç çözüm oluşturmak için kullanılan ikinci yöntem, rastgele makinelere sırayla atama yapılmasıdır. Bu yöntemde ilk olarak hangi makinelere hangi sıra ile atama yapılacağı belirlenmektedir. Makine uyumsuzlukları bu noktada önemli olduğundan seçilen bir makinenin uyumsuz olduğu diğer makineler sonradan atama yapılacak biçimde seçilememektedir. Seçilen makineye, ataması yapılmamış rastgele bir iş seçilerek atanmaktadır. Bu atama sonucunda makinenin yayılma zamanı kısıtına bağlı olarak bazı işlerin makinedeki en az bir iş ile uyumsuz olduğu ve makineye alınamayacağı bilinmektedir. Eğer makineye alınabilecek işler listesinde halen iş bulunmakta ise makineye alınabilecek işler listesinden rastgele bir iş seçilerek devam edilmektedir. Bu şekilde makineye ataması yapılabilecek iş kalmayana kadar işler makineye atanmaktadır. Makineye alınabilecek iş kalmadığında ataması yapılacak bir sonraki makine belirlenerek devam edilmektedir.

Üçüncü yöntemde ise makine seçimi yine rastgele belirlenmekte, seçilen işlerin nasıl seçileceği değiştirilmektedir. İlk olarak bir makineye bir iş rastgele seçilerek atanmaktadır. Sonrasında makinede bulunan işlerin uyumlu oldukları ve henüz bir makineye atanmamış işler, makinede en az boşluk kalacak biçimde seçilmektedir. Eşitlik durumlarında ise bu işlerden bir tanesi rastgele seçilmektedir. Bu şekilde makineye ataması yapılabilecek iş kalmayana kadar işler makineye atanmaktadır. Makineye alınabilecek iş kalmadığında ataması yapılacak bir sonraki makine belirlenerek devam edilmektedir.

Başlangıç çözümünün oluşturulması için kullanılan son yöntemde ise makine seçimi rastgele belirlenmekte, işlerin seçiminde ataması yapılacak makine için olurlu işler arasından rastgele veya kazanç/işlem süresi oranına göre sıra ile atama yapılmaktadır.

Seçim operatörü olarak olasılıksal ikili turnuva kullanılmaktadır. Ancak popülasyonda iyi olan bireyin sıkça seçilmesi durumunda erken yakınsama sorunu ile karşılaşılmakta ve popülasyondaki tüm bireyler aynı olmaktadır. Bu sebeple iyi bireyin seçim olasılığı oldukça düşük tutulmuştur.

Kullanılan çaprazlama operatöründe, ebeveynlerde en yüksek getirili makineye ait çizelge çocuk bireye alınmakta, sonrasında ise çocuk bireye alınmış bütün işler ebeveynlerden tekrar alınamayacağı için ebeveynlerin çizelgelerinden çıkartılmaktadır. Ebeveynlerin çizelgeleri değiştiğinden makine karları da tekrar hesaplanmaktadır. Makine uyumsuzluklarının önüne geçmek için, ebeveyne ait bir makineden işler çocuk bireye aktarıldığında, ebeveynlerin tamamından bu makine ile uyumsuz olan bütün makinelerin çizelgeleri çıkartılmaktadır. Böylece uyumsuz iki makinenin çocuk bireye aktarılmasının önüne geçilmektedir. Bu şekilde, karlılığı artı olan makine çizelgesi kalmayana kadar devam edilmektedir.

Değişinim operatörü olarak kullanılan üç farklı operatör bulunmaktadır. İlk değişinim operatörü, çalıştırılan makinelerden en düşük getiriye sahip makinede bulunan işleri çizelgeden çıkartmaktadır. İkinci değişinim operatörü tüm makinelere atanmış olan işlerden rastgele iş çıkartmaktadır. İşlerin çıkartılma olasılıkları PJobRem parametresi ile kontrol edilmektedir. Bu operatör tüm makinelerden iş çıkartabilmektedir. Üçüncü değişinim operatörü, rastgele bir makine seçerek makinedeki işleri çizelgeden çıkartmaktadır. Her üç değişinim operatörü için ayrı ayrı olasılıklar belirlenmiştir. Değişinim operatörü uygulandığında üç değişim operatöründen sadece bir tanesi kullanılmaktadır.

Sezgisel yerel arama; algoritmanın temel noktasını oluşturmakta ve hem çaprazlama operatörü hem de değişinim operatörü sonrasında uygulanmaktadır. İki aşamalı olan sezgisel yerel aramanın ilk aşaması yinelemeli olarak çalışmakta, her yinelemede belirli kriterlere göre atanmamış bir işi seçmekte ve seçilen işi makinelerden birisine atamaya çalışmaktadır. İş seçimi, atanmamış işlerden ya rastgele ya da en yüksek kazanç/işlem süresi oranına sahip iş seçilerek yapılmaktadır. Hangi iş seçimi yöntemin kullanılacağı hakkında karar vermek için p_s parametresi kullanılmaktadır. Seçilmiş olan iş, tüm olası makinelerde boşluklara atanmaya çalışılmakta, birden fazla makineye atanabilecek ise atama sonunda en az boşluk kalacak makineye atama yapılmaktadır. Eğer seçilen iş, hiçbir makinede mevcut boşluklara alınamıyor ise, seçilen işin makinelere atanabilmesi için bazı işlerin makinelerden çıkartılması gerekmektedir. Her makine için makineden çıkartılması gereken işler, makinede mevcut olup ele alınan iş ile uyumsuz olan işlerdir. Çıkartılması için belirlenen bu işlerin makineye alınmaması durumunda makinedeki getirilerinden vazgeçilmekte, ele alınan işin getirisi kazanılmaktadır. Eğer makinedeki bazı işleri çıkartarak seçilen işi atama hareketi getiri ile sonuçlanıyor ise bu atama hareketi yapılmaktadır. Aksi durumda makineden çıkartılacak işlerin getirileri toplamı, ele alınan işin getirisinden büyük olduğunda ele alınan işi makineye almamak daha iyi olmaktadır. İş seçim yöntemine uygun olarak seçilebilecek tüm işler denenmesine rağmen hiçbir iş iyileştirme sağlamamakta ise sezgisel yerel arama sonlandırılmaktadır. Eğer bir işin herhangi bir makineye alınması getiri artışı ile sonuçlanmakta ise, iş makineye atanmakta, bu iş ile uyumsuz olan işler ise diğer makinelere atanabilecek olduğundan çizelgelenmemiş işler listesi güncellenmekte ve sezgisel yerel arama sürecinin ilk aşaması tekrar başlatılmaktadır. İkinci aşamada ise hedeflenen; uyumsuz makineler listesinde olmayan ve kullanılmayan makineler var ise kullanım dışı olan bu makinelerin çizelgelenmesidir. Çaprazlama operatörü veya değişinim operatörü sonucunda çalıştırılmayan makine olabildiğinden makinenin çizelgelenmesi kar ile sonuçlanabilmektedir. Bu yerel arama operatörünün kullanımı, makinenin çalıştırılmadığı yani makine getirisinin bulunmadığı durumlarda anlamlı olmamaktadır.

Değerlendirme ve değiştirme aşamalarında popülasyon tabanlı bir geçiş kullanılmaktadır. Oluşturulan aday popülasyondaki bireylerin karlılıkları hesaplanmakta ve mevcut popülasyon ile bu popülasyon birleştirilmekte, birleştirilmiş ara popülasyondan en uygun bireyler seçilerek popülasyon güncellenmektedir.

7. UYGULAMA

Sayısal hesaplamalar için iş başlangıç zamanları, iş işlem süreleri, iş getirileri, makine yayılma zamanları ve makine maliyetleri gibi değerler gerekmektedir. Problemlerde bu sayısal değerlerin literatürde üç farklı parametreye göre üretildiği ve diğer bir çalışmada da bu parametrelerin kullanıldığı görülmektedir (Eliiyi ve Azizoğlu, 2006; Rossi ve diğerleri, 2010). Bu parametreler işlerin hazır olma zamanları için r, işlerin işlem süreleri için p ve iş ağırlıkları için w olarak verilmektedir.

r parametresi için;

- $r = 1$ olması durumunda iş hazır olma zamanlarının $[0, 200]$ aralığında tekdüze dağılım ile üretildiği,
- $r = 2$ olması durumunda iş hazır olma zamanlarının %30 ihtimalle $[30, 40]$ aralığında tekdüze dağılım ile üretildiği, %30 ihtimalle $[130, 140]$ aralığında tekdüze dağılım ile üretildiği, %40 ihtimalle $[41, 129] \cup [141, 200]$ aralığında tekdüze dağılım ile üretildiği,

durum incelenmektedir.

p parametresi için;

- $p = 1$ olması durumunda işlem sürelerinin $[5, 10]$ aralığında tekdüze dağılım ile üretildiği,
- $p = 2$ olması durumunda işlem sürelerinin $[5, 40]$ aralığında tekdüze dağılım ile üretildiği,

durum incelenmektedir.

w parametresi için;

- $w = 1$ olması durumunda iş getirilerinin işlem sürelerine eşit üretildiği,

- $w = 2$ olması durumunda iş getirilerinin [5, 10] aralığında tekdüze dağılım ile üretildiği,
- $w = 3$ olması durumunda iş getirilerinin [5, 40] aralığında tekdüze dağılım ile üretildiği,

durum incelenmektedir.

Makine yayılma zamanları ve makine maliyetleri daha önce yapılan çalışmalarda kullanılmadığından dolayı üretimleri için bir öneri bulunmamaktadır.

Çalışma boyunca 3,4 Ghz işlemci hızına, 32 Gb belleğe sahip bir bilgisayar kullanılmıştır.

Literatürde kullanılmakta olan parametreler ile her birinden 100 adet olmak üzere yayılma zamanlı operasyonel sabit iş çizelgeleme problemi üretilmiştir. Bu problemler için yarı tanımlı programlama gevşetmesi ve doğrusal programlama gevşetmesi, en uygun çözüm değeri ile kıyaslanmaktadır. Bu değerler problem için bir üst sınır ifade etmektedir. Problem verileri tamsayılı olduğu için herhangi bir ondalıklı çözüm değerinin olamayacağı bilinmektedir. Bu durum da göz önüne alınarak elde edilen çözüm değerinin tam kısmı alınarak elde edilen değerlerin ortalaması da elde edilmiştir. Ayrıca problemler açgözlü sezgisel algoritma ile de çözülmüş ve problem için alt sınırları hesaplanmıştır. Çalışma sonuçları Tablo 7.1 ile verilmektedir. Tablo 7.1 incelendiğinde, doğrusal programlama gevşetmesinin üst sınır belirlemekte oldukça yetersiz kaldığı, açgözlü sezgisel algoritmanın en uygun değere oldukça yakın alt sınırlar bulduğu görülmektedir. Yarı tanımlı programlama gevşetmesi ise problemlerin önemli bir bölümü için en uygun çözüm değerinin üst sınırını polinom çözüm zamanı ile vermiştir.

Tablo 7.1. DP, TYP, yuvarlamalı YTP Gevşetme Modellerinin ve Açgözlü Sezgisel Algoritmanın En Uygun Çözüme Uzaklık Açısından Kıyaslanması

n	m	r	p	w	DP gevşetmesinin en uygun çözümden % uzaklığının ortalaması	YTP gevşetmesinin en uygun çözümden % uzaklığının ortalaması	YTP gevşetmesinin yuvarlama ile en uygun çözümden % uzaklığının ortalaması	Açgözlü Sezgiselin en uygun çözümden % uzaklığının ortalaması
20	2	1	1	1	38,003	0,001	0	-2,398
20	2	1	1	2	36,811	0	0	-1,536
20	2	1	1	3	32,24	0	0	-1,64
20	2	1	2	1	75,373	0,137	0,104	-0,961
20	2	1	2	2	62,166	0,039	0,036	-1,22
20	2	1	2	3	55,031	0,082	0,068	-1,441
20	2	2	1	1	40,154	0	0	-1,455
20	2	2	1	2	36,76	0	0	-1,625
20	2	2	1	3	37,469	0	0	-1,14
20	2	2	2	1	61,545	0,085	0,059	-0,44
20	2	2	2	2	55,828	0	0	-1,388
20	2	2	2	3	52,106	0	0	-1,699
20	3	1	1	1	18,432	0,014	0,009	-1,984
20	3	1	1	2	17,585	0	0	-1,572
20	3	1	1	3	14,091	0	0	-1,72
20	3	1	2	1	57,081	0,175	0,146	-1,236
20	3	1	2	2	48,989	0,099	0,058	-1,875
20	3	1	2	3	42,7	0,08	0,062	-1,491
20	3	2	1	1	30,895	0	0	-0,902
20	3	2	1	2	29,212	0	0	-0,966
20	3	2	1	3	27,525	0	0	-0,949
20	3	2	2	1	54,464	0,024	0,015	-0,788
20	3	2	2	2	42,797	0,056	0,052	-1,454
20	3	2	2	3	41,884	0,01	0,004	-1,719
20	4	1	1	1	7,472	0,011	0,007	-1,425
20	4	1	1	2	7,695	0	0	-1,089
20	4	1	1	3	4,767	0	0	-1,279
20	4	1	2	1	40,36	0,207	0,175	-1,093
20	4	1	2	2	35,628	0,098	0,075	-1,941
20	4	1	2	3	27,893	0,066	0,055	-1,83
20	4	2	1	1	26,778	0	0	-0,923
20	4	2	1	2	24,629	0	0	-0,889
20	4	2	1	3	21,688	0	0	-1,261
20	4	2	2	1	45,325	0,029	0,024	-0,927
20	4	2	2	2	39,266	0,006	0	-1,54

Tablo 7.1. (Devam) DP, TYP, yuvarlamalı YTP Gevşetme Modellerinin ve Açgözlü Sezgisel Algoritmanın En Uygun Çözüme Uzaklık Açısından Kıyaslanması

n	m	r	p	w	DP gevşetmesinin en uygun çözümden % uzaklığının ortalaması	YTP gevşetmesinin en uygun çözümden % uzaklığının ortalaması	YTP gevşetmesinin yuvarlama ile en uygun çözümden % uzaklığının ortalaması	Açgözlü Sezgiselin en uygun çözümden % uzaklığının ortalaması
20	4	2	2	3	37,153	0,008	0	-1,461
30	2	1	1	1	51,192	0,01	0	-1,346
30	2	1	1	2	48,661	0	0	-1,821
30	2	1	1	3	45,124	0	0	-1,854
30	2	1	2	1	91,111	0,113	0,07	-1,092
30	2	1	2	2	65,279	0,016	0	-1,498
30	2	1	2	3	63,931	0,015	0,008	-1,582
30	2	2	1	1	45,127	0,023	0,022	-1,522
30	2	2	1	2	41,862	0	0	-1,392
30	2	2	1	3	41,536	0,002	0	-0,826
30	2	2	2	1	72,01	0,016	0,006	-1,216
30	2	2	2	2	58,135	0,099	0,049	-1,857
30	2	2	2	3	55,632	0,03	0,022	-1,409
30	3	1	1	1	31,125	0,033	0,023	-1,775
30	3	1	1	2	31,587	0,01	0,006	-1,814
30	3	1	1	3	25,022	0	0	-1,962
30	3	1	2	1	77,677	0,14	0,1	-1,591
30	3	1	2	2	58,732	0,094	0,041	-1,91
30	3	1	2	3	52,971	0,087	0,07	-2,124
30	3	2	1	1	35,888	0	0	-1,005
30	3	2	1	2	33,385	0	0	-1,395
30	3	2	1	3	31,932	0	0	-1,526
30	3	2	2	1	60,257	0,07	0,037	-1,338
30	3	2	2	2	49,996	0,015	0	-1,532
30	3	2	2	3	49,914	0,038	0,034	-1,629
30	4	1	1	1	16,117	0,007	0,005	-1,828
30	4	1	1	2	15,557	0	0	-1,669
30	4	1	1	3	12,295	0	0	-1,776
30	4	1	2	1	63,075	0,16	0,117	-1,245
30	4	1	2	2	50,113	0,105	0,045	-2,47
30	4	1	2	3	41,172	0,035	0,022	-2,718
30	4	2	1	1	28,931	0	0	-1,073
30	4	2	1	2	28,771	0	0	-0,941
30	4	2	1	3	26,637	0	0	-1,236
30	4	2	2	1	53,229	0,047	0,028	-1,598

Tablo 7.1. (Devam) DP, TYP, yuvarlamalı YTP Gevşetme Modellerinin ve Açgözlü Sezgisel Algoritmanın En Uygun Çözüme Uzaklık Açısından Kıyaslanması

n	m	r	p	w	DP gevşetmesinin en uygun çözümden % uzaklığının ortalaması	YTP gevşetmesinin en uygun çözümden % uzaklığının ortalaması	YTP gevşetmesinin yuvarlama ile en uygun çözümden % uzaklığının ortalaması	Açgözlü Sezgiselin en uygun çözümden % uzaklığının ortalaması
30	4	2	2	2	42,889	0,015	0	-1,918
30	4	2	2	3	42,713	0,022	0,017	-1,817
40	2	1	1	1	61,696	0	0	-1,79
40	2	1	1	2	57,456	0,004	0	-1,458
40	2	1	1	3	49,75	0,017	0,015	-2,397
40	2	1	2	1	100,622	0,075	0,033	-2,278
40	2	1	2	2	70,727	0,003	0	-1,497
40	2	1	2	3	67,715	0,007	0,004	-2,288
40	2	2	1	1	52,037	0	0	-1,223
40	2	2	1	2	48,344	0	0	-1,547
40	2	2	1	3	45,778	0	0	-1,739
40	2	2	2	1	82,312	0,057	0,04	-1,308
40	2	2	2	2	62,837	0	0	-1,896
40	2	2	2	3	60,63	0,005	0	-1,369
40	3	1	1	1	41,228	0,006	0	-1,906
40	3	1	1	2	40,341	0,008	0,005	-2,136
40	3	1	1	3	33,768	0,012	0,01	-2,154
40	3	1	2	1	91,266	0,156	0,1	-2,303
40	3	1	2	2	64,666	0,083	0,019	-1,988
40	3	1	2	3	58,917	0,071	0,057	-2,673
40	3	2	1	1	37,87	0	0	-1,322
40	3	2	1	2	36,938	0	0	-1,252
40	3	2	1	3	34,857	0	0	-1,523
40	3	2	2	1	68,291	0,067	0,031	-1,016
40	3	2	2	2	52,124	0,014	0	-1,95
40	3	2	2	3	50,763	0,03	0,02	-1,873
40	4	1	1	1	26,543	0	0	-1,886
40	4	1	1	2	24,178	0,019	0,013	-1,912
40	4	1	1	3	20,051	0	0	-2,287
40	4	1	2	1	81,441	0,247	0,188	-1,684
40	4	1	2	2	56,237	0,16	0,105	-2,835
40	4	1	2	3	51,268	0,139	0,118	-2,744
40	4	2	1	1	31,787	0	0	-1,321
40	4	2	1	2	29,565	0	0	-1,114
40	4	2	1	3	29,185	0	0	-1,081

Tablo 7.1. (Devam) DP, TYP, yuvarlamalı YTP Gevşetme Modellerinin ve Açgözlü Sezgisel Algoritmanın En Uygun Çözüme Uzaklık Açısından Kıyaslanması

n	m	r	p	w	DP gevşetmesinin en uygun çözümden % uzaklığının ortalaması	YTP gevşetmesinin en uygun çözümden % uzaklığının ortalaması	YTP gevşetmesinin yuvarlama ile en uygun çözümden % uzaklığının ortalaması	Açgözlü Sezgiselin en uygun çözümden % uzaklığının ortalaması
40	4	2	2	1	59,677	0,142	0,116	-1,51
40	4	2	2	2	46,951	0,06	0,037	-2,056
40	4	2	2	3	45,42	0,054	0,042	-2,678
50	2	1	1	1	68,638	0,007	0	-1,071
50	2	1	1	2	63,058	0,016	0,012	-1,452
50	2	1	1	3	58,728	0,009	0,007	-1,257
50	2	1	2	1	106,414	0,047	0,016	-3,366
50	2	1	2	2	73,052	0,008	0	-1,451
50	2	1	2	3	66,62	0,008	0,003	-1,772
50	2	2	1	1	56,446	0,018	0,007	-1,012
50	2	2	1	2	52,07	0,006	0	-1,288
50	2	2	1	3	49,413	0,001	0	-1,564
50	2	2	2	1	89,316	0,072	0,034	-1,38
50	2	2	2	2	66,678	0,025	0	-1,596
50	2	2	2	3	63,125	0,038	0,033	-1,992
50	3	1	1	1	52,203	0,005	0	-2,033
50	3	1	1	2	49,433	0,006	0	-1,996
50	3	1	1	3	42,171	0,004	0,001	-1,667
50	3	1	2	1	99,153	0,116	0,069	-2,451
50	3	1	2	2	66,389	0,099	0,04	-1,868
50	3	1	2	3	60,491	0,053	0,045	-2,768
50	3	2	1	1	41,885	0	0	-1,342
50	3	2	1	2	40,331	0	0	-1,309
50	3	2	1	3	37,671	0	0	-1,254
50	3	2	2	1	75,231	0,089	0,062	-1,654
50	3	2	2	2	57,287	0,016	0	-2,673
50	3	2	2	3	54,963	0,035	0,025	-1,821
50	4	1	1	1	36,237	0,007	0,004	-2,113
50	4	1	1	2	32,564	0,007	0,007	-2,065
50	4	1	1	3	26,009	0,006	0,006	-2,058
50	4	1	2	1	92,831	0,207	0,144	-2,147
50	4	1	2	2	62,118	0,162	0,106	-2,712
50	4	1	2	3	55,453	0,117	0,101	-2,777
50	4	2	1	1	33,7	0	0	-1,308
50	4	2	1	2	32,911	0	0	-1,082

Tablo 7.1.(Devam) DP, TYP, yuvarlamalı YTP Gevşetme Modellerinin ve Açgözlü Sezgisel Algoritmanın En Uygun Çözüme Uzaklık Açısından Kıyaslanması

n	m	r	p	w	DP gevşetmesinin en uygun çözümden % uzaklığının ortalaması	YTP gevşetmesinin en uygun çözümden % uzaklığının ortalaması	YTP gevşetmesinin yuvarlama ile en uygun çözümden % uzaklığının ortalaması	Açgözlü Sezgiselin en uygun çözümden % uzaklığının ortalaması
50	4	2	1	3	29,581	0	0	-1,352
50	4	2	2	1	67,897	0,087	0,058	-1,512
50	4	2	2	2	49,375	0,027	0,007	-2,606
50	4	2	2	3	47,627	0,052	0,044	-2,409

Yarı tanımlı programlama problemi için kullanılan çözüm algoritmaları her ne kadar polinom zamanda çözüm elde ediyor olsalar da modelin, problemdeki iş sayısı 50, makine sayısı 4 olduğunda uygulamada oldukça yavaş kaldığı görülmüştür. Yarı tanımlı programlama algoritmalarının ilerleyen zamanlarda gelişeceği ve daha hızlı çalışan algoritmaların geliştirilmesi ile, problemler için üst sınırların makul sürelerde üretilmesinin mümkün hale geleceği düşünülmektedir.

Yayılma zamanlı genelleştirilmiş operasyonel sabit iş çizelgeleme problemi incelendiğinde makine maliyetleri, makine yayılma zamanları ve işlerin makinelerdeki farklı getirileri için bir yöntem literatürde bulunmamaktadır. Makine yayılma zamanları [75, 125] aralığında tekdüze dağılım ile üretilmiştir. Makine temel maliyetleri [0, 100] aralığında tekdüze dağılım ile üretilmiştir. Makineye ait ikinci durumu oluşturan ek maliyetler için birim zamanda artan veya azalan maliyet modelinden hangisinin kullanılacağı rastgele belirlenmiştir. Ek maliyetler makine temel maliyetine göre ±%20 sapmayı aşamayacak biçimde tekdüze sağılıma uygun olarak rastgele belirlenmiştir. Makinelerin ek yayılma zamanları ise makine yayılma zamanının %10'u ile %50'si arasında tekdüze dağılıma uygun olarak rastgele belirlenmiştir. Her makine için iş getirileri ise makinelerin tamamında w parametresine göre üretilmiştir. Sadece w=1 durumunda işlerin işlem süreleri her makinede eşit olduğundan dolayı makine getirileri de eşit olmaktadır.

Yayılma zamanlı genelleştirilmiş operasyonel sabit iş çizelgeleme problemi için 20, 30, 40, 50, 60, 70, 80, 90, 100 iş ve 2, 3, 4 makine için her bir problem kümesinden 10 adet problem oluşturulmuştur. Bu problemler küçük boyutlu problemler olarak ifade edilmektedirler. Ayrıca geliştirilen metasezgisel yöntem için 250, 500, 1000 iş ve 2, 50, 75, 100 makine için her bir problem kümesinden 10 adet problem oluşturulmuştur. Bu problemler ise büyük boyutlu problemler olarak ifade edilmektedir.

Genelleştirilmiş problem için tamsayılı programlama modeli, gurobi yazılımı ile gurobi akademik lisansı ile çözülmüştür. Gurobi yazılımı ile makul sürelerde elde edilebilen çözümler 250 iş, 4(2 makine 2 durum) makine problemleri olmuştur.

Tamsayılı programlama modeli ile çözülebilen problemlerden elde edilen en uygun çözüm değerleri, geliştirilen metasezgisel yöntemin parametrelerinin belirlenmesinde kullanılmıştır. Metasezgisel yöntemde kullanılan parametreler:

- Başlangıç popülasyondaki tam rastgele çözüm sayısı(FullRandomPopulationSize),
- Başlangıç popülasyondaki açgözlü çözüm sayısı(GreedyPopulationSize),
- Başlangıç popülasyondaki rastgele iş seçim yöntemi çözüm sayısı(RandomTasksPopulationSize),
- Başlangıç popülasyondaki tek makine yöntemi çözüm sayısı(SingleProcessorPopulationSize),
- Yineleme sayısı(NoOfIterations),
- Sezgisel yöntemin her makine için bakacağı alt dal sayısı(ph),
- İş seçim stratejisi olarak kazanç/işlem süresi oranı yüksek olanının seçim olasılığı(phgt),
- Daha uygun çözümün seçim olasılığı(Pbetter),
- Çaprazlanacak ebeveyn sayısı(NoOfSelectedParents),
- Çaprazlama oranı(Pc),
- En küçük getirili makineyi çizelgeden çıkartma mutasyonu oranı(PMutationOne),
- Çizelgeden rastgele iş çıkartma mutasyon oranı(PMutationTwo),
- Çizelgeden rastgele makine çıkartma mutasyon oranı(PMutationThree),
- Mutasyonda silinecek işlerin seçilme olasılığı(PJobRem),
- Yerel aramada iş seçim stratejisi olarak kazanç/işlem süresi oranı yüksek olanının seçim olasılığı(Ps),
- Çocuk bireye yerel arama olarak iş yerel araması uygulama olasılığı(PLocalSearchJobs),

- Çocuk bireye yerel arama olarak makine yerel araması uygulama olasılığı(PLocalSearchMachines),

biçiminde ifade edilebilir. Geliştirilen metasezgisel yönteme ait bu parametrelerin f-race algoritması ile belirlenmesi planlanmıştır. Ancak parametre sayısının çok fazla olması, test edilecek parametre uzayının çok büyük olmasına sebep olmaktadır. Parametre uzayının çok büyük olmasından dolayı bazı parametreler sabit tutulmuştur. F-race algoritması ile yapılan çalıştırmalarda geliştirilen metasezgisel algoritmanın 45 saniye boyunca çalışmasına izin verilmiştir. Küçük boyutlu problemlerde f-race ile yapılan parametre belirleme çalışmalarında test edilen parametreler şu şekildedir:

- FullRandomPopulationSize: {10; 20}
- GreedyPopulationSize: {1}
- SingleProcessorPopulationSize: {10; 20}
- RandomTasksPopulationSize: {10; 20}
- NoOfIterations: {500; 2500}
- ph: {5}
- phgt: {0,50}
- Pbetter: {0,01}
- NoOfSelectedParents: {2}
- Pc: {0,60; 0,75; 0,90}
- PMutationOne: {1/3}
- PMutationTwo: {2/3}
- PMutationThree: {3/3}
- PJobRem: {4; 8; 16}
- Ps: {0,25; 0,50; 0,75}
- PLocalSearchJobs: {0,50; 0,70; 0,90}
- PLocalSearchMachines: {0,25; 0,50; 0,75}

GreedyPopulationSize parametresinin başlangıç çözüme oldukça etkisi bulunmaktadır. Bu parametre, başlangıç çözümde bulunacak sezgisel çözümlerin sayısını vermektedir. Sezgisel üretilen çözümler birbirine oldukça benzemekte olduğundan bu parametre değerinin 1'den büyük seçilmesi durumunda başlangıç çözümler çözüm uzayının küçük bir bölümünü temsil etmektedir. Bu sebeple bu parametre 1 değerine sabitlenmiştir.

ph parametresi literatürde kullanıldığı değer olan 5 değerine sabitlenmiştir (Rossi ve diğerleri, 2010).

phgt parametresi literatürde kullanıldığı biçimde alınmamış, çözümlerin yarısının rastgele iş seçimi ile kalan yarısının ise kazanç/işlem süresi oranına göre seçilmesinin uygun olduğu öngörülerek 0,5 değerine sabitlenmiştir.

Pbetter parametresi, hızlı yakınsamanın önüne geçmek için oldukça düşük tutulmuştur. Öncelikle popülasyondaki çözümlerden kötü olanların iyileştirilmesi ve tüm popülasyonun belirli bir seviyeye getirilmesi amaçlanmıştır.

NoOfSelectedParents parametresi birçok genetik algoritmanın çaprazlama operatöründe olduğu gibi iki ebeveyn kullanılarak çaprazlama işleminin gerçekleştirilmesini sağlamaktadır.

PMutationOne, PMutationTwo ve PMutationThree parametreleri mutasyon için kullanılacak yöntemi belirlediğinden eşit olarak seçilmiştir.

Diğer parametreler f-race algoritması ile değerlendirilmiş ve FullRandomPopulationSize parametresi 10; SingleProcessorPopulationSize parametresi 10; RandomTasksPopulationSize parametresi 10; NoOfIterations parametresi 2500; Pc parametresi 0,75; PJobRem parametresi 4; Ps parametresi 0,50; PLocalSearchJobs parametresi 0,70; PLocalSearchMachines parametresi 0,75 olarak bulunmuştur. NoOfIterations parametresi yineleme sayısını ifade etmekte ancak problem büyüklüğüne ve problem karmaşıklığına bağlı olmaktadır. PJobRem parametresinin iş sayısı ve makine sayısına bağlı olması gerektiği düşünüldüğünden parametre değerinin iş ve makine sayısına bağlı bir oranı kullanılmaktadır.

F-race algoritması kullanılarak bulunan parametreler ile küçük boyutlu problemler çözülmüştür. Geliştirilen algoritma ile elde edilen çözümler, tamsayılı programlama kullanılarak elde edilen en uygun çözümler kıyaslanmıştır. Bulunan çözümlerin en uygun çözüme olan uzaklıklarına ait ortalama değerler Tablo 7.2 ile verilmektedir.

Tablo 7.2. Geliştirilen metasezgisel algoritmanın, küçük boyutlu problemlerdeki performansı

n	m	r	p	w	En uygun değerden ortalama % sapma
20	2	1	1	1	0
20	2	1	1	2	0
20	2	1	1	3	0
20	2	1	2	1	0
20	2	1	2	2	0
20	2	1	2	3	0
20	2	2	1	1	0
20	2	2	1	2	0
20	2	2	1	3	0
20	2	2	2	1	0
20	2	2	2	2	0
20	2	2	2	3	0
20	3	1	1	1	0
20	3	1	1	2	0
20	3	1	1	3	0
20	3	1	2	1	0
20	3	1	2	2	0
20	3	1	2	3	0
20	3	2	1	1	0
20	3	2	1	2	0
20	3	2	1	3	0,419
20	3	2	2	1	0
20	3	2	2	2	0
20	3	2	2	3	0
20	4	1	1	1	0,467
20	4	1	1	2	0,233
20	4	1	1	3	0,614
20	4	1	2	1	0
20	4	1	2	2	0
20	4	1	2	3	0,565
20	4	2	1	1	0
20	4	2	1	2	0,212
20	4	2	1	3	1,207
20	4	2	2	1	0
20	4	2	2	2	0
20	4	2	2	3	0,074
30	2	1	1	1	0
30	2	1	1	2	0
30	2	1	1	3	0
30	2	1	2	1	0
30	2	1	2	2	0

Tablo 7.2.(Devam) Geliştirilen metasezgisel algoritmanın, küçük boyutlu problemlerdeki performansı

n	m	r	p	w	En uygun değerden ortalama % sapma
30	2	1	2	3	0
30	2	2	1	1	0
30	2	2	1	2	0
30	2	2	1	3	0
30	2	2	2	1	0
30	2	2	2	2	0
30	2	2	2	3	0
30	3	1	1	1	0,081
30	3	1	1	2	0,303
30	3	1	1	3	0
30	3	1	2	1	0
30	3	1	2	2	0
30	3	1	2	3	0
30	3	2	1	1	0
30	3	2	1	2	0
30	3	2	1	3	0,104
30	3	2	2	1	0,155
30	3	2	2	2	0
30	3	2	2	3	0
30	4	1	1	1	0,152
30	4	1	1	2	0,44
30	4	1	1	3	0,462
30	4	1	2	1	0,035
30	4	1	2	2	0,29
30	4	1	2	3	1,105
30	4	2	1	1	0
30	4	2	1	2	0,4
30	4	2	1	3	1,756
30	4	2	2	1	0,345
30	4	2	2	2	0
30	4	2	2	3	0,316
40	2	1	1	1	0
40	2	1	1	2	0
40	2	1	1	3	0,025
40	2	1	2	1	0
40	2	1	2	2	0
40	2	1	2	3	0
40	2	2	1	1	0
40	2	2	1	2	0
40	2	2	1	3	0
40	2	2	2	1	0

Tablo 7.2.(Devam) Geliştirilen metasezgisel algoritmanın, küçük boyutlu problemlerdeki performansı

n	m	r	p	w	En uygun değerden ortalama % sapma
40	2	2	2	2	0
40	2	2	2	3	0
40	3	1	1	1	0
40	3	1	1	2	0,848
40	3	1	1	3	0,087
40	3	1	2	1	0
40	3	1	2	2	0
40	3	1	2	3	0
40	3	2	1	1	0,448
40	3	2	1	2	0
40	3	2	1	3	0,114
40	3	2	2	1	0,232
40	3	2	2	2	0
40	3	2	2	3	0,098
40	4	1	1	1	0,391
40	4	1	1	2	0,22
40	4	1	1	3	0,145
40	4	1	2	1	0,326
40	4	1	2	2	0,714
40	4	1	2	3	0,384
40	4	2	1	1	0,123
40	4	2	1	2	0,109
40	4	2	1	3	0,359
40	4	2	2	1	0,253
40	4	2	2	2	0
40	4	2	2	3	0,113
50	2	1	1	1	0
50	2	1	1	2	0,39
50	2	1	1	3	0
50	2	1	2	1	0,116
50	2	1	2	2	0
50	2	1	2	3	0
50	2	2	1	1	0
50	2	2	1	2	0,252
50	2	2	1	3	0
50	2	2	2	1	0
50	2	2	2	2	0
50	2	2	2	3	0
50	3	1	1	1	0,123
50	3	1	1	2	0,561
50	3	1	1	3	0,068

Tablo 7.2.(Devam) Geliştirilen metasezgisel algoritmanın, küçük boyutlu problemlerdeki performansı

n	m	r	p	w	En uygun değerden ortalama % sapma
50	3	1	2	1	0,041
50	3	1	2	2	0
50	3	1	2	3	0
50	3	2	1	1	0
50	3	2	1	2	0,322
50	3	2	1	3	0,184
50	3	2	2	1	0,371
50	3	2	2	2	0
50	3	2	2	3	0,043
50	4	1	1	1	0,392
50	4	1	1	2	0,626
50	4	1	1	3	0,042
50	4	1	2	1	0,835
50	4	1	2	2	0,194
50	4	1	2	3	0,326
50	4	2	1	1	0,093
50	4	2	1	2	0,38
50	4	2	1	3	0,383
50	4	2	2	1	0,497
50	4	2	2	2	0
50	4	2	2	3	0
60	2	1	1	1	0
60	2	1	1	2	0
60	2	1	1	3	0
60	2	1	2	1	0
60	2	1	2	2	0
60	2	1	2	3	0,142
60	2	2	1	1	0
60	2	2	1	2	0
60	2	2	1	3	0,089
60	2	2	2	1	0
60	2	2	2	2	0
60	2	2	2	3	0
60	3	1	1	1	0,478
60	3	1	1	2	0,26
60	3	1	1	3	0,491
60	3	1	2	1	0,272
60	3	1	2	2	0
60	3	1	2	3	0,04
60	3	2	1	1	0
60	3	2	1	2	0

Tablo 7.2.(Devam) Geliştirilen metasezgisel algoritmanın, küçük boyutlu problemlerdeki performansı

n	m	r	p	w	En uygun değerden ortalama % sapma
60	3	2	1	3	0,017
60	3	2	2	1	0,914
60	3	2	2	2	0
60	3	2	2	3	0,233
60	4	1	1	1	0,453
60	4	1	1	2	0,684
60	4	1	1	3	0,457
60	4	1	2	1	0,485
60	4	1	2	2	0,546
60	4	1	2	3	0,174
60	4	2	1	1	0
60	4	2	1	2	0
60	4	2	1	3	0,764
60	4	2	2	1	0,471
60	4	2	2	2	0,225
60	4	2	2	3	0,828
70	2	1	1	1	0
70	2	1	1	2	0,245
70	2	1	1	3	0
70	2	1	2	1	0
70	2	1	2	2	0
70	2	1	2	3	0
70	2	2	1	1	0
70	2	2	1	2	0
70	2	2	1	3	0
70	2	2	2	1	0,223
70	2	2	2	2	0
70	2	2	2	3	0
70	3	1	1	1	0,572
70	3	1	1	2	0,286
70	3	1	1	3	0,094
70	3	1	2	1	0,187
70	3	1	2	2	0,297
70	3	1	2	3	0,051
70	3	2	1	1	0
70	3	2	1	2	0
70	3	2	1	3	0,138
70	3	2	2	1	0,592
70	3	2	2	2	0
70	3	2	2	3	0,42
70	4	1	1	1	0,458

Tablo 7.2.(Devam) Geliştirilen metasezgisel algoritmanın, küçük boyutlu problemlerdeki performansı

n	m	r	p	w	En uygun değerden ortalama % sapma
70	4	1	1	2	1,968
70	4	1	1	3	0,586
70	4	1	2	1	0,323
70	4	1	2	2	0
70	4	1	2	3	0,268
70	4	2	1	1	0,221
70	4	2	1	2	0,761
70	4	2	1	3	0,209
70	4	2	2	1	0,851
70	4	2	2	2	0
70	4	2	2	3	0,691
80	2	1	1	1	0,455
80	2	1	1	2	1,429
80	2	1	1	3	0
80	2	1	2	1	0,061
80	2	1	2	2	0
80	2	1	2	3	0
80	2	2	1	1	0,123
80	2	2	1	2	0
80	2	2	1	3	0
80	2	2	2	1	0
80	2	2	2	2	0
80	2	2	2	3	0
80	3	1	1	1	0,271
80	3	1	1	2	0,708
80	3	1	1	3	0,414
80	3	1	2	1	0,207
80	3	1	2	2	0,912
80	3	1	2	3	0,526
80	3	2	1	1	0,188
80	3	2	1	2	0,364
80	3	2	1	3	0,056
80	3	2	2	1	0,317
80	3	2	2	2	0,73
80	3	2	2	3	0,022
80	4	1	1	1	0,531
80	4	1	1	2	0,575
80	4	1	1	3	0,496
80	4	1	2	1	0,562
80	4	1	2	2	0,573
80	4	1	2	3	1,2

Tablo 7.2.(Devam) Geliştirilen metasezgisel algoritmanın, küçük boyutlu problemlerdeki performansı

n	m	r	p	w	En uygun değerden ortalama % sapma
80	4	2	1	1	0,48
80	4	2	1	2	0,827
80	4	2	1	3	0,409
80	4	2	2	1	0,58
80	4	2	2	2	0,333
80	4	2	2	3	0,636
90	2	1	1	1	0,323
90	2	1	1	2	0,234
90	2	1	1	3	0,226
90	2	1	2	1	0
90	2	1	2	2	0,101
90	2	1	2	3	0
90	2	2	1	1	0
90	2	2	1	2	0,162
90	2	2	1	3	0
90	2	2	2	1	0,083
90	2	2	2	2	0
90	2	2	2	3	0,417
90	3	1	1	1	0,795
90	3	1	1	2	1,1
90	3	1	1	3	0,593
90	3	1	2	1	0,337
90	3	1	2	2	0,537
90	3	1	2	3	0,325
90	3	2	1	1	0,612
90	3	2	1	2	1,016
90	3	2	1	3	0,556
90	3	2	2	1	0,673
90	3	2	2	2	0
90	3	2	2	3	0,263
90	4	1	1	1	1,03
90	4	1	1	2	1,349
90	4	1	1	3	0,472
90	4	1	2	1	0,481
90	4	1	2	2	0,26
90	4	1	2	3	0,436
90	4	2	1	1	0,329
90	4	2	1	2	1,553
90	4	2	1	3	0,9
90	4	2	2	1	1,228
90	4	2	2	2	0

Tablo 7.2.(Devam) Geliştirilen metasezgisel algoritmanın, küçük boyutlu problemlerdeki performansı

n	m	r	p	w	En uygun değerden ortalama % sapma
90	4	2	2	3	1,083
100	2	1	1	1	0,182
100	2	1	1	2	0,445
100	2	1	1	3	0,63
100	2	1	2	1	0,511
100	2	1	2	2	0,222
100	2	1	2	3	0,143
100	2	2	1	1	0
100	2	2	1	2	0
100	2	2	1	3	0
100	2	2	2	1	0
100	2	2	2	2	0,198
100	2	2	2	3	0
100	3	1	1	1	0,783
100	3	1	1	2	1,575
100	3	1	1	3	0,48
100	3	1	2	1	0,233
100	3	1	2	2	0
100	3	1	2	3	0,237
100	3	2	1	1	0,374
100	3	2	1	2	0,513
100	3	2	1	3	0,446
100	3	2	2	1	0,118
100	3	2	2	2	0,707
100	3	2	2	3	0,228
100	4	1	1	1	1,658
100	4	1	1	2	2,976
100	4	1	1	3	1,298
100	4	1	2	1	0,657
100	4	1	2	2	1,38
100	4	1	2	3	0,541
100	4	2	1	1	0,272
100	4	2	1	2	0,616
100	4	2	1	3	1,414
100	4	2	2	1	0,664
100	4	2	2	2	0,179
100	4	2	2	3	0,85

Problemlerin çözümünde elde edilen en büyük sapma %14,286 olmuştur. Bu sapma, en uygun çözüm değeri 7 olan, mevcut makinelerden birisinin çalıştırılması ile elde edilen gelirin makine maliyetinden dolayı oldukça düştüğü bir problemde elde edilmiştir. Önerilen yöntem ise bu problem için çözüm değeri olarak 6 bulmuştur.

Tablo 7.2 incelendiğinde çözümlerde ortalama sapma miktarının %0,3'ten az olduğu görülmektedir. Bu durum geliştirilen algoritmanın gerçek hayat uygulamalarında kullanılabilir olduğunu göstermektedir.

Algoritmanın başarımı ufak problemlerde görüldükten sonra algoritmanın büyük boyutlu problemlerde çalıştırılmasına geçilmiştir. Büyük boyutlu problemler çözülürken, algoritmanın problemler için çözüm süresi 5 dakika veya en az bir yineleme yapılması biçiminde sınırlandırılmıştır. Problem boyutu büyüdüğünde açgözlü sezgisel algoritmanın ürettiği çözüm süresi ve yerel arama süresi oldukça artmaktadır. Bir yineleme için 5 dakikalık süre yetersiz hale gelmektedir.

Geliştirilen metasezgisel algoritma ile elde edilen çözüm değerlerinin kıyaslaması, en uygun çözüm değerleri tamsayılı programlama çözümü bilinmediğinden dolayı, açgözlü sezgisel algoritmanın çözüm değerlerini ne kadar iyileştirdiği ile verilmektedir.

Açgözlü sezgisel algoritma üzerine yapılan iyileşmeler Tablo 7.3 ile verilmektedir. Tablo 7.3 incelendiğinde az sayıda makinenin bulunduğu problemlerde yapılabilen iyileştirme miktarının düşük olduğu görülmektedir. Ancak problemlerdeki makine sayısı arttığında elde edilen çözümlerin açgözlü sezgisel algoritma çözümünü oldukça iyileştirdiği görülmektedir.

Tablo 7.3. Geliştirilen metasezgisel algoritmanın, büyük boyutlu problemlerde açgözlü sezgisel algoritmaya göre yüzde iyileştirme

n	m	r	p	w	Açgözlü sezgisel algoritmaya yapılan yüzde iyileştirme ortalaması
250	2	1	1	1	0,937
250	2	1	1	2	0,985
250	2	1	1	3	1,728
250	2	1	2	1	0,657
250	2	1	2	2	4,122
250	2	1	2	3	3,882
250	2	2	1	1	4,002
250	2	2	1	2	1,543
250	2	2	1	3	1,954

Tablo 7.3.(Devam) Geliştirilen metasezgisel algoritmanın, büyük boyutlu problemlerde açgözlü sezgisel algoritmaya göre yüzde iyileştirme

n	m	r	p	w	Açgözlü sezgisel algoritmaya yapılan yüzde iyileştirme ortalaması
250	2	2	2	1	1,518
250	2	2	2	2	1,172
250	2	2	2	3	3,714
250	20	1	1	1	45,2
250	20	1	1	2	19,762
250	20	1	1	3	19,984
250	20	1	2	1	6,181
250	20	1	2	2	19,092
250	20	1	2	3	12,444
250	20	2	1	1	40,507
250	20	2	1	2	32,092
250	20	2	1	3	16,676
250	20	2	2	1	22,421
250	20	2	2	2	22,882
250	20	2	2	3	19,121
250	50	1	1	1	91,042
250	50	1	1	2	45,358
250	50	1	1	3	22,312
250	50	1	2	1	33,62
250	50	1	2	2	21,634
250	50	1	2	3	26,123
250	50	2	1	1	53,141
250	50	2	1	2	30,434
250	50	2	1	3	14,931
250	50	2	2	1	44,712
250	50	2	2	2	19,017
250	50	2	2	3	22,969
250	75	1	1	1	83,182
250	75	1	1	2	43,987
250	75	1	1	3	7,819
250	75	1	2	1	65,196
250	75	1	2	2	13,774
250	75	1	2	3	21,324
250	75	2	1	1	39,874
250	75	2	1	2	25,9
250	75	2	1	3	6,336
250	75	2	2	1	48,552
250	75	2	2	2	28,404
250	75	2	2	3	21,269
250	100	1	1	1	86,233

Tablo 7.3.(Devam) Geliştirilen metasezgisel algoritmanın, büyük boyutlu problemlerde açgözlü sezgisel algoritmaya göre yüzde iyileştirme

n	m	r	p	w	Açgözlü sezgisel algoritmaya yapılan yüzde iyileştirme ortalaması
250	100	1	1	2	46,785
250	100	1	1	3	11,007
250	100	1	2	1	81,533
250	100	1	2	2	9,134
250	100	1	2	3	29,395
250	100	2	1	1	40,988
250	100	2	1	2	34,604
250	100	2	1	3	9,03
250	100	2	2	1	45,334
250	100	2	2	2	24,905
250	100	2	2	3	16,421
500	2	1	1	1	0
500	2	1	1	2	1,821
500	2	1	1	3	0,976
500	2	1	2	1	0
500	2	1	2	2	2,961
500	2	1	2	3	2,634
500	2	2	1	1	2,012
500	2	2	1	2	1,13
500	2	2	1	3	1,589
500	2	2	2	1	0,395
500	2	2	2	2	3,568
500	2	2	2	3	2,665
500	20	1	1	1	8,775
500	20	1	1	2	1,629
500	20	1	1	3	3,434
500	20	1	2	1	0,359
500	20	1	2	2	6,868
500	20	1	2	3	9,53
500	20	2	1	1	27,687
500	20	2	1	2	19,253
500	20	2	1	3	11,373
500	20	2	2	1	6,75
500	20	2	2	2	11,861
500	20	2	2	3	8,814
500	50	1	1	1	45,846
500	50	1	1	2	18,048
500	50	1	1	3	17,801
500	50	1	2	1	3,943
500	50	1	2	2	3,759

Tablo 7.3.(Devam) Geliştirilen metasezgisel algoritmanın, büyük boyutlu problemlerde açgözlü sezgisel algoritmaya göre yüzde iyileştirme

n	m	r	p	w	Açgözlü sezgisel algoritmaya yapılan yüzde iyileştirme ortalaması
500	50	1	1	3	17,801
500	50	1	2	1	3,943
500	50	1	2	2	3,759
500	50	1	2	3	5,315
500	50	2	1	1	55,728
500	50	2	1	2	17,25
500	50	2	1	3	6,838
500	50	2	2	1	30,628
500	50	2	2	2	6,914
500	50	2	2	3	10,354
500	75	1	1	1	55,286
500	75	1	1	2	30,693
500	75	1	1	3	19,043
500	75	1	2	1	10,86
500	75	1	2	2	1,084
500	75	1	2	3	3,153
500	75	2	1	1	59,768
500	75	2	1	2	18,747
500	75	2	1	3	6,896
500	75	2	2	1	23,527
500	75	2	2	2	9,838
500	75	2	2	3	12,271
500	100	1	1	1	61,534
500	100	1	1	2	29,386
500	100	1	1	3	15,69
500	100	1	2	1	19,412
500	100	1	2	2	5,166
500	100	1	2	3	9,303
500	100	2	1	1	55,609
500	100	2	1	2	17,397
500	100	2	1	3	2,25
500	100	2	2	1	12,555
500	100	2	2	2	11,638
500	100	2	2	3	9,771
1000	2	1	1	1	0
1000	2	1	1	2	1,323
1000	2	1	1	3	0,742
1000	2	1	2	1	0
1000	2	1	2	2	3,118
1000	2	1	2	3	1,589

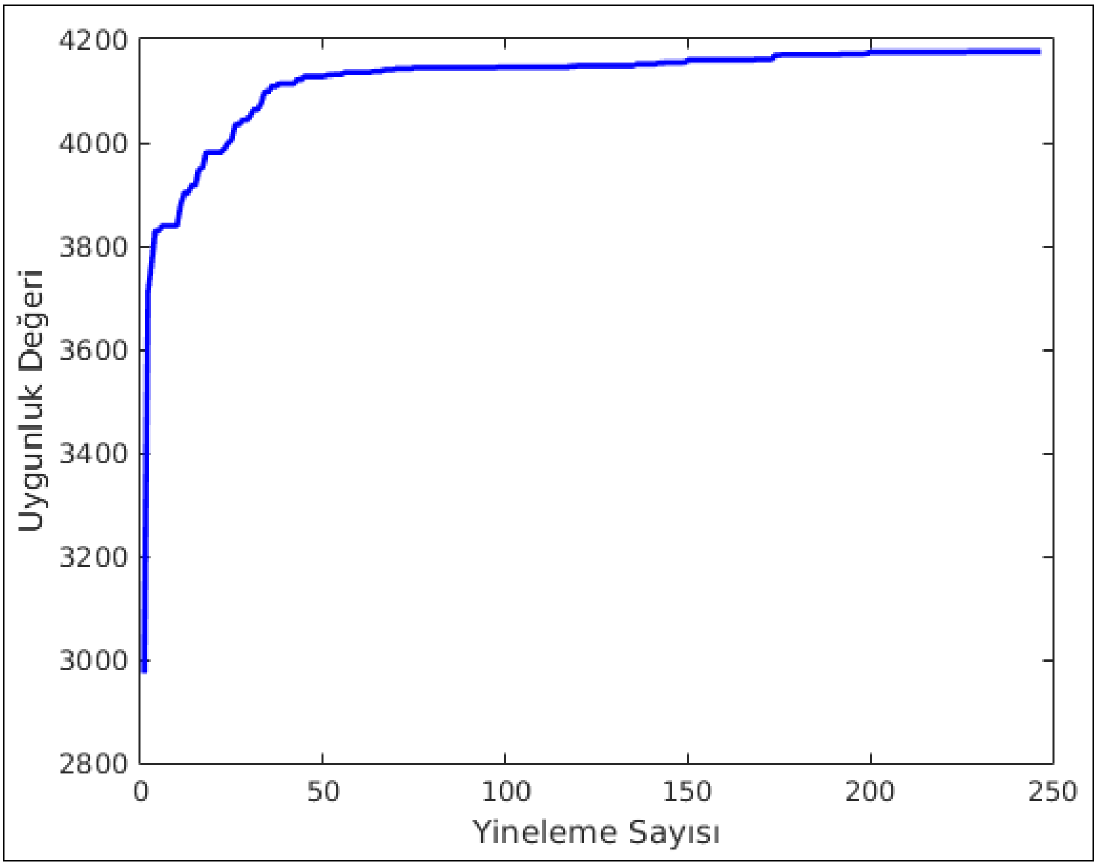
Tablo 7.3.(Devam) Geliştirilen metasezgisel algoritmanın, büyük boyutlu problemlerde açgözlü sezgisel algoritmaya göre yüzde iyileştirme

n	m	r	p	w	Açgözlü sezgisel algoritmaya yapılan yüzde iyileştirme ortalaması
1000	2	2	1	1	0
1000	2	2	1	2	1,165
1000	2	2	1	3	0,715
1000	2	2	2	1	0
1000	2	2	2	2	3,327
1000	2	2	2	3	1,132
1000	20	1	1	1	0,057
1000	20	1	1	2	0,129
1000	20	1	1	3	0,022
1000	20	1	2	1	0
1000	20	1	2	2	0,3
1000	20	1	2	3	1,313
1000	20	2	1	1	3,146
1000	20	2	1	2	0,03
1000	20	2	1	3	0,129
1000	20	2	2	1	0,451
1000	20	2	2	2	0,521
1000	20	2	2	3	1,489
1000	50	1	1	1	5,172
1000	50	1	1	2	0,124
1000	50	1	1	3	0,06
1000	50	1	2	1	0,006
1000	50	1	2	2	0,167
1000	50	1	2	3	0,077
1000	50	2	1	1	0,6
1000	50	2	1	2	1,796
1000	50	2	1	3	0,428
1000	50	2	2	1	0,409
1000	50	2	2	2	0,436
1000	50	2	2	3	0,342
1000	75	1	1	1	8,959
1000	75	1	1	2	0,798
1000	75	1	1	3	4,039
1000	75	1	2	1	0,029
1000	75	1	2	2	0,272
1000	75	1	2	3	0,156
1000	75	2	1	1	0,308
1000	75	2	1	2	1,309
1000	75	2	1	3	0,085
1000	75	2	2	1	4,665

Tablo 7.3.(Devam) Geliştirilen metasezgisel algoritmanın, büyük boyutlu problemlerde açgözlü sezgisel algoritmaya göre yüzde iyileştirme

n	M	r	p	w	Açgözlü sezgisel algoritmaya yapılan yüzde iyileştirme ortalaması
1000	75	2	2	2	0,458
1000	75	2	2	3	0,115
1000	100	1	1	1	0,255
1000	100	1	1	2	3,619
1000	100	1	1	3	6,769
1000	100	1	2	1	0,071
1000	100	1	2	2	0,314
1000	100	1	2	3	0,246
1000	100	2	1	1	0,731
1000	100	2	1	2	0,697
1000	100	2	1	3	0,097
1000	100	2	2	1	7,052
1000	100	2	2	2	0,572
1000	100	2	2	3	0,253

Büyük problemlerde tek bir yineleme yapılabilmiş olmasına rağmen, çözümlerde iyileşme sağlandığı, en büyük problemlerde ise iyileşmenin etkisinin düştüğü görülmektedir. En büyük problemlerde karşılaşılan durum için tek bir problem seçilerek zaman sınırı 480 dakikaya(8 saate) çıkartılarak durum incelenmiştir. Elde edilen sonuçlar, algoritmaya yeterli zamanın verilmediğini, problem büyüklüğü ile zaman sınırının arttırılması gerektiğini göstermiştir. Rastgele seçilen 1000 iş 75 makine problemi için elde edilen iyileşme grafiği Şekil 7.1 ile verilmektedir.



Şekil 7.1. 1000 iş 75 makine problemi için uzun süreli iyileşme grafiği

Algoritmanın büyük problemlerde daha uzun süreler çalıştırılması durumunda iyileşme sağlayabildiği, %30'dan daha büyük miktarda iyileşmelerin elde edilebildiği görülmüştür.

8. SONUÇLAR VE ÖNERİLER

Çalışmada yayılma zamanlı operasyonel sabit iş çizelgeleme problemleri için, polinom zamanda çözülebilen bir gevşetme modeli olan yarı tanımlı programlama gevşetme modeli verilmiş ve problem için üst sınır bulmakta etkili olduğu gösterilmiştir. Ancak yarı tanımlı programlama modelleri teorik olarak polinom zamanda çözülebiliyor olmasına rağmen ele alınan yarı tanımlı programlama modeline günümünüz teknolojisinde kullanılan çözücüler ile anlamlı sürelerde belirli bir büyüklüğe kadar çözüm elde edilebilmiştir. Önerilen modelin etkinliğinin gelecekte daha etkin çözücülerin geliştirilmesi ile daha anlamlı olacağı görülmektedir.

Literatürde tanımlanmış olan yayılma zamanlı operasyonel sabit iş çizelgeleme problemlerinde bulunan kabullerin bir kısmı kaldırılmaya çalışılmış ve literatürde görülmeyen yayılma zamanlı genelleştirilmiş operasyonel sabit iş çizelgeleme problemi tanımlanmıştır. Kaldırılan kabullerin tamamı makineler ile ilgilidir. Makine bağımlı yayılma zamanı kısıtları, makine bağımlı iş getirileri, makine maliyetleri ve makine uyumsuzlukları ilgili kısıtlar probleme eklenmiştir. Tanımlanan genelleştirilmiş problem için tamsayılı doğrusal programlama modeli verilmiş, tamsayılı doğrusal programlama modelinin makul sürelerde çözebildiği problem büyüklüğü - 250 iş 4 makine olarak - belirlenmiştir. Daha büyük problemlerin çözümü için ise bir açgözlü sezgisel algoritma ve bir hibrit metasezgisel algoritma önerilmiştir.

Önerilen hibrit metasezgisel algoritmanın bazı parametreleri sabit kabul edilmiş, sabit kabul edilmemiş parametrelerinin belirlenmesi için de literatürde kullanılan f-race algoritması uygulanmıştır. Küçük boyutlu problemlerde f-race algoritması, metasezgisel yöntemi 45 saniye süre sınırı ile çalıştırarak çözümler elde etmiş ve bu çözümlerden, sabitlenmemiş olan parametrelerin belirlenmesi sağlanmıştır. F-race algoritmasından elde edilen parametreler ile metasezgisel algoritma büyük problemlerde 5 dakika sınırı ile çalıştırılmıştır. Büyük problemlerde elde edilen çözüm değerleri açgözlü sezgisel algoritmanın elde ettiği çözüm değerleri ile karşılaştırılmış, algoritmanın iyileştirme sağlamış olduğu gösterilmiştir.

Ayrıca önerilen metasezgisel algoritmanın 5 dakika sınırı ile büyük problemlerde ancak birkaç yineleme boyunca çalışabildiği görülmüştür. Algoritma performansının, algoritmanın daha fazla yineleme yapabildiği durumda analiz edilebilmesi için zaman sınırı arttırılarak büyük bir problemde çalıştırılmıştır. Çalıştırma sonunda algoritmanın başlangıç çözüme göre iyileşme sağladığı gösterilmiştir.

Problemin gerçek yaşam uygulamalarında kullanılabileceği, işletmelerin benzer problemler üzerinde çalışmalar yaparak süreçlerini iyileştirmeleri, daha az maliyet ile daha fazla gelir elde edebilmeleri mümkün görülmektedir.

KAYNAKLAR

Azimli A., *Matematiksel Optimizasyon*, 1. Basım, Papatya Yayıncılık Eğitim, İstanbul, 2011.

Alabsi F., Naoum R., Comparison of Selection Methods and Crossover Operations using Steady State Genetic Based Intrusion Detection System, *Journal of Emerging Trends in Computing and Information Sciences*, 2012, **3**, 1053-1058.

Ast J. V., Babuska R., Schutter B. D., Particle swarms in optimization and control, *Proceedings of the 17th IFAC World Congress*,Seoul, Korea, 6-11 July 2008.

Bazaraa S. B., Sherali H. D., Shetty C. M., *Nonlinear Programming Theory and Algorithms*, 3rd ed., John Wiley & Sons, New Jersey, 2006.

Birattari M., Stützle T., Paquete L., Varrentrapp K., A racing algorithm for configuring metaheuristics, *Proceedings of the Genetic and Evolutionary Computation Conference*, New York, USA, 9-13 July 2002.

Blokland K., Solution Approaches for Solving Stochastic Job Shop and Blocking Job Shop Problems, Master Of Science Graduation Thesis, Utrecht University, Utrecht, 2012, ICA-3235777.

Boyd S., Vandenberghe L., *Convex Optimization*, 7th ed., Cambridge University Press, New York, 2009.

Chudasama C., Shah S. M., Panchal M., Comparison of Parents Selection Methods of Genetic Algorithm for TSP, *Proceedings published by International Journal of Computer Applications (IJCA)*, 2011, **1**, 85-87.

Coello C. A. C., Pulido G. T., Lechuga M. S., Handling Multiple Objectives With Particle Swarm Optimization, *IEEE Transactions on Evolutionary Computation*, 2004, **8**, 256-279.

Cura T., *Modern Sezgisel Teknikler ve Uygulamaları*, 1. Basım, Papatya Yayıncılık Eğitim, İstanbul, 2008.

Dorigo M., Gambardella L. M., Ant colonies for the traveling salesman problem, *BioSystems*, 1997, **43**, 73-81.

Eliiyi D. T., Azizoğlu M., Heuristics for operational fixed job scheduling problems with working and spread time constraints, *International Journal of Production Economics*, 2011, **132**, 107-121.

Eliiyi D. T., Azizoğlu M., Spread time considerations in operational fixed job scheduling, *International Journal of Production Research*, 2006, **44**, 4343-4365.

Eliiyi D. T., Örnek A., Karakütük S.S., A vehicle scheduling problem with fixed trips and time limitations, *International Journal of Production Economics*, 2009, **117**, 150-161.

Geetha T., Muthukumaran K., An Observational Analysis of Genetic Operators, *International Journal of Computer Applications*, 2013, **63**, 24-34.

Goemans M. X., Williamson D. P., Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming, *Journal of the Association for Computing Machinery*, 1995, **42**,.1115-1145.

Guanlong D, Zhenhao X., Xingsheng G., A Discrete Artificial Bee Colony Algorithm for Minimizing the Total Flow Time in the Blocking Flow Shop Scheduling, *Chinese Journal of Chemical Engineering*, 2012, **20**, 1067-1073.

Hlaing Z. C. S. S., Khine M. A., Solving Traveling Salesman Problem by Using Improved Ant Colony Optimization Algorithm, *International Journal of Information and Education Technology*, 2011, **1**, 404-409.

Karaboğa D., Baştürk B., A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, *J. Glob. Optim.*, 2007, **39**, 459-471.

Karaboğa D., Ökdem S., A Simple and Global Optimization Algorithm for Engineering Problems: Differential Evolution Algorithm, *Turk. J. Elec. Engin.*, 2004, **12**, 53-60.

Karaboğa D., *Yapay Zeka Optimizasyon Algoritmaları*, 3. Basım, Nobel Akademik Yayıncılık Eğitim Danışmanlık, Ankara, 2014.

Karaboğa D., Öztürk C., Fuzzy clustering with artificial bee colony algorithm, *Scientific Research and Essays*, 2010, **5**, 1899-1902.

Kaya S., Engin O., Sabit iş çizelgeleme problemleri: Literatür araştırması ve meta sezgisel yöntemler ile çözüm önerisi, *itüdergisi/d mühendislik*, 2009, **8**, 37-47.

Laurent M., Rendl F., Semidefinite Programming and Integer Programming, *CWI*, PNA-R0210, 2002.

Megow N., Uetz M., Vredeveld T., Models and Algorithms for Stochastic Online Scheduling, *Mathematics of Operations Research*, 2006, **31**, 513–525.

Mendes J. M., A Comparative Study of Crossover Operators for Genetic Algorithms to Solve the Job Shop Scheduling Problem, *WSEAS Transactions on Computers*, 2013, **4**, 164-173.

Nannen V., Eiben A. E., Relevance Estimation and Value Calibration of Evolutionary Algorithm Parameters, *IJCAI-07 Proceedings of the 20th international joint conference on Artifical intelligence*, Hyderabad, India, 6-12 January 2007.

Otman A., Jaafar A., A Comparative Study of Adaptive Crossover Operators for Genetic Algorithms to Resolve the Traveling Salesman Problem, *International Journal of Computer Applications,* 2011, **31**, 49-57.

Özsağlam M. Y., Çunkaş M., Optimizasyon Problemlerinin Çözümü için Parçaçık Sürü Optimizasyonu Algoritması, *Politeknik Dergisi*, 2008, **11**, 299-305.

Özsoydan F. B., Saraç T., A Discrete Particle Swarm Optimization Algorithm for Bicriteria Warehouse Location Problem, *Ekonometri ve İstatistik*, 2011, **13**, 114–124.

Özkale C., Fığlalı A., Çok-Amaçlı Karınca Kolonisi Algoritma Performanslarının İki Kriterli Karesel Atama Problemlerinde Değerlendirilmesi, *XI. Üretim Araştırmaları Sempozyumu*, İstanbul, Türkiye, 23-24 Haziran 2011.

Pandey S., Kumar S., Enhanced Artificial Bee Colony Algorithm and It's Application to Travelling Salesman Problem, *HCTL Open Int. J. of Technology Innovations and Research*, 2013, **2**, 137-146.

Pinedo M., *Scheduling: Theory, Algorithms, and Systems*, 2nd ed., Prentice Hall, New Jersey, 2002.

Ponnambalam S. G., Jawahar N., Chandrasekaran S., Discrete Particle Swarm Optimization Algorithm for Flowshop Scheduling, Editor: Lazinica A., *Particle Swarm Optimization*, 1st ed., Intech, Croatia, 397-422, 2009.

Rajendran A., Ziegler H., Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs, *European Journal of Operational Research*, 2004, **155**, 426-438.

Rapaic M. R., Kanovic Z., Jelicic Z. D., Discrete Particle Swarm Optimization Algorithm for Solving Optimal Sensor Deployment Problem, *Journal of Automatic Control*, 2008, **18**, 9-14.

Razali N. M., Geraghty J., Genetic Algorithm Performance with Different Selection Strategies in Solving TSP, *Proceedings of the World Congress on Engineering*, London, U.K., 6-8 July 2011.

Reddy M. J., Kumar D. N., Multi-objective particle swarm optimization for generating optimal trade-offs in reservoir operation, *Hydrological Processes*, 2007, **21**, 2897-2909.

Rossi A., Singh A., Sevaux M., A metaheuristic for the fixed job scheduling problem under spread time constraints, *Computers and Operations Research*, 2010, **37**, 1045-1054.

Sahu S. K., Pandey M., Hybrid Ant System Algorithm for Solving Quadratic Assignment Problems, *International Journal of Computer Science and Information Technologies*, 2014, **5**, 5950-5956.

Schmitt L. M., Theory of genetic algorithms, *Theoretical Computer Science*, 2001, **259**, 1-61.

Shamekhi A., An Improved Differential Evolution Optimization Algorithm, *IJRRAS*, 2013, **15**, 132-145.

Sharma P., Wadwha A., Komal, Analysis of Selection Schemes for Solving an Optimization Problem in Genetic Algorithm, *International Journal of Computer Applications*, 2014, **93**, 1-3.

Shekhawat A., Poddar P., Boswal D., Ant colony Optimization Algorithms: Introduction and Beyond, Indian Institute of Technology Bombay, http://mat.uab.cat/~alseda/MasterOpt/ACO_Intro.pdf (Ziyaret tarihi: 24 Temmuz 2015).

Solyalı O, Özpeynirci Ö., Operational fixed job scheduling problem under spread time constraints: a branch-and-price algorithm, *International Journal of Production Research*, 2009, **47**, 1877-1893.

Storn R., Price K., Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, *Journal of Global Optimization*, 1997, **11**, 341-359.

Sundar S., Singh A., Rossi A., An Artificial Bee Colony Algorithm for the 0-1 Multidimensional Knapsack Problem, *Communications in Computer and Information Science*, Noida, India, 9-11 August 2010.

Yağmahan B., Yenisey M. M., A multi-objective ant colony system algorithm for flow shop scheduling problem, *Expert Systems with Applications*, 2010, **37**, 1361–1368.

Yan X., Zhang C., Luo W., Li W., Chen W., Liu H., Solve Traveling Salesman Problem Using Particle Swarm Optimization Algorithm, *International Journal of Computer Science Issues*, 2012, **9**, 264-271.

Zhang Y., Wu L., Wang S., Magnetic Resonance Brain Image Classification by an Improved Artificial Bee Colony Algorithm, *Progress In Electromagnetics Research*, 2011, **116**, 65-79.

KİŞİSEL YAYIN VE ESERLER

Cihan A., Fığlalı N., Fığlalı A., Yayılma Zamanlı Operasyonel Sabit İş Çizelgeleme Problemlerinde Polinom Zamanlı Gevşetme Modellerinin Kıyaslanması, *34. Ulusal Yöneylem Araştırması ve Endüstri Mühendisliği Kongresi*, Bursa, Türkiye, 25-27 Haziran 2014.

ÖZGEÇMİŞ

1983 yılında Ankara'da doğdu. İlkokul öğrenimini Ankara'da ortaokul ve lise öğrenimini İstanbul'da tamamladı. 2002 senesinde girdiği Kocaeli Üniversitesi Endüstri Mühendisliği bölümünden, 2007 yılında mezun oldu. 2007 yılında girdiği Kocaeli Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği anabilim dalından 2009 senesinde mezun olarak yine Kocaeli Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği anabilim dalında doktora öğrenimine başladı. Çalışma alanı matematiksel programlama ve metasezgisel yöntemler olup C, C# ve matlab programlama dillerini bilmektedir. İngilizce bilmekte ve boş zamanlarında amatör telsizcilik ile uğraşmaktadır.