c

Contents lists available at *Dergipark*

Journal of Scientific Reports-B

journal homepage: *https://dergipark.org.tr/en/pub/jsrb*

E-ISSN: 2717-8625 **Sayı(Number) 8, Aralık(December) 2023**

ARAŞTIRMA MAKALESİ/RESEARCH ARTICLE

Geliş Tarihi(Receive Date): 13.10.2023 *Kabul Tarihi(Accepted Date): 12.12.2023*

İşlerin Bölünebildiği Esnek Atölye Çizelgeme Problemi için İki Matsezgisel Algoritmanın Karşılaştırılması

Büşra TUTUMLU[1]*, Tuğba SARAÇ[2]

[a]*Kütahya Dumlupınar Üniversitesi, Endüstri Mühendisliği Bölümü, Kütahya, Türkiye*
[b]*Eskişehir Osmangazi Üniversitesi, Endüstri Mühendisliği Bölümü, Eskişehir, Türkiye*

Özet

Esnek atölye çizelgeme problemlerinde (EAÇP) işlerin alt partilere bölünerek farklı makinelerde gerçekleştirilmesi, işletmelerin müşteri taleplerini daha hızlı bir şekilde karşılamasını ve makinelerin de daha verimli kullanılmasını sağlamaktadır. Bu çalışmada, ele alınan problem işlerin bölünebildiği EAÇP'dir. Amaç, son işin tamamlanma zamanının enküçüklenmesidir. Problemin çözümü için hem matsezgisel tavlama benzetimi algoritması (MTB) hem de matsezgisel değişken komşuluk arama algoritması (MDKA) önerilmiştir. Ele alınan problemde işlerin hangi makinelere atanacağı, hangi sırada işleneceği ve alt parti büyüklüklerinin ne olacağının belirlenmesi gerekmektedir. Önerilen algoritmalarda alt parti büyüklüklerinin ne olacağı matematiksel model ile belirlenmektedir. Böylelikle sezgisel algoritmaların hızlı bir şekilde çözüm uzayında arama yapması avantajı ile matematiksel modellerin alt problemlerdeki en iyi çözümü elde etmesi avantajı bir araya getirilmiştir. Önerilen algoritmaların performansını gösterebilmek için rassal türetilen test problemleri ve literatürden alınan bir matematiksel model kullanılmıştır. Ayrıca MTB'nin ve MDKA'nın performansları da kıyaslanmıştır.

Anahtar Kelimeler: İş Bölünmesi; Esnek Atölye Çizelgeleme; Matsezgisel Algoritma; Tavlama Benzetimi; Değişken Komşuluk Arama

* Corresponding author. Tel.: +0-000-000-0000 ; fax: +0-000-000-0000 .
E-mail address: author@institute.xxx
http://dx.doi.org/10.1016/j.cviu.2017.00.000

.

Comparison of two matheuristic algorithms for the flexible job-shop scheduling problem with lot-streaming

Abstract

In flexible job-shop scheduling problems (FJSP), splitting into sub-lots of jobs and performing them on different machines enables businesses to satisfy customer demands more quickly and to use their machines more efficiently. In this study, the FJSP with lot-streaming is discussed. The objective function is to minimize the makespan. To solve the considered problem, matheuristic simulated annealing algorithm (MSA) and matheuristic variable neighborhood search algorithm (MVNS) are proposed. In the problem addressed, it is necessary to determine which machines the jobs will be assigned to, in which sequence they will be processed, and what the sub-lot sizes will be. The proposed algorithms determine the sub-lot sizes by the mathematical model. In this way, the advantage of heuristic algorithms to quickly search the solution space and the advantage of mathematical models to obtain the optimum solution in sub-problems are considered together. To show the performance of the proposed algorithms, randomly generated test problems and a mathematical model taken from the literature are used. Additionally, the performances of MSA and MVNS are compared.

Keywords: Lot- Streaming; Flexible Job Shop Scheduling; Matheuristic Algorithm; Simulated Annealing; Variable Neighborhood Search

1. Giriş

Günümüz rekabet ortamında işletmelerin en önemli hedeflerinden biri işlerin zamanında teslim edilmesidir. İşlerin makinelere bölünerek daha kısa sürelerde tamamlanabilmesi mümkündür. Fakat literatürde çizelgeleme problemlerini ele alan çalışmaların çoğunda işlerin bölünemeyeceği varsayılmaktadır. İşlerin bölünmesine izin vermek sadece işlerin tamamlanma zamanının azaltılmasını değil makinelerin de daha verimli kullanılmasını sağlamaktadır. Kısaca, işlerin bölünmesi, makineler üzerindeki malzeme akışının hızını artırmaya ve üretim süresini, döngü süresini ve süreç içi stoğu azaltmaya büyük ölçüde katkıda bulunmaktadır. Bu çalışmada, esnek atölye çizelgeleme probleminde (EAÇP) işlerin bölünmesi ele alınmıştır.

Literatürde son yıllarda işlerin bölünmesine izin verilen EAÇP ile ilgili çalışmaların sayısı artmaktadır. İncelenen literatürde iş bölünmeli çalışmalar alt partilerin sabit ve değişken olmasına göre ve alt parti sayılarının önceden bilinmesine ve bilinmemesine göre sınıflandırılabilmektedir. Alt partilerin sabit olduğu ve alt parti sayılarının ise önceden bilindiği çalışmalara Zhang vd. [3], Lei ve Guo [4], Liu vd. [5], Xu vd. [6], Defersha, Chen [7] ve Rooyani ve Defersha [8] örnek verilebilir. Zhang vd. [3], büyük ölçekli toplu üretimi içeren eşit boyutlu parti bölmeli EAÇP problemi için bir ikili parçacık sürüsü optimizasyon algoritması önermiştir. Lei ve Guo [4], son işin tamamlanma zamanını enküçüklemek için bir yapay arı kolonisi algoritması önermişlerdir. Liu vd. [5], sabit sayıda iş bölme yaklaşımı kullanarak bu parametreyi sistematik olarak değiştirerek, sabit sayıda alt partinin parti akışının performansı üzerindeki etkisini incelemiş ve genetik algoritma (GA) tabanlı bir iş bölme yaklaşımı önermiştir. Xu vd. [6], işleri belirli sayıda partiye ayırmaya izin veren bir evrimsel algoritma önermişlerdir. Defersha ve Chen [7], çok akışlı EAÇP için bir matematiksel model önermişlerdir ve önerilen modeli çözmek için bir GA geliştirmişlerdir. Rooyani ve Defersha [8] çok amaçlı bir matematiksel model önermiştir ve iki aşamalı bir GA geliştirmiştir. Alt parti büyüklüklerinin değişken olduğu ve alt parti sayılarının önceden bilindiği çalışmalara Daneshamooz vd. [9] ve Abderrabi vd. [10] örnek verilebilir. Daneshamooz vd. [9], problemi işlerin makinelere atanması ve makinelerdeki işlerin sırasının belirlemesi olmak üzere iki alt probleme ayırmış ve değişken komşuluk arama algoritmasına dayalı iki algoritma önermiştir. Abderrabi vd. [10], iş bölmeli EAÇP için bir matematiksel model önermiştir.

Novas [11] çalışmasında, alt partilerin sabit olduğu ve alt parti sayılarının ise önceden bilinmediği durumu ele almıştır. Bu çalışmada, bir kısıt programlama (constraint programming) modeli önerilmiştir. Bu yöntem ile hem partilerin bölünmesi hem de işlerin çizelgelenmesi sağlanmıştır. Alt partilerin değişken olduğu ve alt parti sayılarının önceden bilinmediği çalışmalara Bozek ve Werner [12], Fan vd. [13], Li vd. [14] ve Tutumlu ve Saraç [15] örnek

verilebilir. Bozek ve Werner [12], parti akışı ve değişken alt partilerin parti boyutlandırılması ile EAÇP için iki aşamalı bir çözüm yaklaşımı önermiştir. İlk aşamada, tamamlama süresi enküçüklenmiş ve ikinci aşamada ise, alt parti büyüklükleri enbüyüklenmiştir. Alt partilerin boyutları alt ve üst değerler ile sınırlandırılmıştır. Fan vd. [13], klasik GA'nın benimsendiği bir matsezgisel değişken komşuluk arama algoritması önermiştir. Alt parti sayıları maksimum bölünebilecek parti sayısı ile sınırlandırılmıştır. Li vd. [14], hem son işin tamamlanma zamanının hem de toplam enerji tüketiminin enküçüklenmesini amaçlamıştır ve problemin çözümü için emperyalist rekabet algoritması geliştirmişlerdir. Alt partiler, alt ve üst değerler ile sınırlandırılmıştır. Tutumlu ve Saraç [15], işlerin bölünmesinde alt partilerin boyutunun ve sayısının sınırlandırılmadığı bir matematiksel model önermişlerdir. Büyük boyutlu problemlerin çözümü için parti boyutunun ve sayısının yerel arama algoritması ile belirlendiği melez bir genetik algoritma önerilmiştir.

İşlerin bölünmesine izin verilen EAÇP literatürüne bakıldığında çözüm yöntemi olarak genellikle metasezgisel yöntemlerin kullanıldığı görülmektedir. Erişilen literatürde sadece bir çalışmada [13] çözüm yöntemi olarak matsezgizel algoritma kullanılmıştır. Ele alınan problem türünde matsezgisel algoritmalar nadir kullanılmış olmasına rağmen, bu algoritmalar hem sezgisel yöntemlerin hem de matematiksel modellerin başarılı yönlerinin birleştirilmesiyle ortaya çıktığından büyük bir potansiyele sahiptir. Bu nedenle bu çalışmada, işlerin bölünmesine izin verilen EAÇP problemi için iki farklı matsezgisel algoritma geliştirilmiştir. Bu algoritmalar, matsezgisel tavlama benzetimi ve matsezgisel değişken komşuluk aramadır. Tavlama benzetimi algoritmasının tercih edilmesi, literatürde çoğu problem tipin çok sık kullanılması ve başarılı çözümler elde etmesidir. Değişken komşuluk arama algoritmasının tercih edilmesinin sebebi ise matsezgisel algoritma öneren çalışmada [13] başarıyla kullanılmasıdır. Bu algoritmaların performansının gösterilebilmesi için literatürdeki hiçbir sınırlandırma olmadan iş bölünmesini ele alan Tutumlu ve Saraç [15]'ın çalışmasındaki matematiksel model kullanılmıştır. Farklı boyutta test problemleri türetilerek algoritmaların başarısı karşılaştırılmıştır.

Bu çalışmanın izleyen bölümünde ele alınan problem açıklanmış ve literatürdeki matematiksel model verilmiştir. Üçüncü bölümde önerilen matsezgisel algoritmalar tanıtılmış ve dördüncü bölümde sayısal sonuçlar verilmiştir. Son bölümde ise sonuç ve öneriler sunulmuştur.

2. Problemin Tanımlanması ve Matematiksel Model

Bu çalışmada ele alınan problemde n adet iş, m adet makinede işlem görmektedir. İşler operasyondan oluşmaktadır. Her işin operasyonları arasında öncelik sırası vardır. Bu sıra rota olarak adlandırılmaktadır ve önceden bilinmektedir. Operasyonların hangi makinede ve hangi sırada gerçekleştirileceği, işlerin rotaları göz önünde bulundurularak belirlenmelidir. Bazı operasyonlar yalnızca belirli makinelerde gerçekleştirilebilirken, bazıları aynı özelliklere sahip makinelerde bölünmeden veya farklı alt gruplar halinde bölünerek birden fazla makinede gerçekleştirilebilmektedir.

Bu çalışmada Tutumlu ve Saraç [15] tarafında önerilen matematiksel model ($Model_{TS}$) kullanılmıştır. Modelde, son işin tamamlanma zamanının en küçüklenmesi amaçlanmıştır.

Modelin varsayımları şu şekildedir:

- Planlama periyodunun başında tüm işler hazırdır.
- Bir makine aynı anda tek bir iş işleyebilir.
- İşlerin önceliği yoktur.
- İşlerin hazırlık süreleri, işlem süreleri ve rotaları önceden bilinmektedir.
- İşlerin hazırlık süreleri, sıra bağımlı değildir.
- İşler birden fazla operasyondan oluşabilir.
- İşler bölünebilir ve farklı alt parti büyüklüklerinde farklı paralel makinelere atanabilir. Alt parti büyüklükleri model tarafından belirlenmektedir.

- Her operasyonun gerçekleştirildiği her makinede hazırlık süresine ihtiyaç vardır. Hazırlık süreleri her makinede farklıdır. Eğer bir iş alt partilere bölünüyorsa, her biri için hazırlık süresi gerekmektedir.

$Model_{TS}$'in indisleri, parametreleri, karar değişkenleri, amaç fonksiyonu ve kısıtları aşağıda verilmiştir.

indisler:
i, k: iş indisi
j, l: operasyon indisi
r, a: makine indisi
q, b: sıra indisi

parametreler:
g_i: *i.* işin parti büyüklüğü
o_i: *i.* işin operasyon sayısı
t_{ijr}: *r.* makinedeki *i.* işin *j.* operasyonunun birim işlem süresi
s_{ijr}: *r.* makinedeki *i.* işin *j.* operasyonunun hazırlık süresi
u_{ijr}: *r.* makinede *i.* işin *j.* operasyonu işlem görüyorsa 1, işlem görmüyorsa 0.
M: yeterince büyük bir pozitif sayı
e: alt parti büyüklüğü için en küçük değer (Bu değer sıfıra eşit olduğunda, alt parti boyutu için bir sınır yoktur.)

karar değişkenleri:
x_{ijrq}: eğer *r.* makinenin *q.* sırasında *i.* işin *j.* operasyonu işleniyorsa 1, işlenmiyorsa 0.
C_{ijr}: *r.* makinede *i.* işin *j.* operasyonunun tamamlanma zamanı
α_{ijr} : *r.* makinede *i.* işin *j.* operasyonunun alt parti büyüklüğü
C_{enb}: son işin tamamlanma zamanı

amaç fonksiyonu:

$$enk\ f = C_{enb} \tag{1}$$

kısıtlar:

$$\sum_r \alpha_{ijr} = g_i \qquad \forall_{i,j\,:\,j\le o_i} \tag{2}$$

$$\alpha_{ijr} \le g_i \sum_q x_{ijrq} \qquad \forall_{i,j,r\,:\,j\le o_i} \tag{3}$$

$$\alpha_{ijr} \ge e \sum_q x_{ijrq} \qquad \forall_{i,j,r\,:\,j\le o_i} \tag{4}$$

$$\sum_q x_{ijrq} \le 1 \qquad \forall_{i,j,r\,:\,j\le o_i} \tag{5}$$

$$\sum_i \sum_{j:j\le o_i} x_{ijrq} \le 1 \qquad \forall_{r,q} \tag{6}$$

$$x_{ijrq} \le u_{ijr} \qquad \forall_{i,j,r,q\,:\,j\le o_i} \tag{7}$$

$$q - b \ge 1 - M\left(2 - x_{ijrq} - x_{ilrb}\right) \qquad \forall_{i,j,l,r,q,b\,:\,j\ne l, j\le o_i, q\ne b} \tag{8}$$

$$\sum_i \sum_{j:\, j \le o_i} x_{ijrq} - \sum_k \sum_{l:\, l \le o_k} x_{klr(q-1)} \le 0 \qquad \forall_{r,q\,:\,q>1} \tag{9}$$

$$C_{ijr} + M\left(1 - x_{ijrq}\right) \ge s_{ijr} + t_{ijr}\alpha_{ijr} \qquad \forall_{i,j,r,q\,:\,q=1,j=1} \tag{10}$$

$$C_{ijr} + M\left(2 - x_{ijrq} - x_{klr(q-1)}\right) \ge C_{klr} + s_{ijr} + t_{ijr}\alpha_{ijr} \qquad \forall_{i,j,k,l,r,q\,:\,q>1,j=1,i\neq k} \tag{11}$$

$$C_{ijr} + M\left(2 - x_{ijrq} - x_{i(j-1)ab}\right) \ge C_{i(j-1)a} + s_{ijr} + t_{ijr}\alpha_{ijr} \qquad \forall_{i,j,r,a,q,b\,:\,q=1,j>1,r\neq a} \tag{12}$$

$$C_{ijr} + M\left(2 - x_{ijrq} - x_{klr(q-1)}\right) \ge C_{klr} + s_{ijr} + t_{ijr}\alpha_{ijr} \qquad \forall_{i,j,k,l,r,q\,:\,q>1,k\neq i,j\le o_i,l\le o_k} \tag{13}$$

$$C_{ijr} + M\left(2 - x_{ijrq} - x_{i(j-1)ab}\right) \ge C_{i(j-1)a} + s_{ijr} + t_{ijr}\alpha_{ijr} \qquad \forall_{i,j,r,q,a,b\,:\,j>1,j\le o_i} \tag{14}$$

$$C_{enb} \ge C_{ijr} \qquad \forall_{i,j,r} \tag{15}$$

$$C_{ijr} \ge 0 \qquad \forall_{i,j,r} \tag{16}$$

$$C_{enb} \ge 0 \tag{17}$$

$$x_{ijrq} \in \{0,1\} \qquad \forall_{i,j,r,q} \tag{18}$$

$$\alpha_{ijr} \ge 0 \ \textit{ve tamsayı} \qquad \forall_{i,j,r} \tag{19}$$

Amaç (Eş. (1)), C_{enb} değerinin enküçüklenmesidir. Eş. (2), *i.* işin alt parti büyüklüklerinin toplamının g_i'ye eşit olmasını sağlamaktadır. Eş. (3) ve (4), α_{ijr} ve x_{ijrq} karar değişkenleri için ilişki kısıtlarıdır. Eş. (5) bir işin bir operasyonunun sadece bir makineye atanmasını sağlamaktadır. Eş. (6), bir makinenin bir sırasına en fazla bir işin atanmasını sağlamaktadır. Eş. (7), *i.* işin *j.* operasyonu *r.* makinede gerçekleştirilemiyorsa ilgili operasyonun *r.* makineye atanmamasını sağlamaktadır. Eş. (8) ve (9), makinelerde işlerin operasyonlarının sıra atlamaksızın işlem görmesini sağlamaktadır. Eş. (10), işlerin ilk operasyonları ilk sırada gerçekleştirildiğinde tamamlanma zamanını hesaplamaktadır. Eş. (11), işlerin ilk operasyonu ikinci ve sonraki sırada gerçekleştiğinde tamamlanma zamanını hesaplamaktadır. Eş. (12), işlerin ikinci ve sonraki operasyonları ilk sırada gerçekleştiğinde tamamlanma zamanını hesaplamaktadır. Eş. (13), işlerin ilk sıra dışındaki operasyonların tamamlanma zamanını hesaplamaktadır. Eş. (14) işlerin ikinci ve sonraki operasyonlarının tamamlanma zamanını hesaplamaktadır. Eş. (15), C_{enb} değerini hesaplayan kısıttır. Eş. (16)-(19) işaret kısıtlarıdır.

3. Önerilen Matsezgisel Algoritmalar

Bu çalışmada, Matsezgisel Tavlama Benzetimi ve Matsezgisel Değişken Komşuluk Arama algoritmaları önerilmiştir. Bu algoritmalarda işlerin bölündükleri makinelerdeki alt parti büyüklüğünün ne olacağı kararı matematiksel model ($Model_{APB}$) ile belirlenmektedir. İzleyen alt bölümlerde algoritmalarda kullanılan çözüm gösterimi, $Model_{APB}$, kontrol işlemi ve algoritmalar detaylı olarak açıklanmıştır.

3.1. Algoritmalarda Kullanılan Çözüm Gösterimi

Bu çalışmada ele alınan problemin kısıtlarına uygun üç satırlı bir çözüm gösterimi yapısı önerilmiştir. Önerilen bu yapı, her işin hangi makineye atandığı ve atandığı makinelerdeki sırası ve parti büyüklüğü bilgilerini göstermektedir. Şekil 1'de önerilen çözüm gösterimine örnek verilmiştir. Gösterimin uzunluğu, her işin operasyonlarının işlenebileceği makine sayısının toplamıdır.

	O_{31}	O_{21}	O_{11}	O_{21}	O_{12}	O_{22}	O_{22}	O_{12}	O_{32}	O_{32}
İş	3	2	1	2	1	2	2	1	3	3
Makine	2	1	2	2	1	2	1	2	1	2
Parti Büyüklüğü	100	13	100	87	70	100	0	30	100	0

Şekil 1. Önerilen çözüm gösterimi.

Şekil 1'de verilen örnekte 2 makine, 3 iş ve her işin 2 operasyonu bulunmaktadır. O_{ij}, *i.* işin *j.* operasyonunu temsil etmektedir. O_{11} ve O_{31} sadece M2 makinesine, diğer operasyonlar her iki makineye de atanabilmektedir. Gösterimin uzunluğu 10'dur. İlk satır işleri göstermektedir. Bu satıra göre işlerin operasyonları belirlenmektedir. Örneğin ilk sütunda 3. iş bulunmaktadır. İlk defa 3. iş olması sebebiyle O_{31} operasyonunu temsil eder. O_{31} operasyonu tek makinede gerçekleştirildiği için ikinci defa tekrarlanan 3. iş O_{32} operasyonunu temsil eder. Bu nedenle dokuzuncu sütundaki 3. iş ise O_{32} operasyonunu ifade etmektedir. İkinci satır, işlerin operasyonlarının hangi makinede işlendiğini göstermektedir. Üçüncü satır, işlerin bölünüp bölünmediğini hem de bölündüğüyse atandığı makinede işlenmesi gereken parti büyüklüğünü (100 br.) göstermektedir. Şekil 1'de O_{22} ve O_{32} operasyonları iki makinede de işlem görebilecekken işler bölünmemiş tek bir makinede parti büyüklüğü kadar işlenmektedirler. O_{12} ve O_{21} operasyonları iki makineye bölünerek işlenmektedir. O_{12}, M1'de 70 br. ve M2'de 30 br. işlenmektedir. O_{21} ise M1'de 13 br. ve M2'de 87 br. işlenmektedir.

3.2. Algoritmalarda başlangıç çözümün oluşturulması

Algoritmaların adımlarını gerçekleştirebilmek için uygun bir çözüm ile başlanması gerekir. Bu çalışmada aşağıdaki adımlar izlenerek başlangıç çözüm oluşturulmuştur.

Adım 1: Satır sayısı 3 olan sütun sayısı ise tüm işlerin operasyonlarının işlenebileceği makine sayısının toplamı ($\sum_i \sum_j MS_{ij}$) olan bir matris oluşturulur. (MS_{ij}: *i.* işin *j.* operasyonunun işlenebileceği makine sayısı)
Adım 2: Matrisin ilk satırı işleri gösterdiğinden her iş numarasının $\sum_j MS_{ij}$ kadar tekrar ettiği bir karışık liste oluşturulur. Bu liste matrisin ilk satırıdır.
Adım 3: Matrisin ikinci satırı makineleri göstermektedir. Bu satırı oluşturabilmek için matrisin ilk satırındaki bilgi kullanılarak hangi işin hangi operasyonu olduğu belirlenir ve bu operasyonun gerçekleştirilebileceği makinelerden biri seçilir. Bu şekilde ikinci satırdaki tüm hücreler tamamlanır.
Adım 4: Matrisin üçüncü satırı işlerin operasyonlarının kaça bölündüğünü ve alt parti büyüklüklerini göstermektedir. Başlangıç çözüm için bu adımda sadece kaça bölündüğü bilgisi belirlenir. Daha sonra algoritmalarda matematiksel model ile alt parti büyüklüğü belirlenir. Bu yüzden bu adımda ilk iki satırın bilgileri kullanılır. Eğer operasyon tek makinede işlem görüyorsa ilgili hücreye 1 değeri atanır. Eğer birden fazla makinede işlem görebiliyorsa bir makineye mutlaka atanmasını sağlamak için rassal olarak atanabileceği herhangi bir makineye 1 değeri atanır. Bu işlem bu satır tamamlanıncaya kadar gerçekleştirilir.

3.3. Önerilen $Model_{APB}$

İşlerin bölünmesine izin verilen çalışmalarda arama uzayı çok büyük olması nedeniyle klasik algoritmaların adımları ile alt parti büyüklüklerini belirlemek çok zordur. Bu nedenle bu çalışmada matsezgisel algoritmaların avantajı kullanarak ele alınan problemde işlerin makinelerdeki alt parti büyüklükleri bir doğrusal model ($Model_{APB}$) ile belirlenmektedir. Ayrıca bu model ile çözümün amaç fonksiyonu değeri de hesaplanmaktadır. Önerilen $Model_{APB}$, $Model_{TS}$ temel alınarak geliştirilmiştir. $Model_{TS}$'deki $\boldsymbol{x_{ijkl}}$ karar değişkeni, $Model_{APB}$'de parametredir. $\boldsymbol{x_{ijrq}}$ değerleri çözüm gösteriminden belirlenir. Çözüm gösteriminin ilk iki satırından *i.* işin *j.* operasyonunun *r.* makinede *q.* sırada işlem gördüğü bilgisi elde edilir ve üçüncü satırdan da o makineye atanıp atanmadığı belirlenir. Yani satırdaki ilgili hücrenin değeri sıfırdan büyük ise $\boldsymbol{x_{ijrq}}$ 1 değerini almaktadır. Sıfır olduğunda ise $\boldsymbol{x_{ijrq}}$ 0 değerini almaktadır.

$Model_{APB}$'nın amaç fonksiyonu ve kısıtları aşağıda verilmiştir.

$Model_{APB}$

amaç fonksiyonu:

$$enk\ f = C_{enb}$$

kısıtlar:
Eş. (2-4)
Eş. (10-17)
Eş. (19)

3.4. Kontrol işlemi

Ele alınan problemde işlerin kaç alt partiye bölüneceğine algoritma ile alt parti büyüklüklerine ise $Model_{APB}$ ile karar verilmektedir. Modeldeki *n* parametresi işler bölünüyorsa alt parti büyüklüğünün olabilecek en küçük değeridir. Bu değer sıfır alınabileceği gibi farklı değerlerde alınabilmektedir. Gerçek hayat problemlerinde bir birim parti büyüklüğüne sahip işin farklı makinede işlem görmesi çokta mantıklı değildir. Bu nedenle gerçek hayat problemlerine uygun olması için *e* parametresinin değeri sıfır alınmamıştır. Ayrıca bu çalışmada bir iş en küçük parti büyüklüğü kadar bölünmüş ise bölünmediği durum da değerlendirilmiş ve hangi durumun amaç fonksiyonu değeri daha başarılı ise o durum dikkate alınmıştır.

Kontrol işleminin adımları aşağıda verilmiştir.

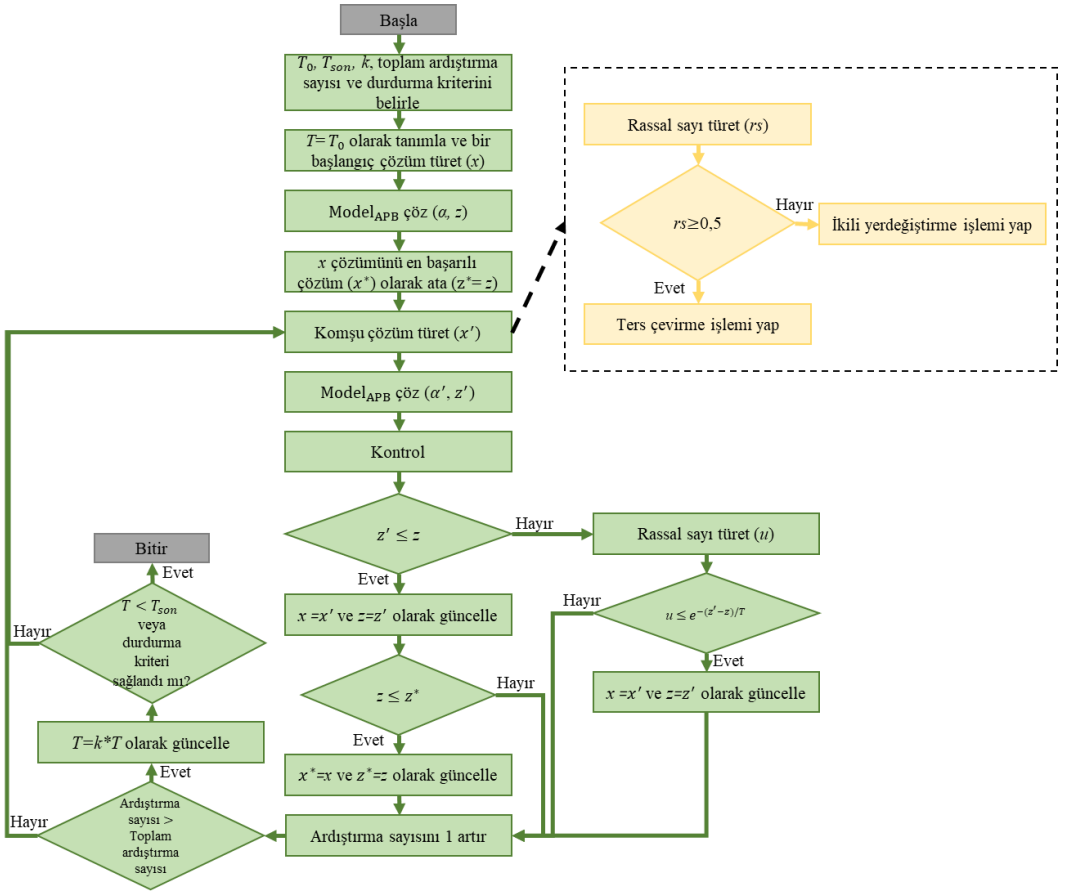
Adım 1: Eldeki çözümün α_{ijr} değerlerini kaydet.
Adım 2: Eğer $\alpha_{ijr} = e$ ise $\sum_q x_{ijrq} = 0$ olarak güncelle.
Adım 3: $\text{Model}_{\text{APB}}$ çöz.
Adım 4: Eldeki çözümden daha iyi bir çözüm bulunduysa eldeki çözümü güncelle.
Adım 5: Her α_{ijr} için *Adım 2*, *Adım 3* ve *Adım 4*'ü tekrarla.

3.5. Matsezgisel tavlama benzetimi

Tavlama Benzetimi (TB), ilk kez Kirkpatrick vd. tarafından 1983 yılında tanıtılmıştır [16]. Bu algoritmada katıların fiziksel tavlanma süreçlerinden ilham alınılmıştır. Bu nedenle doğal süreçlerden ilham alınan ilk algoritmadır. TB, birçok problem türünde uygulanabildiği için yaygın olarak kullanılmaktadır. Yerel en iyiden

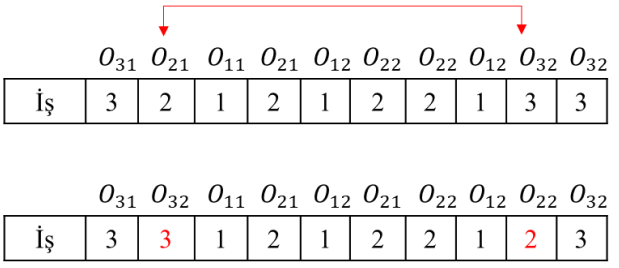
kaçmak için kötü çözümlerin kabul etmesi ve yakınsama özelliği ile kısa sürede büyük boyutlu problemlere çözüm bulabilmektedir. Bir başlangıç sıcaklığı ile başlanır ve her adımda belli bir oranda bu sıcaklık düşürülür. Sıcaklık değeri yüksek iken yeni çözümlerin çoğu kabul edilirken sıcaklık düştükçe yeni çözümlerin kabulü azalarak yakınsama sağlanır. TB'nin parametreleri, başlangıç sıcaklığı (T_0), son sıcaklık değeri (T_{son}), sıcaklığı azaltma oranı (k), her sıcaklıktaki ardıştırma sayısı ve durdurma kriteridir.

Bu çalışmada, önerilen $Model_{APB}$'i TB algoritmasına entegre edilmiştir. Böylelikle $Model_{TS}$'nin belirlenen süre limitleri içerisinde çözüm bulamadığı büyük boyutlu problemler için çözüm bulunmuştur. Ayrıca işlerin makinelerdeki alt parti büyüklükleri model tarafından belirlendiği için başarılı çözümler elde edilmiştir. Önerilen matsezgisel tavlama benzetimi (MTB) algoritmasının adımları akış şeması ile gösterilmiştir. Akış şeması Şekil 2'de verilmiştir. Akış şemasındaki z, amaç fonksiyonun değeri ve α ise işlerin operasyonlarının alt parti büyüklüklerini ifade etmektedir.

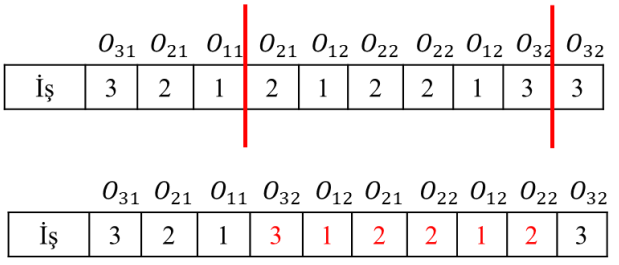


Şekil 2. Önerilen matsezgisel tavlama benzetimi algoritmasının akış şeması.

Bu algoritmada komşu çözüm türetmek için çözüm gösterimindeki iş satırı için ikili yer değiştirme veya ters çevirme operatörleri kullanılmıştır. Şekil 3 ve 4'de operatörlerin uygulaması verilmiştir.



Şekil 3. İkili yerdeğiştirme operatörü.



Şekil 4. Ters çevirme operatörü.

Görüldüğü gibi, Şekil 3'te gösterilen ikili yer değiştirme operatörü için rastgele iki konum seçilir ve bu konumlardaki değerler birbirleriyle değiştirilir. Ters çevirme operatörü için rastgele iki konum seçilir ve bu iki konum arasındaki hücrelerdeki değerler ters çevrilir. Çözüm gösteriminin makine satırı yani ikinci satır ilk satırdaki değişiklikten dolayı tekrar düzenlenir. Üçüncü satır için rastgele [1, *PMO*/3] aralığından bir sayı seçilir. *PMO*, tüm işlerin paralel makinelerde işlem görebilen operasyonlarının toplamıdır.Bu sayı kadar paralel makinelerde işlenebilen işlerin operasyonları seçilir ve her operasyonun işlenebileceği tüm makineler için rassal sayı türetilir. Rassal sayı 0,5'ten büyük ise 1, küçük ise 0 değeri atanır. Diğer işlerin operasyonlarının değerlerinde herhangi bir değişiklik yapılmaz.

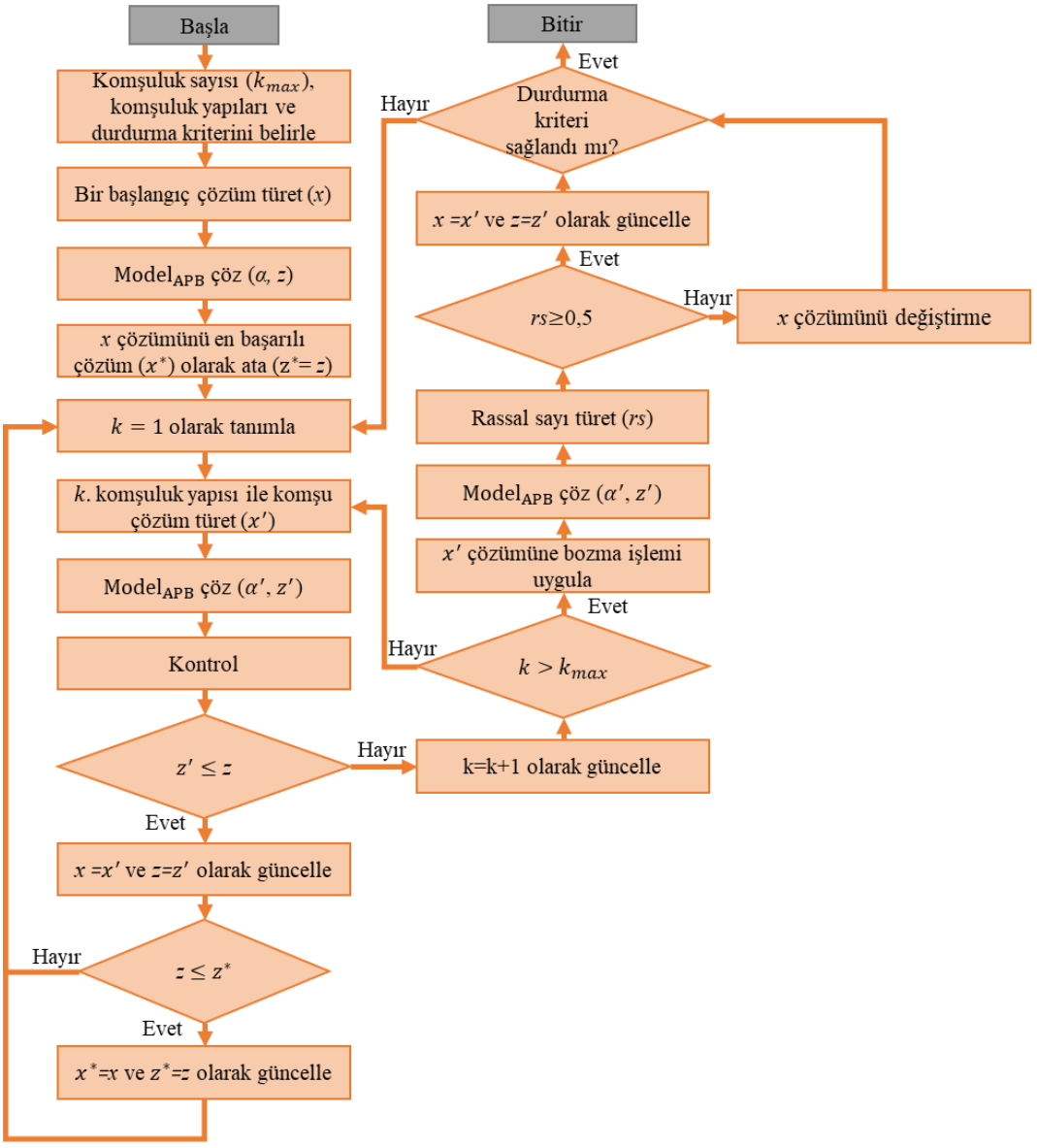
3.6. Matsezgisel değişken komşuluk arama

Değişken komşuluk arama algoritması ilk kez Mladenovic ve Hansen [17] tarafından 1997'de geliştirilmiştir. Tavlama benzetimi gibi tek çözüm tabanlı bir algoritmadır. Diğer sezgisel algoritmalara kıyasla bu algoritma, basit yapısı, farklı çözüm teknikleriyle entegre edilebilmesi ve çok az parametre gerektirmesi nedeniyle diğer algoritmalara göre önemli avantajlar sunmaktadır [18]. Tek bir komşuluk yapısı yerine birden fazla komşuluk yapısı kullanılmaktadır. Bu çalışmada, değişken komşuluk arama algoritmasına da önerilen $Model_{APB}$ entegre edilerek matsezgisel değişken komşuluk arama (MDKA) algoritması önerilmiştir. 5 farklı komşuluk yapısı kullanılmıştır. Bu yapılar aşağıda verilmiştir.

Komşuluk 1 (İkili Yer Değiştirme Operatörü): Çözüm gösterimindeki alt parti sayılarını değiştirmeden işlerin sıralarının değiştirilmesi sağlanmaktadır.

Komşuluk 2 (Ters Çevirme Operatörü): Komşuluk 1'deki gibi çözüm gösterimindeki alt parti sayılarını değiştirmeden işlerin sıralarının değiştirilmesi sağlanmaktadır. Fakat komşuluk 1'e göre bu komşuluk yapısı daha büyük değişikliklere neden olmaktadır.
Komşuluk 3 (Alt Parti Sayısının Değiştirilmesi): Rastgele [1, *PMO*/3] aralığından bir sayı seçilir. Bu sayı kadar paralel makinelerde işlenebilen işlerin operasyonları seçilir ve her operasyonun işlenebileceği tüm makineler için rassal sayı türetilir. Rassal sayı 0,5'ten büyük ise 1, küçük ise 0 değeri atanır.
Komşuluk 4 (İkili Yer Değiştirme Operatörü + Alt Parti Sayısının Değiştirilmesi): Hem Komşuluk 1 hem de Komşuluk 3 yapısındaki değişiklikler birlikte yapılmaktadır.
Komşuluk 5 (Ters Çevirme Operatörü + Alt Parti Sayısının Değiştirilmesi): Hem Komşuluk 2 hem de Komşuluk 3 yapısındaki değişiklikler birlikte yapılmaktadır.

Önerilen MDKA'nın adımları akış şeması ile gösterilmiştir. Akış şeması Şekil 5'de verilmiştir.

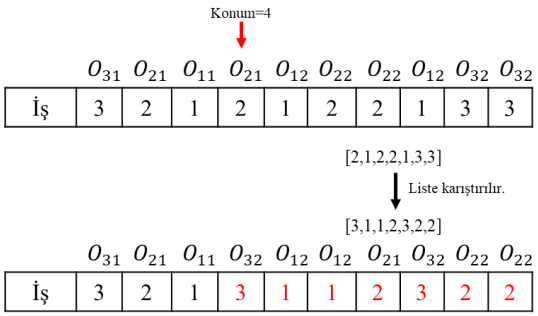


Şekil 5. Önerilen değişken komşuluk arama algoritmasının akış şeması.

Şekil 5'ten de görülebileceği gibi, değişken komşuluk arama algoritmasında komşu çözüme bozma işlemi uygulanmaktadır. Bu işlem, yerel eniyiye takılmamak ve aramayı etkili bir şekilde çeşitlendirmek için gerçekleştirilmektedir. Bu çalışmada bozma işlemi için aşağıdaki adımlar uygulanmaktadır.

Adım 1: Bozma işlemi uygulanacak komşu çözümün ilk satırından rastgele bir konum seçilir. Bu konumdan sonraki işler karıştırılır. Şekil 6'da bu işlemin nasıl gerçekleştirildiği gösterilmiştir.
Adım 2: İkinci satıra tekrar makine atamaları yapılır.

Adım 3: Çözümün bozulmadığı durumdaki işlerin operasyonlarının alt parti sayıları ve büyüklüklerine göre üçüncü satır düzenlenir.



Şekil 6. Çözümün ilk satırını bozma işlemi.

4. Sayısal Sonuçlar

Bu bölümde, önerilen MTB ve MDKA algoritmalarını karşılaştırabilmek için farklı boyutta test problemleri türetilmiştir. Türetilen test problemler önerilen algoritmalar ile Python 3.9.7 ve Gurobi 9.5.0 kullanılarak çözülmüştür. Ayrıca algoritmaların performansını değerlendirebilmek içinde türetilen test problemleri literatürdeki $Model_{TS}$ ile de GAMS 23.7'nin CPLEX çözücüsü kullanılarak çözülmüştür. Tüm test problemleri, 2.40 GHz Intel Core i5 ve 8 GB RAM'e sahip bir bilgisayarda çözülmüştür.

4.1. Test problemlerinin türetilmesi

Önerilen MTB ve MDKA algoritmalarının performansı, rastgele oluşturulmuş küçük, orta ve büyük boyutta problemler çözdürülerek test edilmiştir. Problemlerin özellikleri Tutumlu ve Saraç [15]'ın çalışmasındaki gibi ele alınmıştır. Küçük, orta ve büyük boyutlu problemlerin özellikleri; iş sayısı 4, 6 ve 10, işlerin operasyon sayısı 3, 4 ve 5 ve makine sayısı 5, 8 ve 11'dır. Operasyonların gerçekleştirebileceği alternatif makine sayısı küçük boyutlu problemde 2 iken diğer boyuttaki problemlerde 3'tür. Ayrıca test problemlerinin t_{ijr} $[1/g_i, 99/g_i]$ve s_{ijr} [1,30] aralığında düzgün dağılıma uygun türetilmiştir. e ve g_i sırasıyla 10 ve 100 olarak tanımlanmıştır.

4.2. Önerilen algoritmaların parametreleri

Algoritmaların uygulanabilmesi için bazı parametre değerlerinin belirlenmesi gerekmektedir. MTB için başlangıç sıcaklığı (T_0), son sıcaklık değeri (T_{son}), sıcaklığı azaltma oranı (k), her sıcaklıktaki ardıştırma sayısı ve durdurma kriterine karar verilmesi gerekmektedir. MDKA algoritması için ise komşuluk sayısı ve durdurma kriterine karar verilmesi gerekmektedir. İki algoritmada da durdurma kriteri olarak iterasyon sayısı uygulanmıştır. Tablo 1'de önerilen algoritmaların parametre değerleri verilmiştir.

Tablo 1. Önerilen algoritmaların parametre değerleri.

MTB		*MDKA*	
Parametre	*Değeri*	*Parametre*	*Değeri*
T_0	100	*Komşuluk sayısı*	5
T_{son}	10	*İterasyon sayısı*	10000

k	0,90
Ardıştırma sayısı	5
İterasyon sayısı	500

Tablo 1'den de görülebileceği gibi MTB'de birden fazla parametre değerine karar verilmesi gerekirken, MDKA'da ise daha az sayıda parametre değerine karar verilmesi gerekmektedir. MDKA, sezgisel yöntemler arasında en az parametreye sahip algoritmalardan biridir. İki algoritmanın birbiri ile kıyaslanabilmesi için araştırdıkları çözüm sayılarının dengeli olması gerekir. Parametre değerleri belirlenirken bu durum dikkate alınmıştır. İki algoritmada da yaklaşık 50000 tane komşu çözüm araştırılmaktadır.

4.3. Oyuncak problem

İş sayısının 4, işlerin operasyon sayısının 2, makine sayısının 3 ve operasyonların gerçekleştirebileceği alternatif makine sayısının 2 olduğu bir oyuncak problem türetilmiştir. Problemde işlerin operasyonlarının gerçekleştirilebileceği makinelerdeki (u_{ijr}=1) t_{ijr} ve s_{ijr} süreleri Tablo 2'de verilmiştir.

Tablo 2. Toy problemin t_{ijr} ve s_{ijr} parametre değerleri.

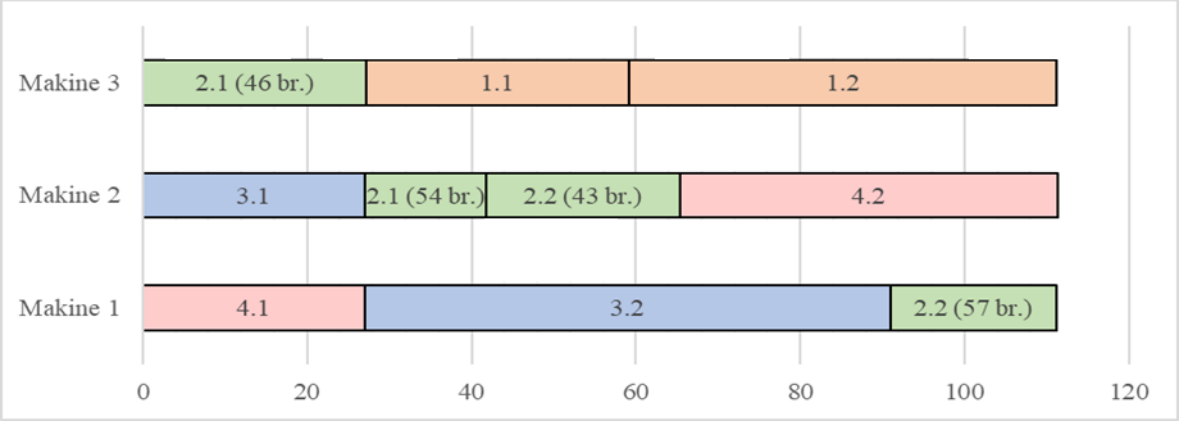
i	j	r	t_{ijr}	s_{ijr}
1	1	1	0,13	19
1	1	3	0,17	15
1	2	3	0,38	14
2	1	2	0,09	10
2	1	3	0,20	18
2	2	1	0,11	14
2	2	2	0,15	17
3	1	2	0,12	15
3	1	3	0,85	28
3	2	1	0,47	17
4	1	1	0,13	14
4	1	3	0,32	28
4	2	2	0,39	7
4	2	3	0,45	24

Oyuncak problem Model_{TS}, MTB ve MDKA ile çözülerek Tablo 3'de sunulan C_{enb} ve çözüm süreleri elde edilmiştir.

Tablo 3. C_{enb} değerleri ve çözüm süreleri.

Yöntem	C_{enb}	*Süre (sn.)*
$Model_{TS}$	111,31	13,23
MTB	111,31	29,83
MDKA	111,31	75,23

Tablo 3'ten de görülebileceği gibi oyuncak problemin en iyi amaç fonksiyonu değeri olan 111,31 hem MTB hem de MDKA ile bulunmuştur. Böylelikle iki algoritmanın da eniyi çözüme ulaşabildiği gösterilmiştir. Şekil 7'de oyuncak problemin Gantt Şeması verilmiştir.



Şekil 7. Oyuncak problemin Gantt Şeması.

Şekil 7 incelendiğinde hem O_{21} hem de O_{22} operasyonları iki alt partiye bölünmüştür. O_{21} operasyonu makine 2'de 54 br. ve makine 3'te 46 br. ve O_{22} operasyonu da makine 1'de 57 br. ve makine 2'de 43 br. işlenmektedir. Ayrıca işlerin alt partilere bölünmesinin makine kullanımlarını da dengelediği görülmektedir.

4.4. Test sonuçları

Rastgele türetilen test problemleri literatürden alınan [15] matematiksel model ($Model_{TS}$) ve melez genetik algoritma (MGA_{TS}) ve önerilen MTB ve MDKA algoritmaları ile çözülmüştür. Çözüm süreleri 7200 saniye ile sınırlıdır. Farklı boyuttaki test problemleri için elde edilen sonuçlar ve çözüm süreleri Tablo 4'de verilmiştir. Tablo 4'de verilen MTB ve MDKA için C_{enb} değerleri 5 kez çalıştırma sonucunda elde edilen en başarılı sonuçlardır.

Tablo 4. Elde edilen sonuçlar ve çözüm süreleri.

Test Problemleri	$Model_{TS}$		MGA_{TS}		*MTB*		*MDKA*	
	C_{enb}	*Süre (sn.)*	C_{enb}	*Süre (sn.)*	C_{enb}	*Süre (sn.)*	C_{enb}	*Süre (sn.)*
Küçük	195,78	483,13	195,78	234,97	195,78	311,12	195,78	413,62
Orta	515,2	7200	285	520,93	283	1635,51	284,96	3015,92
Büyük	-	-	481,04	1486,71	461,72	1718,63	473	5307,41

Tablo 4 incelendiğinde, literatürdeki $Model_{TS}$ ile küçük boyutlu problemde en iyi çözüm ve orta boyutlu problemde süre limiti içerisinde bir uygun çözüm elde edilmiştir. Fakat büyük boyutlu problemde süre limiti içerisinde bir çözüm elde edilememiştir. MGA_{TS}, MTB ve MDKA algoritmaları ile tüm boyutlardaki problemler için bir çözüm bulunmuştur. Hatta küçük boyutlu problemin eniyi çözümüne tüm algoritmalar ile kısa sürede ulaşılmıştır. Önerilen algoritmalar çözüm süreleri açısından karşılaştırıldığında küçük, orta ve büyük boyutlu problemlerde sırasıyla %24,78, %45,77 ve %67,62 oranında MTB, MDKA'dan daha kısa sürede çözüm bulmuştur.

Literatürden alınan ve önerilen çözüm yöntemlerinin başarısı kıyaslanarak iyileştirme yüzdeleri Tablo 5'te verilmiştir. İyileştirme yüzdeleri Denklem (20)'de verilen formül ile hesaplanmıştır. Burada iyileştirme yüzdesi, kıyaslama yapılan ikinci yöntemin amaç fonksiyonunun (z_2) ilk yönteme (z_1) kıyasla yüzdesel olarak ne kadar başarılı olduğu anlamına gelmektedir.

$$\text{İ}yileştirme\ (\%) = \frac{(z_1 - z_2)}{z_1}\ 100 \quad (20)$$

Tablo 5. İyileştirme yüzdeleri.

	İyileştirme (%)				
Test Problemleri	*$Model_{TS}$-MTB*	*$Model_{TS}$-MDKA*	*MGA_{TS}-MTB*	*MGA_{TS}-MDKA*	*MDKA-MTB*
Küçük	0	0	0	0	0
Orta	45,07	44,69	0,70	0,01	0,69
Büyük	-	-	4,02	1,67	2,38

Tablo 5'ten de görülebileceği gibi, küçük boyutlu problemin tüm yöntemler ile eniyi çözümüne ulaşılmıştır. Orta boyutlu test probleminde MTB ve MDKA sırasıyla *$Model_{TS}$*'den %45,07 ve %44,69 daha başarılı sonuç bulmuşlardır. MGA_{TS}'nin amaç fonksiyonu değeri ise sırasıyla %0,7 ve %0,01 oranında iyileştirilmiştir. Orta boyutlu problemde MTB ve MDKA çok yakın sonuçlar elde etmişlerdir. Büyük boyutlu problemde MTB ve MDKA algoritmaları ile sırasıyla MGA_{TS}'den %4,02 ve %1,67 daha başarılı amaç fonksiyonu değeri elde edilmiştir. Ayrıca MTB, MDKA'ya göre %2,38 daha başarılı çözüm bulmuştur.

Kısaca özetlersek, MTB ve MDKA'nın küçük boyutlu problemde *$Model_{TS}$* ile aynı sonucu yani en iyi çözümü bulduğu gösterilmiştir. Önerilen iki algoritma kıyaslandığında ise MTB, MDKA'dan daha kısa süre de daha başarılı çözümler bulmuştur.

5. Sonuçlar ve Öneriler

Bu çalışmada, işlerin bölünebildiği esnek atölye çizelgeleme problemi ele alınmıştır. Bu problemde işlerin kaç alt partiye bölüneceği ve alt parti büyüklüklerinin ne olacağını belirlemek kolay değildir. Bu nedenle genellikle literatürdeki çalışmalarda alt parti büyüklükleri bir sayının katı olarak ele alınmaktadır veya alt ve üst sınırlar ile sınırlandırılmaktadır. Tutumlu ve Saraç [15] hiçbir sınırlandırma olmadan ele alınan problem için bir matematiksel model ve melez bir GA önermişlerdir. Önerilen sezgisel yöntemde alt parti büyüklüklerinin ne olacağını belirlerken mevcut çözümdeki alt parti büyüklükleri artırılarak veya azaltılarak belirlenmektedir. Bu işlem ile geniş arama uzayında kısa sürelerde başarılı çözümler bulmak zor olabilir. Oysaki son yıllarda hem matematiksel modellerin hem de sezgisel yöntemlerin birlikte kullanıldığı matsezgisel algoritmaların kullanımı artmıştır. Matsezgisel algoritmalar sayesinde alt parti büyüklükleri matematiksel model ile belirlenebilmektedir. Böylelikle sezgisel algoritmanın türettiği komşu çözümlerin en iyi alt parti büyüklükleri bulunmaktadır. Bu çalışmada ele alınan problem için hem MTB hem de MDKA algoritmaları önerilmiştir, her iki algoritmada da literatürden farklı olarak işlerin alt parti büyüklükleri, matematiksel model ile bulunarak hem alt problemin eniyi çözümü hem de sezgisel algoritmanın arama yapacağı çözüm uzayı daraltılmıştır. Algoritmaların performansı farklı boyutlarda rastgele türetilen test problemleri ve literatürden alınan matematiksel model kullanılarak gösterilmiştir. Ayrıca önerilen algoritmalarla elde edilen sonuçlar kıyaslanmıştır. Elde edilen sonuçlara göre MTB, MDKA'ya göre kısa sürelerde daha başarılı çözümler elde etmektedir.

Gelecek çalışmalarda, problem çok amaçlı olarak ele alınıp, çok amaçlı matsezgisel yöntemler geliştirilebilir.

Teşekkür

Teşekkür edeceğimiz herhangi bir kişi veya kuruluş bulunmamaktadır.

Kaynaklar

[1] J. W. Stevenson, *Production/Operations Management*, Irwin, 1996.
[2] H. Liu, A. Abraham ve Z. Wang, "A multi-swarm approach to multi-objective flexible job-shop scheduling problems", Fundamenta Informaticae, cilt 95, sayı 4, s. 465-489, 2009, doi: 10.3233/FI-2009-160.
[3] H. P. Zhang, J. H. Ye, X. P. Yang, N. W. Muruve ve J. T. Wang, "Modified binary particle swarm optimization algorithm in lot-splitting scheduling involving multiple techniques", International Journal of Simulation Modelling, cilt 17, sayı 3, s. 534-542, Eylül 2018, doi: 10.2507/IJSIMM17(3)CO13.
[4] D. Lei ve X. Guo, "Scheduling job shop with lot streaming and transportation through a modified artificial bee colony", International Journal of Production Research, cilt 51, sayı 16, s. 4930-4941, Ağustos 2013, doi: 10.1080/00207543.2013.784404.
[5] C. H. Liu, L. S. Chen ve P. S. Lin, "Lot streaming multiple jobs with values exponentially deteriorating over time in a job-shop environment", International Journal of Production Research, cilt 51, sayı 1, s. 202–214, 2013, doi: 10.1080/00207543.2012.657255.
[6] X. L. Xu, L. Li, L. X. Fan, J. Zhang, X. H. Yang ve W. L. Wang, "Hybrid discrete differential evolution algorithm for lot splitting with capacity constraints in flexible job scheduling", Mathematical Problems in Engineering, 2013, doi: 10.1155/2013/986218.
[7] F. Defersha ve M. Chen, "Jobshop lot streaming with routing flexibility, sequence-dependent setups, machine release dates and lag time", International Journal of Production Research, cilt 50, sayı 8, s. 2331–2352, 2012, doi: 10.1080/00207543.2011.574952.
[8] D. Rooyani ve F. Defersha, "A two-stage multi-objective genetic algorithm for a flexible job shop scheduling problem with lot streaming", Algorithms, cilt 15, sayı 7, Temmuz 2022, doi: 10.3390/a15070246.
[9] F. Daneshamooz, P. Fattahi ve S. M. H. Hosseini, "Scheduling in a flexible job shop followed by some parallel assembly stations considering lot streaming", Engineering Optimization, cilt 54, sayı 4, s. 614-633, Nisan 2022, doi: 10.1080/0305215X.2021.1887168.
[10] F. Abderrabi vd., "Flexible job shop scheduling problem with sequence dependent setup time and job splitting: Hospital catering case study", Applied Sciences-Basel, cilt 11, sayı 4, Şubat 2021, doi: 10.3390/app11041504.
[11] J. M. Novas, "Production scheduling and lot streaming at flexible job-shops environments using constraint programming", Computers & Industrial Engineering, cilt 136, s. 252-264, Ekim 2019, doi: 10.1016/j.cie.2019.07.011.
[12] A. Bozek ve F. Werner, "Flexible job shop scheduling with lot streaming and sublot size optimisation", International Journal of Production Research, cilt 56, sayı 19, s. 6391-6411, Ekim 2017, doi: 10.1080/00207543.2017.1346322.
[13] J. X. Fan, C. J. Zhang, W. M. Shen ve L. Gao, "A matheuristic for flexible job shop scheduling problem with lot-streaming and machine reconfigurations", International Journal Of Production Research, cilt 61, sayı 19, s. 6565-6588, Ekim 2022, doi: 10.1080/00207543.2022.2135629.
[14] Y. B. Li, Z. P. Yang, L. Wang, H. T. Tang, L. B. Sun ve S. S. Guo, "A hybrid imperialist competitive algorithm for energy-efficient flexible job shop scheduling problem with variable-size sublots", Computers & Industrial Engineering, cilt 172, Ekim 2022, doi: 10.1016/j.cie.2022.108641.
[15] B. Tutumlu ve T. Saraç, "A MIP model and a hybrid genetic algorithm for flexible job-shop scheduling problem with job-splitting", Computers and Operations Research, cilt 155, Temmuz 2023, doi: 10.1016/j.cor.2023.106222.
[16] W. H. T. Meira vd., "Scheduling of a single-source multiproduct pipeline system by a matheuristic approach: Combining simulated annealing and MILP", Computers & Chemical Engineering, cilt 136, Mayıs 2020, doi: 10.1016/j.compchemeng.2020.106784.
[17] N. Mladenovic ve P. Hansen, "Variable neighborhood search", Pergamon, cilt 24, sayı 11, s. 1097-1100, Kasım 1997, doi: 10.1016/S0305-0548(97)00031-2.
[18] P. Hansen ve N. Mladenovic, "J-Means: a new local search heuristic for minimum sum of squares clustering", Pattern Recognition, cilt 34, sayı 2, s. 405-413, Şubat 2001, doi: 10.1016/S0031-3203(99)00216-2.