ÇOK AMAÇLI PERMÜTASYON AKIŞ TİPİ ÇİZELGELEME PROBLEMİ İÇİN BİR NSGA-II ALGORİTMASI

Tuğba SARAÇ[1*], Nilay BİLGİÇER[2]

[1]Eskişehir Osmangazi Üniversitesi, MMF, Endüstri Mühendisliği Bölümü, Eskişehir
ORCID No : https://orcid.org/0000-0002-8115-3206
[2]Eskişehir Osmangazi Üniversitesi, MMF, Endüstri Mühendisliği Bölümü, Eskişehir
ORCID No : https://orcid.org/0000-0001-7825-1393

Anahtar Kelimeler

Çok amaçlı akış tipi çizelgeleme problemi, Sıra bağımlı hazırlık süreleri, Genetik algoritma, NSGA-II algoritması.

Öz

Bu çalışmada, sıra bağımlı hazırlık sürelerinin olduğu çok amaçlı permütasyon akış tipi çizelgeleme problemi ele alınmıştır. Problemin amaçları, son işin tamamlanma zamanının, toplam gecikmenin ve toplam erken tamamlanma süresinin enküçüklenmesidir. Ele alınan problemin çözümüne yönelik olarak bir genetik algoritma ve problemin çok amaçlı doğası dikkate alınarak bir NSGA-II algoritması önerilmiştir. Ayrıca, literatürde tek makine çizelgeleme problemleri için önerilmiş olan öncelik kurallarından bazıları uyarlanarak, ilk neslin başarısını arttırmakta kullanılmıştır. Önerilen algoritmaların başarısı, rassal türetilen test problemleri kullanılarak gösterilmiştir.

AN NSGA-II ALGORİTHM FOR MULTI-OBJECTIVE PERMUTATION FLOW SHOP SCHEDULING PROBLEM

Keywords

Multi objective flow shop scheduling problem, Sequence dependent setup times, Genetic algorithm, NSGA-II algorithm.

Abstract

In this study, multi objective permutation flow shop scheduling problems with sequence dependent setup times is considered. Objectives of the problem are to minimize the makespan, total tardiness and total earliness. To solve the considered problem, a genetic algorithm and by taking into multi-objective nature of the problem, a NSGA-II algorithm are proposed. Adapting some of dispatching rules, used in literature for single machine scheduling problems, for flow shop scheduling problems, success of the first generation of NSGA-II is increased. Performance of the proposed algorithms is showed by using randomly-generated test problems.

Araştırma Makalesi		Research Article	
Başvuru Tarihi	: 21.09.2019	Submission Date	: 21.09.2019
Kabul Tarihi	: 07.05.2020	Accepted Date	: 07.05.2020

1. Giriş

Her bir işin aynı rotayı izleyerek sırayla makinelerde işlem gördüğü üretim şekline akış tipi üretim denir. Akış tipi üretim, çizelgeleme literatüründe yer alan en önemli problemlerden birisidir. Sanayileşen dünyada rekabetin artmasıyla, ucuz ve kaliteli üretim yapmak, tam zamanlı ve sağlıklı bir şekilde ürünleri sevk ve idare etmek vazgeçilmez bir gereklilik halini almıştır. Pek çok firma bu rekabet gücünü elde edebilmek amacıyla değişkenlikleri azaltma, işleri standartlaştırma ve hızlı üretim yapma yolunu seçmiştir. Bu gereksinim, akış tipi üretimin akademik dünyada da öneminin artmasına yol açmıştır. Gerçek hayat problemlerinin hemen hepsinin çok amaçlı yapıya sahip olması araştırmacıları çok amaçlı akış tipi çizelgeleme problemlerine yöneltmiştir.

Çok amaçlı akış tipi çizelgeleme problemleri literatürde geniş bir yere sahiptir. Robert ve Rajkumar (2019), son işin tamamlanma zamanının ve toplam akış süresinin enküçüklendiği iki amaçlı akış tipi çizelgeleme problemi için melez bir çözüm yaklaşımı önermişlerdir. Yuan, Xu ve Yin (2019),

*Sorumlu yazar; e-posta : tsarac@ogu.edu.tr

çalışmalarında bulanık üretim sürelerinin olduğu permütasyon akış tipi çizelgeleme problemi için bulanık son işin tamamlanma zamanı ve bulanık toplam akış süresi amaçlarını enküçükleyen yerel arama tabanlı bir çözüm yaklaşımı önermişlerdir. Jiang ve Wang (2019), sıra bağımlı hazırlık süreli ve enerji etkin permütasyon akış tipi çizelgeleme problemini ele almışlardır. Problemin amaçları son işin tamamlanma zamanını ve enerji tüketimini enküçüklemektir. Problemin çözümü için çok amaçlı bir evrimsel algoritma geliştirilmiştir. Seif, Yu ve Rahmanniyay (2018), makinaların bakım gereksinimlerinin de dikkate aldığı çok amaçlı permütasyon akış tipi çizelgeleme problemi için karma tamsayılı matematiksel model önermişlerdir. Problemin amaçları toplam bakım maliyetinin ve toplam gecikmenin enküçüklenmesidir. Deng ve Wang (2017), çalışmalarında çok amaçlı dağıtık permütasyon akış tipi çizelgeleme problemi için bir memetik algoritma önermişlerdir. Ele aldıkları amaç fonksiyonları son işin tamamlanma zamanının ve toplam gecikmenin enküçüklenmesidir. Karimi ve Davudpour (2014), akış tipi çizelgeleme probleminde son işin tamamlanma zamanını ve toplam ağırlıklı gecikme süresini enküçüklemek amacıyla değişken komşu arama (VNS) yaklaşımını kullanan bir genetik algoritma önermişlerdir. Ciavotta, Minella ve Ruiz (2013), sıra bağımlı hazırlık sürelerinin olduğu akış tipi çizelgeleme problemi için son işin tamamlanma zamanını, toplam akış süresini ve toplam ağırlıklı gecikmeyi enküçüklemek amacıyla RIPG (*Restarted Iterated Pareto Greedy*) methodunu kullanmışlardır. Schaller ve Valente (2013), beklemelere izin verilmeyen ve kümesel hazırlık sürelerinin olduğu permütasyon akış tipi çizelgeleme probleminde toplam erken tamamlanma ve toplam gecikme sürelerinin enküçüklenmesi amacıyla sezgisel bir algoritma geliştirmişlerdir. Lacoste, Ibanez ve Stützle (2011), son işin tamamlanma zamanı, toplam tamamlanma süresi, toplam ağırlıklı gecikme ve toplam gecikme kriterlerini ikili amaçlar halinde ele almışlar ve problemlerin çözümüne yönelik olarak melez bir çözüm yaklaşımı geliştirmişlerdir. Mokotoff (2011), klasik akış tipi çizelgeleme probleminde son işin tamamlanma zamanını ve toplam tamamlanma zamanını enküçüklemek için tavlama benzetimi yöntemini kullanmıştır. Yagmahan ve Yenisey (2010), son işin tamamlanma zamanını ve toplam akış süresini enküçüklemek amacıyla karınca kolonisi algoritmasını ve yerel arama yaklaşımını birleştirerek çok amaçlı karınca koloni sistem algoritmasını önermişlerdir. Naderi, Tavakkoli-Moghaddam ve Khalili (2010), son işin tamamlanma zamanını ve toplam ağırlıklı gecikmeyi enküçüklemek amacıyla melez bir algoritma önermişlerdir. Karimi, Zandieh ve Karamooz (2010), son işin tamamlanma zamanını ve toplam ağırlıklı gecikmeyi enküçüklemek için sıra bağımlı hazırlık sürelerini göz önünde bulunduran çok fazlı bir çözüm yaklaşımı geliştirilmiştir. Hatami, Ebrahimnejad, Tavakkoli-Moghaddam, ve Maboudian (2010), ortalama akış süresi ve en büyük gecikmeyi enküçüklemek için tavlama benzetimi ve yasaklı arama algoritmalarını kullanmışlardır. Behnamian, Fatemi Ghomi ve Zandieh (2009), hazırlık sürelerinin sıra bağımlı olduğu ve ara stokların dikkate alındığı akış tipi çizelgeleme problemi için tavlama benzetimi ve en yakın komşu arama yöntemlerini önermişlerdir. Problemin amaçları, son işin tamamlanma zamanını, toplam erken tamamlanma ve gecikme sürelerini enküçüklemektir. Dhingra ve Chandna (2009), sıra bağımlı hazırlık sürelerini dikkate aldıkları çalışmalarında son işin tamamlanma zamanını, toplam gecikme süresini ve toplam erken tamamlanma süresini enküçükleyecek melez bir genetik algoritma geliştirmişlerdir. Mokotoff (2009), son işin tamamlanma zamanını ve toplam akış süresini enküçükleyecek tavlama benzetini tabanlı iki yeni algoritma önermiştir. Xu ve Zhou (2009), stokastik akış tipi çizelgeleme problemini ele almışlardır. Problemin amaçları, toplam tamamlanma süresini ve toplam erken üretme süresini enküçüklemektir. Yazarlar, bir benzetim modelini genetik algoritma (GA) ile birleştirmişlerdir. Pan, Wang ve Qian (2009), kesikli diferansiyel evrim algoritmasını kullanarak, makinelerin boş beklemesine izin verilmemesi kısıtı altında toplam üretim süresini ve en büyük gecikmeyi enküçüklemeyi amaçlamışlardır. Vahed, Dangchi, Rafiei ve Salimi (2009), toplam erken tamamlanma ve toplam gecikme sürelerinin ağırlıklı ortalamalarının enküçüklenmesi amacını ele almışlardır. Pareto çözümleri bulabilmek için komşu değişken arama (VNS) tabanlı yeni bir çok amaçlı melez algoritma geliştirmişlerdir. Qian, Wang, Huang ve Wang (2009), ardışık makineler arasındaki sınırlı stok kısıtı altında son işin tamamlanma zamanını ve toplam gecikmeleri enküçüklemek amacıyla permütasyon akış tipi çizelgeleme problemi için diferansiyel evrim tabanlı melez bir algoritma önermişlerdir. Qian, Wang, Huang, Wang ve Wang (2009), makinelerin boş beklememe kısıtı altında son işin tamamlanma zamanının ve toplam gecikmelerin ayrıca toplam geciken iş sayısının ve toplam makine boş kalma süresinin enküçüklenmesi amaçlarıyla diferansiyel evrim tabanlı bir memetik

algoritma geliştirmiştir. Ciavotta, Ruiz ve Minella (2009), hazırlık sürelerinin olduğu ve olmadığı durumlar için permütasyon akış tipi çizelgeleme probleminde toplam üretim süresi ve toplam ağırlıklı gecikme süresi ile toplam üretim süresi ve toplam akış süresini enküçüklemek amaçlarıyla RIPG algoritmasını geliştirmişlerdir. Huang ve Yang (2009), makine boş kalma zamanı, işlerin toplam bekleme süresini ve toplam gecikmeyi enküçüklemek amacıyla karınca kolonisi algoritmasını önermişlerdir. Kahraman, Engin ve Yilmaz (2009), ortalama gecikmeyi ve geciken iş sayısını enküçüklemek amacıyla çok amaçlı bulanık akış tipi atölye problemi için yapay bağışıklık sistemi algoritmasını önermişlerdir. Naderi, Zandieh, Balagh ve Roshanaei (2009), toplam tamamlanma zamanı ve toplam gecikmeyi enküçüklemek amacıyla tavlama benzetimi tabanlı bir meta-sezgisel geliştirmiştir. Sha ve Lin (2009), toplam üretim süresi, ortalama akış süresi ve toplam makine boş kalma zamanları olmak üzere üç kriterli akış tipi çizelgeleme problemi için parçacık sürü optimizasyonu algoritmasını kullanmışlardır. Son yıllarda çok amaçlı permütasyon akış tipi çizelgeleme problemlerini konu alan çalışmalar incelendiğinde çalışmaların önemli bir kısmında hazırlık sürelerinin ihmal edildiği ve amaç sayısının iki olduğu dikkat çekmektedir. En çok ele alınan amaçlar ise son işin tamamlanma zamanı ve toplam gecikmelerin enküçüklenmesidir. Bu çalışmada, sıra bağımlı hazırlık sürelerinin olduğu, son işin tamamlanma zamanını, toplam gecikmeyi ve toplam erken tamamlanma süresini enküçüklemek amacıyla üç amaçlı permütasyon akış tipi üretim problemi ele alınmıştır. Ele alınan problem için üç sıralama kuralı, bir GA ve bir NSGA-II algoritması önerilmiştir.

2. Problem ve Matematiksel Modeli

Bu çalışmada, üç amaçlı, m makine ve r tane işin mevcut olduğu permütasyon akış tipi çizelgeleme (M):

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad \forall\, i \tag{1}$$

$$\sum_{i=1}^{r} x_{ij} = 1 \qquad \forall\, j \tag{2}$$

$$C_{11} = \sum_{l=1}^{r} (p_{l1} + h_{l1}) x_{l1} \tag{3}$$

problemi ele alınmıştır. Amaçlar, son işin tamamlanma zamanının, toplam gecikmenin ve toplam erken tamamlanma süresinin enküçüklenmesidir. Hazırlık sürelerinin her makine için farklı ve sıra bağlımlı olduğu varsayılmıştır. Ele alınan problem, literatürde $Fm \mid prmu, ST_{sd} \mid C_{enb}, \sum T_i, \sum E_i$ şeklinde gösterilmektedir. Problemin matematiksel modeli aşağıda verilmiştir.

İndis Kümesi:

$I = \{i,l \mid i=l = 1,..,r\}$ iş kümesi,
$J = \{j \mid j = 1,..,n\}$ sıra kümesi,
$K = \{k \mid k = 1,..,m\}$ makine kümesi,

Parametreler:

d_i : i. işin teslim zamanı
p_{ik} : i. işin k. makinedeki işlem süresi
h_{ik} : i. işin k. makinedeki hazırlık süresi (birinci sıradaki iş için geçerlidir)
s_{ilk} : i. işten l. işe geçildiğinde k. makinedeki hazırlık süresi
w_1, w_2, w_3 : sırasıyla birinci, ikinci ve üçüncü amaçların ağırlıkları

Karar Değişkenleri:

x_{ij} : i. iş j. sırada çizelgelenmişse, 1; diğer durumlarda, 0.
a_{jk} : j. sıradaki işin k. makinede en erken başlayabileceği zaman
C_{jk} : j. sıradaki işin k. makinede tamamlanma zamanı
C_{enb} : son işin tamamlanma zamanı
T_i : i. işin gecikme süresi
E_i : i. işin teslim zamanından ne kadar önce tamamlandığı

$$C_{1k} \geq C_{1(k-1)} + p_{lk} + h_{l1} - M(1 - x_{l1}) \quad \forall\, l,k \quad k \neq 1 \tag{4}$$

$$C_{j1} \geq C_{(j-1)1} + p_{l1} + s_{il1} - M\left(2 - x_{i(j-1)} - x_{lj}\right) \quad \forall\, i,j,l \quad j \neq 1 \tag{5}$$

$$a_{jk} \geq C_{(j-1)k} \quad \forall\, j,k \quad k \neq 1, \quad j \neq 1 \tag{6}$$

$$a_{jk} \geq C_{j(k-1)} \quad \forall\, j,k \quad k \neq 1, \quad j \neq 1 \tag{7}$$

$$C_{jk} = a_{jk} + p_{lk} + s_{ilk} - M\left(2 - x_{i(j-1)} - x_{lj}\right) \quad \forall\, i,j,k,l \quad k \neq 1, \quad j \neq 1 \tag{8}$$

$$E_i \geq d_i - C_{jm} - M\left(1 - x_{ij}\right) \quad \forall\, i,j \tag{9}$$

$$T_i \geq C_{jm} - d_i - M\left(1 - x_{ij}\right) \quad \forall\, i,j \tag{10}$$

$$a_{jk},\ C_{jk},\ E_i, T_i \geq 0 \tag{11}$$

$$x_{ij} \in \{0,1\} \tag{12}$$

k.a.

$$enk\ z = (C_{enb}, \textstyle\sum_{i=1}^{n} T_i, \sum_{i=1}^{n} E_i) \tag{13}$$

(1) ve (2) numaralı kısıtlar sırasıyla, her işin bir sıraya atanmasını ve her sıraya bir işin atanmasını sağlar. (3) numaralı kısıt, birinci sıraya atanmış işin birinci makinedeki tamamlanma zamanını ifade eder. (4) numaralı kısıt, birinci sıraya atanan işin birinci makine hariç her makinedeki tamamlanma zamanını, (5) numaralı kısıt ise, her işin birinci makinedeki tamamlanma zamanını ifade etmektedir. (6) ve (7) numaralı kısıtlar, *j*. sıradaki işin *k*. makinede en erken başlayabileceği zamanı göstermektedir. Buna göre *j*. sıradaki iş, *k*. makinede bir önceki işin bitiminden itibaren ya da bir önceki makinede tamamlanmasından sonra hangisi daha geç olmuşsa *j*. sıradaki iş en erken o zaman başlayabilir. Sekizinci kısıt, *j*. sıradaki işin *k*. makinedeki tamamlanma zamanını ifade eder. (9) numaralı kısıt, *i*. işin gecikme süresini, (10) numaralı kısıt ise, *i*. işin erken tamalanma süresini ifade etmektedir. (11) ve (12) numaralı kısıtlar da işaret kısıtlarıdır. Amaç fonksiyonları, (13) numaralı ifade de verilmiştir ve son işin tamamlanma zamanının, toplam gecikmenin ve toplam erken tamamlanma süresinin enküçüklenmesini ifade etmektedir.

Matematiksel modelin çözüm performansını test edebilmek amacıyla, 5 iş–2 makine (*Örnek-1*) ve 15 iş–7 makine (*Örnek-2*) olmak üzere 2 örnek problem türetilmiştir. Örneklerin çözümünde, amaç fonksiyonları klasik ağırlıklandırma yöntemi kullanılarak ağırlıklandırılmış ve 36 farklı ağırlık seti (w_1, w_2, w_3) kullanılmıştır.

Örnek-1, GAMS/Cplex çözücüsü ile 36 farklı ağırlık seti ile çözülmüş olmasına rağmen sadece (24, 40, 0) çözümü elde edilmiştir. Örnek-2 için ise çözüm elde edilememiştir. Akış tipi üretim çizelgeleme problemleri NP-zor sınıfa girmektedir. NP-zor problemlerin karmaşıklığı üstel olarak arttığından büyük boyutlu problemlerin çözümü için sezgisel yöntemlerin kullanması gerekmektedir.

3. Ele Alınan Problem İçin Önerilen Çözüm Yöntemleri

3.1 Literatürdeki algoritmalar

Erişilen literatür incelendiğinde, *Fm|prmu,* $ST_{sd}|C_{enb}, \sum T_i, \sum E_i$ problemini ele almış olan tek bir çalışmaya ulaşılmıştır. Dhingra ve Chandna (2009), sıralama kuralı tabanlı dört tane sezgisel geliştirmiştir. Bunlar: En küçük teslim zamanı (EDD), en büyük teslim zamanı (LDD), en küçük işlem ve teslim zamanı (EPDD) ve en büyük işlem ve teslim zamanı (LPDD) tabanlı dört farklı melez GA algoritmasıdır. Geliştirdikleri algoritmalarda seçim için ikili turnuva, çaprazlama için iki noktalı sıra çaprazlaması ve mutasyon için SWAP operatörünü kullanmışlardır.

Bu çalışmada, öncelikle literatürde yer alan sıralama kuralları incelenerek ele alınan problemin çözümü için uyarlanmıştır. Daha sonra bir genetik algoritma ve NSGA-II algoritması önerilmiştir. Önerilen NSGA-

II algoritmasının başarısı, geliştirilen sıralama kurallarının dâhil edilmesiyle, arttırılmıştır.

Öncelik ya da sıralama kuralları; iş merkezlerinde işlerin tek bir iş merkezine veya makineye hangi sıra ile ele alınacağına kılavuzluk eden kurallardır. Her bir öncelik kuralı, farklı performans ölçütlerinde daha başarılı olmaktadır. SPT (*Shortest Processing Time*-en kısa işlem süresi), ortalama akışı ve buna bağlı olarak sistemdeki iş sayısını ve ortalama tamamlanma zamanını enküçüklerken; EDD, ortalama iş gecikmesini ve erken tamamlanma süresini enküçüklemektedir. CR (*Critical Ratio*-kritik oran) kuralı, uzun işlem süreli işlerde daha iyi sonuçlar vermektedir. ATC (*Apparent Tardiness Cost*-görünen gecikme maliyeti) kuralı ise ağırlıklı toplam gecikmeyi enküçükleyen öncelik kurallarından birisidir (Valente, 2003).

Bu çalışmada ele alınan her bir amaç fonksiyonu gözönünde bulundurularak, SST (*Shortest Setup Time*-en kısa hazırlık süresi) kuralı, sıra bağımlı hazırlık sürelerinin varlığında, toplam hazırlık süresini enküçüklemek, dolayısıyla da son işin tamamlanma zamanını (C_{enb}) enküçüklemek amacıyla kullanılmıştır. EDD kuralı toplam gecikme ve toplam erken tamamlanma süresini ve ATCS (*Apparent Tardiness Cost with Setups*-hazırlık süreli görünen gecikme maliyeti) kuralı da toplam gecikmeyi enküçüklemek amacıyla seçilmiştir. Ancak her üç sıralama kuralı da tek makine çizelgeleme problemleri için geliştirilmiş olduğundan, $Fm|prmu,ST_{sd}|C_{enb},\sum T_i,\sum E_i$ probleminin çözümünde kullanılabilmeleri için problemin doğasına uygun olup olmadığının incelenmesi ve gerekiyorsa uyarlanmaları gerekmektedir. Seçilen üç sıralama kuralının ele alınan problemin çözümünde nasıl kullanılabileceği aşağıda açıklanmıştır.

SST öncelik kuralına göre, işler en az hazırlık süresi gerektiren işten başlanır ve bir sonraki iş yine en az hazırlık süresine sahip olacak şekilde belirlenir. Ancak, $Fm|prmu,ST_{sd}|C_{enb},\sum T_i,\ \sum E_i$ probleminde birden fazla makine vardır ve işlerin her makine için hazırlık süreleri farklıdır. Bu nedenle önerilen uyarlanmış SST (SST-u) öncelik kuralında, makine bazında sıra bağımlı hazırlık süreleri toplanır ve elde edilen son matris üzerinden küçükten büyüğe sıralama yapılır. Eğer aynı hazırlık süresine sahip iş varsa rastgele bir seçim yapılarak küçükten büyüğe sıralamaya devam edilir. Elde edilen bu sıra işlerin işlem sırasıdır.

EDD öncelik kuralında işler teslim tarihlerine göre küçükten büyüğe sıralanır. Bu sıralama kuralının ele alınan problem için uyarlanmasına ihtiyaç yoktur.

ATCS öncelik kuralı, MS, WSPT ve SST kurallarının birleşmesinden oluşmaktadır. Bunun için öncelikle klasik ATCS kuralını açıklamak gerekir. ATCS kuralında (15)-(20) formülleri kullanılarak hesaplanan k_1 ve k_2 değerleri (14) numaralı formülde yerine konularak her bir iş için $I_i(t)$ göstergesi hesaplanır. İşler bu gösterge kullanılarak sıralanır.

p_{ort}: Ortalama işlem süresi
s_{ort}: Ortalama hazırlık süresi
t: Göstergenin hesaplandığı zaman
d_{ort}: Ortalama teslim zamanı
$\hat{C}_{enb}$: Tahmini son işin tamamlanma zamanı
d_{enb}: Enbüyük teslim zamanı
d_{enk}: Enküçük teslim zamanı
τ: Teslim zamanı sıkılığı
n: Hazırlık süresi ağırlığı
R: Teslim zamanı aralığı
k_1: Teslim zamanı faktörü
k_2: Hazırlık süresi faktörü

$$\mathrm{I_i(t)} = \frac{w_i}{\mathrm{p_i}}\, \mathrm{e}^{\left(-\frac{\mathrm{enb(d_i-p_i-t,0)}}{\mathrm{k_1 p_{ort}}}\right)} \mathrm{e}^{\left(-\frac{\mathrm{s_{il}}}{\mathrm{k_2 s_{ort}}}\right)} \tag{14}$$

$$\tau = 1 - \frac{d_{ort}}{\hat{C}_{enb}} \tag{15}$$

$$n = \frac{s_{ort}}{p_{ort}} \tag{16}$$

$$R = \frac{(d_{enb} - d_{enk})}{\hat{C}_{enb}} \tag{17}$$

$$k_1 = 4{,}5 + R \rightarrow R \leq 0{,}5 \tag{18}$$

$$k_1 = 6 - 2R \rightarrow R > 0{,}5 \tag{19}$$

$$k_2 = \frac{\tau}{2\sqrt{n}} \tag{20}$$

Ele alınan problemde birden fazla makine olduğundan, bu öncelik kuralının kullanılabilmesi için tüm makinaların parametrelerinden tek bir değer türetilmesine gereksinim vardır. Bu nedenle klasik ATCS formülünde (14) yer alan göstergenin türetilmesinde yapılan uyarlamalar izleyen şekildedir. Öncelikle (21) ve (22) numaralı formüller kullanılarak sırasıyla i. işin toplam işlem süresi ve

toplam hazırlık süresi hesaplanır. Daha sonra (23) ve (24) numaralı formüller kullanılarak işlerin ortalama işlem süresi ve ortalama hazırlık süreleri hesaplanır.

$$p_i = \sum_{k=1}^{m} p_{ik} \quad \forall i \tag{21}$$

$$s_i = \sum_{k=1}^{m} s_{ik} \quad \forall i \tag{22}$$

$$p_{ort} = \frac{\sum_{i=1}^{r}\sum_{k=1}^{m} p_{ik}}{r} \tag{23}$$

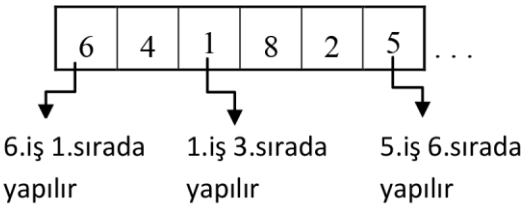
$$s_{ort} = \frac{\sum_{i=1}^{r}\sum_{k=1}^{m} s_{ik}}{r} \tag{24}$$

Son olarak klasik ATCS formülünde olduğu gibi (21)-(24) numaralı eşitliklerden elde edilen değerler (14) nolu denklemde yerlerine konulur. Burada wj=1 olarak alınır. Bu öncelik kuralı uyarlanmış ATCS'dir (ATCS-u).

3.2 Ele Alınan Problem için Bir Genetik Algoritma

Genetik algoritmalar, doğada gözlemlenen evrimsel sürece benzer bir şekilde çalışan arama ve eniyileme yöntemidir. Karmaşık çok boyutlu arama uzayında en iyinin hayatta kalması ilkesine göre bütünsel eniyi çözümü arar. Bunu yaparken de birtakım yöntem ve operatörleri kullanır.

Kodlama: Genetik algoritmaların problemin çözümündeki başarısını etkileyen en önemli faktörlerden birisi, problemin çözümünü temsil eden bireylerin (kromozomların) gösterimidir. Çizelgeleme problemlerinde en yaygın gösterim biçimlerinden birisi, gen diziliminin sıraları, gen değerlerinin ise işleri temsil ettiği gösterimdir. Şekil 1'de bu gösterim biçimi için örmek bir kromozom verilmiştir. Şekil 1'de verilen kromozomun ilk geninin değeri 6'dır, bu 6 nolu işin birinci sırada yapılacağını göstermektedir.



Şekil 1. Çözümün gösterimi

İlk Popülasyonun Türetilmesi: İlk nesil rassal olarak türetilmiştir.

Uyum Değerinin Hesaplanması: Uyum değeri hesaplanırken, amaç fonksiyonları klasik ağırlıklandırma yöntemi kullanılarak skalerleştirilmiştir. Ağırlık seti olarak (w_1, w_2, w_3) = (0.5, 0.25, 0.25) kullanılmıştır.

Seçim Operatörü: Önerilen algoritmada, rulet tekeri, ikili turnuva ve üçlü turnuva yöntemi olmak üzere üç seçim operatörü kodlanmıştır. Üçlü turnuva seçim operatöründe, rastgele üç kromozom seçilmektedir. Bunların arasından uyum değeri en küçük olan kromozom, bir sonraki nesle aktarılmaktadır.

Çaprazlama Operatörü: Önerilen algoritmada, tek noktalı sıra çaprazlaması, iki noktalı sıra çaprazlaması ve *PMX* kullanılmıştır. İki noktalı sıra çaprazlamasına göre, kromozomlarda çaprazlama yapılacak iki nokta belirlenir ve bu parçalar karşılıklı olarak yer değiştirilerek iki yeni nesil kromozom elde edilir. Diğer genler sırasıyla birinci ebeveynden birinci yeni nesil kromozoma aktarılır. Aynı şekilde ikinci ebeveynden ikinci yeni nesil kromozoma aktarma yapılır. Aktarma sırasında eğer aktarılan gen zaten yeni kromozomda mevcutsa, diğer gene geçilir, mevcut değilse yeni kromozoma bu gen aktarılır. Tek noktalı sıra çaprazlamasında ise sadece tek bir nokta belirlenir ve geri kalan adımlar iki noktalı sıra çaprazlamasında olduğu gibi uygulanır. *PMX* çaprazlama operatöründe ise rassal iki nokta seçilir ve bu noktalar arasında gen alışverişi olur. Bu iki nokta arasında karşı gelen her gen *işaretleme* yapılarak adreslenir ve yeni nesil kromozomlar bu *işaretleme* listesine bakılarak doldurulur.

Mutasyon Operatörü: Bu çalışmada '*Swap*', '*Insert*' ve '*Shift*' mutasyon operatörleri kullanılmıştır. *Swap* operatörü uygulandığında rastgele belirlenen bir kromozomdaki rastgele belirlenmiş iki gen yer değiştirir. *Insert* mutasyonda ise rastgele belirlenmiş bir gen, yine rastgele belirlenmiş bir genin arkasına eklenmektedir. *Shift* mutasyonda ise kromozomda bir nokta belirlenir ve ondan sonraki gen dizilimi genlerin değerine göre küçükten büyüğe sıralanır.

3.3 Ele Alınan Problem için Bir NSGA-II Algoritması

Önerilen GA ile bir probleme farklı etkin çözümler türetilebilmesi için farklı ağırlık setlerini kullanarak tekrar tekrar çözülmesi gerekmektedir. NSGA-II algoritması ise tekrarlı çözümlere ve ağırlık setlerine gereksinim duymaksızın çok sayıda etkin çözüm bulabilen, hızlı ve etkin bir algoritmadır. NSGA-II algoritmasında genetik algoritmalardan farklı olarak baskınlık ve yığılma uzaklığı sıralaması olmak üzere iki ara işlem uygulanmaktadır.

Baskınlık: Popülasyondaki her birey her bir amaç için birbirleriyle kıyaslanır. Her bireyin baskın olduğu bireylerin sayısı ve o bireye baskın gelen bireylerin sayısı tutulur. Eğer bir bireyin baskınlık derecesi sıfırsa yani o bireye başka hiçbir birey baskın gelmemişse o birey en iyiler sınıfı olan F1 kümesine dahil edilir. Eğer bir bireyin baskınlık derecesi birse, sadece bir tane birey o bireye baskın gelmiş demektir. O halde F2 sınıfına dâhildir.

Yığılma Uzaklığı Sıralaması: Her amaç için bireyler küçükten büyüğe sıralanırlar. Uç değerlere yani enküçük ve en büyük değerlere '∞' değeri atanır. Daha sonra o amaç için ikinci bireyden başlanarak sondan bir önceki bireye kadar (25) nolu formül uygulanarak ilgili birey için yığılma uzaklığı değeri (CD_i) hesaplanır.

$$CD_i = \sum_{g=1}^{n} \frac{|f_{i+1}^g - f_{i-1}^g|}{f_{enb}^g - f_{enk}^g} \quad \forall i \qquad (25)$$

(25) numaralı formülde *g*, amaç, *i*, birey indisidir. f_i^g ise *i*. bireyin *g*. amaç fonksiyonunun değerini ifade etmektedir.

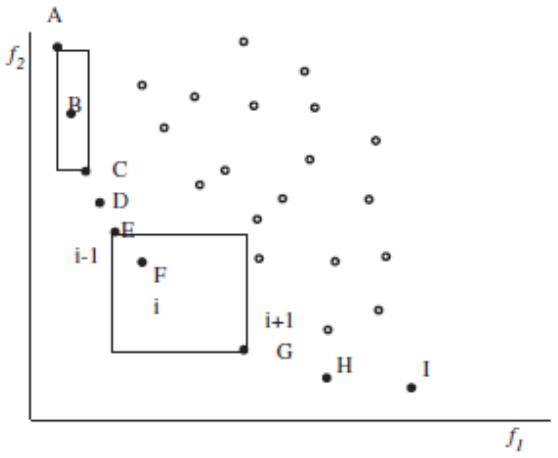
$$var_i = \frac{1}{n}\sum_{g=1}^{n}\left(\left|f_{i+1}^g - f_{i-1}^g\right| - CD_i\right)^2 \qquad (27)$$

Önerilen NSGA-II algoritmasında kromozom gösterimi, ilk neslin üretilmesi, seçim, çaprazlama ve mutasyon operatörleri geliştirilen genetik algoritma ile aynıdır. Önerilen NSGA-II'nin ana prosedürü aşağıda verilmiştir.

Öncelikle, başlangıç popülasyon için '*N-3*' adet birey rassal olarak türetilir. Daha sonra SST-u, EDD ve ATCS-u öncelik kuralları ile elde edilen üç birey de ilk nesle dâhil edilir. Böylece *N* büyüklüğünde bir başlangıç popülasyon elde edilir. *N* büyüklüğündeki ebeveyn popülasyondan GA operatörleri kullanılarak yine *N* birey içeren yeni bir nesil türetilir. Bu iki popülasyon birleştirilerek *2N* büyüklüğünde bir popülasyon elde edilir.

Popülasyondaki bireyler baskınlıklarına göre sınıflandırılır. Sınıflandırılan popülasyondan yine *N* büyüklüğünde yeni bir popülasyon oluşturmak için en iyi sınıf olan F_1 kümesinden başlanarak sırasıyla *F* kümeleriyle yeni ebeveyn kümesi doldurulur. Eğer herhangi bir *F* kümesi ebeveyn popülasyondan taşarsa yani $F_1+F_2+\ldots+F_n>N$ olursa bu durumda F_n

Yığılma uzaklığı, o bireyin her amaç için komşuluklarına olan uzaklığının toplamı olduğundan her zaman en büyük uzaklığa sahip olan birey seçilecektir. Şekil 2'de enk (f_1,f_2) amaçları altında en büyük alana sahip olan *F* bireyi seçilirken *C*, *D* ve *E* bireyleri baskın çözüm olduğu halde küçük alana sahip olduğu için elenmiş olup bir sonraki nesle aktarılamamıştır.



Şekil 2. Bireylerin Yığılma Uzaklığı

Yığılma uzaklığı bazı iyi çözümleri gözardı edebileceği için bu çalışmada, *dinamik yığılma uzaklığı* kullanılmıştır. Dinamik yığılma uzaklığı formülü (26)'da verilmiştir.

$$DCD_i = \frac{CD_i}{\log\left(\frac{1}{Var_i}\right)} \qquad (26)$$

(26) nolu formüldeki var_i değerleri (27) nolu formül kullanılarak hesaplanabilir.

kümesine dinamik yığılma uzaklığı prosedürü uygulanır. Buna göre F_n kümesindeki bireylerin dinamik yığılma uzaklığı değerleri büyükten küçüğe sıralanır ve sıralanmış F_n kümesinin ebeveyn popülasyonu dolduracak sayıdaki ilk bireyleri yeni popülasyona seçilir. Daha sonra yine aynı prosedür işletilir.

Fakat ikinci döngüde klasik turnuva seçim operatörü yerine yığılma karşılaştırma operatörü kullanılır. Bu operatöre göre rastgele seçilen iki bireyden hangisinin baskınlığı fazla ise başka bir deyişle değer olarak düşük F kümesine sahip olan seçilir. Örneğin rastgele seçilmiş biri F_1, diğeri F_2'de olan iki bireyden F_1'de olan birey seçilir. Eğer bu rastgele seçilen bireylerin ikisi de aynı baskınlık düzeyine sahipse yani aynı F kümesinde ise bu sefer yığılma uzaklığı değerlerine bakılır. Yığılma uzaklığı değeri büyük olan birey seçilir.

Bu çalışmada araştırma ve yayın etiğine uyulmuştur.

4. Deneysel Sonuçlar

Türetilen test problemleri ilk olarak önerilen sıralama kuralları kullanılarak çözülmüştür. Daha sonra aynı test problemleri, erişilebilen literatürde $Fm|prmu,ST_{sd}|C_{enb},\sum T_i,\sum E_i$ problemini ele almış tek çalışma olan Dhingra ve Chandna'nın (2009) geliştirmiş olduğu dört algoritma ile çözülmüştür. Son olarak önerilen *GA* ve *NSGA-II* algoritmaları Dhingra ve Chandna'nın (2009) algoritmaları ile kıyaslanmıştır.

Sözü geçen tüm algoritmalar C# programlama dilinde kodlanmış ve testler 2.68 Ghz, 3,49 GB RAM ve Intel Core Two Quad CPU özelliklerindeki bir bilgisayar kullanılarak yapılmıştır.

4.1 Test Problemlerinin Türetilmesi

Test problemleri, küçük, orta ve büyük olmak üzere üç farklı boyutta türetilmiştir. Küçük problemler 25, orta problemler 75 ve büyük problemler 200 işten oluşmaktadır. Makine sayıları her örnek problem için 10, 30 ve 50 olarak belirlenmiştir. Böylece 25_10, 25_30, 25_50, 75_10, 75_30, 75_50, 200_10, 200_30, 200_50 olmak üzere 9 farklı problem tipi tariflenmiştir. Her bir problem tipi için 5 adet örnek türetilmiştir. Test problemleri, '*problem tipi_örnek numarası*' biçiminde isimlendirilmiştir. Örneğin; 200 işin ve 30 makinanın olduğu 200_30 problem tipinin 4. örneği 200_30_4 olarak adlandırılmıştır. Test problemlerinin işlem süreleri (p_{ik}) [100,990] aralığında hazırlık süreleri (s_{ijk}) ise [10,150] aralığında rassal türetilmiştir. Teslim zamanları (d_i) (28) numaralı formül (Rajendran ve Ziegler, 2003) kullanılarak türetilmiştir.

$$d_i = \bar{s} + \sum_1^m p_{ik} + u(0,1)(n-1)\bar{p} \qquad (28)$$

Burada $\bar{s}$, ortalama hazırlık süresi, $\bar{p}$, ortalama işlem süresi, $u(0-1)$, 0-1 arasında rassal sayı, m, makine sayısı ve n iş sayısıdır. İşlerin hesaplanan d_i değerlerine göre öncelik kuralları kullanılarak elde edilen çözümlerin amaç fonksiyonu değerleri $z = w_1F_1 + w_2F_2 + w_3F_3$ şeklinde bir fayda fonksiyonu yardımıyla tek bir değere dönüştürülmüştür. Burada, (w_1, w_2, w_3) = (0.50, 0.25, 0.25) olarak alınmıştır. Elde edilen çözümler Tablo 1'de verilmiştir.

Tablo 1
Öncelik Kurallarının Kıyaslanması

problem__no	ATCS-u	EDD-u	SST-u	problem__no	ATCS-u	EDD-u	SST-u
	z	z	z		z	z	z
25_10_1	46559,75	**36574,25**	55472,75	**75_30_4**	308113,3	**225008**	410587,5
25_10_2	44365	**24169,5**	55195,5	**75_30_5**	348470	**236808,5**	397952,3
25_10_3	45516,25	**41620,5**	46572,25	**75_50_1**	432895,5	**360589,8**	446829
25_10_4	45815	**30206**	48855,5	**75_50_2**	416172,5	**320959,3**	462760,3
25_10_5	35592,5	**27069,75**	47981,5	**75_50_3**	421354	**339240,5**	490877,3
25_30_1	68867,5	**60474,75**	77518	**75_50_4**	375337,8	**326733,3**	486992,3
25_30_2	67789,75	**62565,5**	70873	**75_50_5**	385398,8	**311289,8**	443829
25_30_3	56030,5	**55346**	72860,5	**200_10_1**	1987540	**506192,3**	2016829
25_30_4	72693,5	**52267,75**	73922,75	**200_10_2**	1923403	**756747**	2154158
25_30_5	62218,75	**47240**	72515,25	**200_10_3**	1890509	**627780,8**	2135047
25_50_1	**76081,5**	83079,75	86804,75	**200_10_4**	1908256	**618228,8**	1960673
25_50_2	**69110,25**	74361,25	85742,25	**200_10_5**	1995498	**594287,3**	1992422
25_50_3	77262,5	**68719,75**	95884,5	**200_30_1**	2403737	**1191211**	2199795
25_50_4	**76993,75**	83002,75	90555,5	**200_30_2**	2419952	**1077819**	2298204
25_50_5	**78162,25**	82584,5	94053,25	**200_30_3**	2362490	**1281720**	2449266
75_10_1	288882,75	**140106**	343945	**200_30_4**	2326285	**1440272**	2401350
75_10_2	329336	**138753**	303645,5	**200_30_5**	2237716	**1262837**	2394574
75_10_3	293454,75	**146767,75**	336389,5	**200_50_1**	2473732	**1398978**	2499811
75_10_4	291917,25	**171993**	343670	**200_50_2**	2519424	**1507731**	2443086
75_10_5	300280,75	**149807,5**	343074,3	**200_50_3**	2285061	**1410237**	2337438
75_30_1	327493,75	**221745,25**	363420,5	**200_50_4**	2488508	**1522102**	2574540
75_30_2	395268,25	**239793,25**	368966,5	**200_50_5**	2284195	**1354507**	2431841
75_30_3	358617	**225907**	400595,5				

Tablo 1 incelendiğinde, *EDD*'nin diğer sıralama kurallarına göre daha başarılı olduğu söylenebilir. *SST-u* hiçbir test problemi için başarılı bir çözüm üretemezken, ATCS-*u* bazı problemlerde *EDD*'yi geçebilmiştir.

4.2 Test Sonuçları

GA ve *NSGA-II* algoritmalarının parametre değerlerini belirleyebilmek üzere deney tasarımı yapılmıştır. Deney planı Tablo 2'de verilmiştir. Her bir problem tipi için bir örnek problem kullanılarak dokuz ayrı deney seti çözülmüştür. Elde edilen çözümler MINITAB programında değerlendirilmiştir.

Tablo 2
Deney Planı

	Düzey 1	Düzey 2	Düzey 3
Nesil Sayısı	500	1000	
Populasyon Büyüklüğü	50	100	
Mutasyon Oranı	0,001	0,01	
Çaprazlama Oranı	0,01	0,1	
Seçim Operatörü	rulet	ikili turnuva	üçlü turnuva
Çaprazlama Operatörü	tek noktalı sıra çaprazlama	iki noktalı sıra çaprazlama	pmx
Mutasyon Operatörü	swap	insert	shift

Tablo 3'de her problem tipi için seçilmiş olan uygun parametre değerleri verilmiştir.

Tablo 3
Her Bir Problem Tipi için Uygun Parametre Değerleri

Problem Tipi	GA							NSGA-2						
	Pop. Byk.	Nesil Sys.	mo	co	sop	cop	mop	Pop. Byk.	Nesil Sys.	mo	co	sop	cop	mop
25_10	50	1000	0,01	0,1	ikili turnuva	PMX	insert	50	500	0,001	0,1	rulet	iki noktalı	insert
25_30	50	1000	0,01	0,01	üçlü turnuva	tek noktalı	insert	50	500	0,001	0,1	rulet	tek noktalı	insert
25_50	50	1000	0,01	0,01	ikili turnuva	PMX	insert	50	1000	0,001	0,1	üçlü turnuva	tek noktalı	insert
75_10	50	1000	0,01	0,01	üçlü turnuva	PMX	insert	50	500	0,001	0,1	rulet	PMX	insert
75_30	50	1000	0,01	0,01	üçlü turnuva	PMX	insert	50	1000	0,001	0,1	ikili turnuva	PMX	insert
75_50	50	1000	0,001	0,01	üçlü turnuva	PMX	insert	50	500	0,001	0,1	üçlü turnuva	PMX	insert
200_10	50	1000	0,001	0,01	üçlü turnuva	PMX	insert	100	1000	0,001	0,1	rulet	tek noktalı	insert
200_30	50	1000	0,001	0,01	üçlü turnuva	PMX	insert	50	500	0,001	0,1	rulet	iki noktalı	insert
200_50	100	1000	0,001	0,01	üçlü turnuva	PMX	insert	50	500	0,001	0,1	üçlü turnuva	iki noktalı	insert

*Pop. Byk. :Populasyon Büyüklüğü, mo: Mutasyon Oranı, co: Çaprazlama Oranı, sop: Seçim opt, cop: Çaprazlama opt, mop: Mutasyon opt

Önerilen algoritmalarının başarısının sınanabilmesi amacıyla öncelikle *Örnek1* (5 iş-2 makine) ve *Örnek2* (15 iş ve 7 makine) sonrasında ise türetilen küçük, orta ve büyük boyutlu test problemleri hem Dhingra ve Chandna'nın (2009) sezgiselleriyle hem de önerilen yöntemlerle çözülmüştür. Elde edilen sonuçlar sırasıyla Tablo 4, 5, 6 ve 7'de verilmiştir. Dhingra ve Chandna'nın (2009) sezgiselleri için yazarların önerdikleri parametre değerleri, GA ve NSGA-II için ise Tablo 3'teki parametre değerleri kullanılmıştır. Tüm algoritmalar beş tekrarlı çalıştırılmıştır. Öncelikle her bir algoritma ile edilen çözümler içinden baskın olanlar seçilmiştir. Daha sonra elde edilen bu çözümler, algoritma bazında birbirleriyle kıyaslanmıştır. Birbiriyle kıyaslanan bu çözümler tabloda baskın çözüm sayısı kısmında belirtilmiştir. Örneğin: Tablo 5'te 25_10_1 nolu problem için HGA-1 algoritması beş kez çalıştırılmıştır. Elde edilen beş çözüme kendi içinde baskınlık testi yapılmıştır.

Test sonucunda üç baskın çözüm elde edilmiştir. Bu üç çözüm diğer algoritmaların baskın çözümleri ile kıyaslanmıştır. HGA-1, HGA-2, HGA-3 ve HGA-4 bu problem için baskın çözüm bulamazken GA, 3 ve NSGA-II ise 28 baskın çözüm bulmuştur. Ayrıca tabloların (Tablo 4-Tablo 7) süre bölümünde beş tekrar için geçen toplam süre saniye cinsinden verilmiştir.

Tablo 4
Örnek 1 ve *Örnek 2* için Test Sonuçları

Yöntem	HGA-1			HGA-2			HGA-3			HGA-4			GA			NSGA-2		
Problem no	Çözüm Sayısı	Etkin Çözüm Sayısı	Süre	Çözüm Sayısı	Etkin Çözüm Sayısı	Süre	Çözüm Sayısı	Etkin Çözüm Sayısı	Süre	Çözüm Sayısı	Etkin Çözüm Sayısı	Süre	Çözüm Sayısı	Etkin Çözüm Sayısı	Süre	Çözüm Sayısı	Etkin Çözüm Sayısı	Süre
5_2	1	1	0,4266	1	1	0,4098	1	1	0,4213	1	1	0,4568	1	1	0,3266	4	4	10,187
15_7	1	0	0,7956	1	0	0,8905	1	0	0,9877	1	0	0,9956	1	0	1,6224	139	78	27,089

Tablo 4'ten de görülebileceği gibi, *Örnek1* için tüm diğer algoritmalar ile sadece 1 baskın çözüm bulunabilirken, *NSGA-II* algoritması ile 4 adet baskın çözüme ulaşılmıştır. *Örnek2* için ise *NSGA-II* algoritması ile bulunan 78 birbirine baskın olmayan çözüm diğer algoritmaların çözümlerini bastırmıştır. Türetilen problemlerden elde edilen sonuçlar sırasıyla Tablo 5, 6 ve 7'de verilmiştir. Tablo 8'de ise GA ve NSGA-II'nin kıyaslanması yapılmıştır.

Tablo 5
Küçük Boyutlu Problemlerin Test Sonuçları

Yöntem	HGA-1			HGA-2			HGA-3			HGA-4			GA			NSGA-2		
Problem no	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre
25_10_1	3	0	3,824041	4	0	3,777386	4	0	4,257961	3	0	4,207885	3	3	3,96875	29	28	10,34375
25_10_2	4	0	4,172051	5	0	3,911645	4	0	3,542066	2	0	4,39704	2	2	3,90625	26	20	10,01563
25_10_3	4	0	3,696479	2	0	4,168716	1	0	3,927335	2	0	3,899283	5	5	3,9375	28	27	10,98438
25_10_4	3	0	3,82393	4	0	4,416936	4	0	4,166883	3	0	3,55806	5	5	3,953125	24	24	10,4375
25_10_5	5	0	3,653423	5	0	3,57925	4	0	3,993437	4	0	4,300269	5	4	3,921875	32	25	11,03125
25_30_1	5	0	10,49917	3	0	10,03288	5	0	9,633157	4	0	10,47059	4	4	9,5625	38	38	24,21875
25_30_2	4	0	9,622336	2	0	9,785921	3	0	9,823582	3	0	9,911091	3	3	9,5625	31	31	23,73438
25_30_3	5	0	9,735577	5	0	10,18994	5	0	9,771104	1	0	9,943177	4	4	9,546875	42	38	22,26563
25_30_4	4	0	10,41116	4	0	10,03244	1	0	10,00317	5	0	10,43945	3	3	9,5625	20	10	22,29688
25_30_5	5	0	9,953031	2	0	10,02611	3	0	10,42599	4	0	9,519853	2	2	9,5625	21	17	22,8125
25_50_1	5	0	16,19091	4	0	16,20792	4	0	15,63163	3	0	16,44717	4	4	15,5625	19	16	71,34375
25_50_2	2	0	16,12873	3	0	16,39348	4	0	15,51728	2	0	15,84351	3	3	15,54688	32	32	70,78125
25_50_3	2	0	16,3591	4	0	15,63222	3	0	15,84469	5	0	16,21338	2	4	15,57813	35	35	67,17188
25_50_4	3	0	16,23397	3	0	16,07278	4	0	16,03629	3	0	15,50003	2	2	15,57813	36	36	69,04688
25_50_5	5	0	16,19706	3	0	16,06237	3	0	15,77385	2	0	16,28073	3	3	15,5625	35	30	75,45313

Tablo 5 incelendiğinde, 25 iş ve 10 makineden oluşan problemlerde HGA algoritmalarının hiçbirisi baskın çözüm bulamamıştır. GA ve NSGA-II algoritmaları baskın çözümler elde etmiştir. HGA-1 algoritması 25 iş ve 10 makine için ortalama 0,766 sn'de çözüm bulurken, HGA-2 ortalama 0,793sn, HGA-3 ortalama 0,794 sn ve HGA-4 ise ortalama 0,814 sn' de çözüm bulmuştur. GA ortalama 0,787 sn ve *NSGA-II* ise ortalama 2,112 sn'de çözüm bulmuştur.

25 iş ve 30 makineden oluşan problemlerde HGA algoritmaları baskın çözüm bulamamıştır. GA ve NSGA-II algoritmaları baskın çözümler elde etmiştir. HGA-1, HGA-2, HGA-3 ve HGA-4 algoritmalarının ortalama çözüm süreleri sırasıyla, 2,009, 2,003, 1,986 ve 2,011 sn' dir. GA bu problemlere ortalama 1,911 sn ve NSGA-II ise ortalama 4,613 sn'de çözüm bulmuştur.

25 iş ve 50 makineden oluşan problemlerde HGA algoritmaları baskın çözüm bulamamıştır. GA ve NSGA-II algoritmaları baskın çözümler elde etmiştir. HGA-1 algoritması 25 iş ve 50 makine problemlerine ortalama 3,244 sn'de çözüm bulurken, HGA-2 ortalama 3,215 sn, HGA-3 ortalama 3,152 sn ve HGA-4 ise ortalama 3,211 sn' de çözüm bulmuştur. GA ortalama 3,113 sn ve NSGA-II ise ortalama 14,152 sn'de çözüm bulmuştur.

25 boyutlu test problemlerinin tümü dikkate alındığında NSGA-II algoritmasının çözüm süresinin diğer algoritmalardan fazla olmasına karşın türettiği baskın çözüm sayısı çarpıcı bir şekilde fazladır.

Tablo 6
Orta Boyutlu Problemlerin Test Sonuçları

Yöntem	HGA-1			HGA-2			HGA-3			HGA-4			GA			NSGA-2		
Problem no	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre
75_10_1	3	0	12,49371	5	0	11,61545	4	0	11,68614	4	0	11,71274	5	5	11,82813	32	25	23,625
75_10_2	4	0	12,10037	3	0	12,04318	4	0	11,98164	5	0	11,83003	4	4	11,79688	29	26	22,71875
75_10_3	4	0	12,38148	3	0	12,18156	4	0	11,63348	3	0	11,86572	2	2	11,75	36	22	23,98438
75_10_4	2	0	11,79432	2	0	11,84119	3	0	12,45673	4	0	11,7322	3	3	11,84375	31	18	23,15625
75_10_5	5	0	12,32998	5	0	11,66835	2	0	12,20952	3	0	11,61993	5	5	11,76563	23	19	23,07813
75_30_1	4	0	30,20392	3	0	30,09371	5	0	29,89235	5	0	30,48849	3	3	29,14063	26	25	120,8125
75_30_2	2	0	29,78825	4	0	30,13826	2	0	29,88293	4	0	29,91383	3	3	29,1875	36	25	120,8906
75_30_3	4	0	30,33296	2	0	30,11469	2	0	30,38318	3	0	29,94848	3	3	29,1875	33	18	118,7813
75_30_4	1	0	30,41268	4	0	29,66089	4	0	30,07746	2	0	30,04014	2	2	29,20313	27	21	119,8906
75_30_5	2	0	30,32535	3	0	29,91872	5	0	29,76565	5	0	30,23968	3	3	29,20313	35	14	118,7656
75_50_1	4	0	47,0624	3	0	47,03763	5	0	47,29859	4	0	46,78446	3	2	46,71875	36	31	98,71875
75_50_2	4	0	46,93492	5	0	47,31411	5	0	46,74463	4	0	46,84779	3	3	46,64063	41	31	46,46466
75_50_3	4	0	47,49218	5	0	47,14127	4	0	47,20876	5	0	46,71968	3	3	46,67188	15	9	99,65625
75_50_4	3	0	47,08748	3	0	46,88095	4	4	47,06043	4	0	46,82329	4	4	46,65625	23	21	96,03125
75_50_5	4	0	46,57606	3	0	46,99811	5	5	47,36339	4	4	46,57555	3	3	46,67188	19	16	96,01563

Tablo 6 incelendiğinde, 75 iş ve 10 makineden oluşan problemlerde GA ve NSGA-II algoritmaları baskın çözümler elde edilebilirken, HGA algoritmalarının hiçbirisi ile baskın çözüm bulunamamıştır. Ortalama çözüm süreleri HGA algoritmaları için sırasıyla 2,444, 2,374, 2,399 ve 2,350sn'dir. GA ve NSGA-II için ise 2,359 ve 4,663 sn'dir.

75 iş ve 30 makineden oluşan problemlerde yine HGA algoritmaları baskın çözüm bulamamıştır. GA ve NSGA-II algoritmaları baskın çözümler elde etmiştir. HGA-1 algoritması için ortalama 6,043 sn'de çözüm bulurken, HGA-2 ortalama 5,997 sn, HGA-3 ortalama 6,000 sn ve HGA-4 ise ortalama 6,025 sn' de çözüm bulmuştur. GA ortalama 5,837 sn ve NSGA-II ise ortalama 23,966 sn'de çözüm bulmuştur.

75 iş ve 50 makineden oluşan problemlerde HGA-1 ve HGA-2 algoritmaları baskın çözüm bulamamıştır. HGA-3 4 ve 5 nolu problemlerde baskın çözüm elde ederken HGA-4 5 nolu problemde baskın çözüm bulabilmiştir. GA ve NSGA-II algoritmaları baskın çözümler elde etmiştir. HGA-1 algoritması 75 iş ve 50 makine için ortalama 9,406 sn'de çözüm bulurken, HGA-2 ortalama 9,415 sn, HGA-3 ortalama 9,427 sn ve HGA-4 ise ortalama 9,350 sn' de çözüm bulmuştur. GA ortalama 9,334 sn ve NSGA-II ise ortalama 17,475 sn'de çözüm bulmuştur. Orta boyutlu test problemlerinin tümü dikkate alındığında NSGA-II algoritmasının türettiği baskın olmayan çözüm sayısı diğer algoritmalara kıyasla çarpıcı bir şekilde fazladır.

Tablo 7
Büyük Boyutlu Problemlerin Test Sonuçları

Yöntem	HGA-1			HGA-2			HGA-3			HGA-4			GA			NSGA-2		
Problem no	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre	Çözüm Sayısı	Etkin Çözüm	Süre
200_10_1	2	0	30,35656	4	0	30,48311	4	0	29,83073	5	0	30,37157	1	1	29,85938	28	27	157,1094
200_10_2	3	0	30,34607	5	0	29,74033	3	0	30,38106	4	0	30,34917	2	2	29,96875	24	21	158,7031
200_10_3	3	0	30,00174	5	0	30,3815	3	0	29,72826	5	0	29,5065	2	2	29,89063	24	20	158,0938
200_10_4	4	0	30,37309	2	0	29,78691	4	0	29,95422	3	0	29,96499	4	4	29,89063	27	27	157,7031
200_10_5	5	0	30,24875	4	0	30,3548	4	0	30,23612	3	0	30,02071	3	3	29,89063	17	15	156,6094
200_30_1	4	0	77,02619	3	0	76,79076	4	0	76,69762	4	0	77,46195	3	3	76,29688	30	15	291,2344
200_30_2	3	0	76,90417	3	0	76,79211	4	0	77,49617	2	0	77,10003	1	1	76,28125	22	12	288,875
200_30_3	5	0	77,24166	2	0	76,62877	4	0	76,70994	5	0	76,73223	2	2	76,23438	19	10	293,0313
200_30_4	2	0	76,84741	2	0	77,46558	5	0	76,84866	5	0	77,4834	2	2	76,26563	16	11	295,1406
200_30_5	4	0	77,43304	4	0	77,0026	4	0	76,60896	4	0	76,9572	1	1	76,3125	19	12	287,0625
200_50_1	3	0	247,3081	3	0	247,3071	4	0	246,9781	5	0	246,578	3	3	246,7344	24	14	378,5156
200_50_2	3	0	244,0789	4	0	243,9133	5	0	244,0604	5	0	243,9435	2	2	243,1563	21	5	395,9063
200_50_3	4	0	246,4724	4	0	245,7407	3	0	246,4453	3	0	245,7585	2	2	245,9688	18	13	391,4531
200_50_4	4	0	244,4015	4	0	244,4076	4	0	243,7332	2	0	243,7557	5	5	243,2813	11	5	376,9219
200_50_5	3	0	246,7023	4	0	246,6772	4	0	247,4929	4	0	247,3968	3	3	246,1094	18	8	387,0938

Tablo 7 incelendiğinde, 200 iş ve 10 makineden oluşan problemlerde HGA algoritmaları baskın çözüm bulamamıştır. GA ve NSGA-II algoritmaları baskın çözümler elde etmiştir. HGA-1 algoritması 200 iş ve 10 makine için ortalama 6,053 sn'de çözüm bulurken, HGA-2 ortalama 6,030 sn, HGA-3 ortalama 6,005 sn ve HGA-4 ise ortalama 6,009 sn' de çözüm bulmuştur. GA ortalama 5,980 sn ve NSGA-II ise ortalama 31,529 sn'de çözüm bulmuştur.

200 iş ve 30 makineden oluşan problemlerde HGA algoritmaları baskın çözüm bulamamıştır. GA ve NSGA-II algoritmaları baskın çözümler elde etmiştir. HGA-1 algoritması 200 iş ve 30 makine için ortalama 15,418 sn'de çözüm bulurken, HGA-2 ortalama 15,387 sn, HGA-3 ortalama 15,374 sn ve HGA-4 ise ortalama 15,429 sn' de çözüm bulmuştur. GA ortalama 15,256 sn ve NSGA-II ise ortalama 58,214 sn'de çözüm bulmuştur.

200 iş ve 50 makineden oluşan problemlerde HGA algoritmaları baskın çözüm bulamamıştır. GA ve NSGA-II algoritmaları baskın çözümler elde etmiştir. HGA-1 algoritması 200 iş ve 50 makine için ortalama 49,159 sn'de çözüm bulurken, HGA-2 ortalama 49,122 sn, HGA-3 ortalama 49,148 sn ve HGA-4 ise ortalama 49,097 sn' de çözüm bulmuştur. GA ortalama 49,010 sn ve NSGA-II ise ortalama 77,196 sn'de çözüm bulmuştur. Büyük boyutlu test problemlerinin tümü dikkate alındığında NSGA-II algoritmasının türettiği baskın çözüm sayısı diğer algoritmalara kıyasla çarpıcı bir şekilde fazladır. Özellikle literatürdeki daha önce önerilmiş yöntemlerin hiçbirisinin baskın bir çözüm üretememiş olması dikkat çekicidir.

Önerilen GA ile bu bölümdeki tüm testlerde kullanılmış olan (w_1, w_2, w_3) = (0.5, 0.25, 0.25) ağırlık değerleri yerine farklı ağırlıklar kullanarak farklı çözümler elde etmek mümkündür. 25-10-3 nolu problem on farklı ağırlık seti için GA ile çözülmüş ve elde edilen sonuçlar NSGA-II ile kıyaslanmıştır. Seçilen ağırlıklar ve bunlara karşılık bulunan GA çözümleri ile NSGA-II sonuçları Tablo 8'de verilmiştir. Tablodan da görülebileceği gibi belirlenmiş ağırlıklarla GA çalıştırıldığında elde edilen çözümlerden beş tanesi baskın çözümdür. Buna karşılık NSGA-II algoritması 28 çözüm bulmuştur. Bunlardan 27'si baskın çözümdür. Ağırlık setinin büyütülmesi ile GA'nın daha çok çözüm bulabileceği açıktır. Ancak NSGA-II'nin ikinci bir tekrara gereksinim duymadan ulaşabildiği etkin çözüm sayısı ve ağırlık setine ihtiyaç duymaması gibi faktörler göz önünde bulundurulduğunda önerilen NSGA-II algoritmasının başarısını yadsımak mümkün olamayacaktır.

Tablo 8
GA ve NSGA-II'nin Kıyaslanması

GA için belirlenmiş ağırlıklar			GA			NSGA-2		
			Cmax	TE	TT	Cmax	TE	TT
1	0	0	22188	10204	66996	**19791**	**28782**	**99852**
0,9	0,1	0	21054	12148	68507	**19928**	**26916**	**101292**
0,9	0	0,1	22357	10996	68141	**20821**	**10510**	**112661**
0,8	0	0,2	21125	11860	68725	**20317**	**17755**	**92796**
0,7	0,3	0	21226	10462	69155	**19964**	**22585**	**100970**
0,7	0,2	0,1	**20878**	**9709**	**66401**	**20031**	**21030**	**90993**
0,7	0,1	0,2	**22044**	**6927**	**67929**	**21831**	**425**	**109887**
0,7	0	0,3	**20991**	**9639**	**68673**	**21523**	**495**	**98285**
0,8	0,2	0	**21348**	**5859**	**70180**	**20459**	**15774**	**107136**
0,8	0,1	0,1	**21119**	**8712**	**68028**	**21433**	**4899**	**88538**
						21390	**5268**	**88856**
						21166	**7284**	**90723**
						21782	**2683**	**94395**
						20407	**16675**	**100872**
						21246	**1847**	**96056**
						19688	**22822**	**106750**
						19746	**19380**	**104777**
						20048	**16158**	**107209**
						19665	**23469**	**109172**
						20527	**10663**	**91152**
						21243	**3799**	**96357**
						21528	**4145**	**89976**
						21528	**3710**	**92218**
						20461	**17240**	**87153**
						20290	**18031**	**85010**
						21038	**5443**	**99757**
						20783	**13368**	**84191**
						21568	8497	87104

5. Sonuç ve Öneriler

Bu çalışmada, çok amaçlı, sıralamaya bağlı hazırlık süreli permütasyon akış tipi üretim çizelgeleme problemi ele alınmıştır. Ele alınan problemin, en büyük tamamlanma zamanını, toplam gecikmeyi ve toplam erken tamamlanma süresini enküçüklemek olmak üzere üç amaç fonksiyonu vardır. Son yıllarda çok amaçlı akış tipi problemlerini ele alan çalışmalar incelendiğinde, çalışmaların genellikle özel bir süreç kısıtını ele almadıkları ya da sadece tek bir süreç özelliğine yoğunlaştıkları (çoğunlukla da sıralamaya bağlı hazırlık süreleri) görülmektedir. Bu çalışmaların önemli bir kısmı iki amaçlıdır (son işin tamamlanma zamanı ve toplam gecikmenin enküçüklenmesi). Çalışmalarda farklı çözüm teknikleri kullanılmakla beraber GA en çok kullanılan yöntem olmuştur. Dhingra ve Chandna'nın (2009) çalışması, $Fm \mid prmu, ST_{sd} \mid C_{enb}, \sum T_i, \sum E_i$ problemini ele almış erişilebilen tek çalışmadır. Dhingra ve Chandna (2009), bu problem için dört farklı *GA* önermiştir. Erişilebilen literatürde, $Fm \mid$

prmu, ST_{sd} | *Cenb*, $\sum T_i$, $\sum E_i$ probleminin *NSGA-II* algoritması ile çözüldüğü bir çalışmaya rastlanmamıştır.

Problemin matematiksel modeli ile sadece 5 iş ve 2 makineli küçük örnek probleme, GAMS paket programı kullanılarak çözüm üretilebilmiştir. GAMS, çok büyük boyutlu olmamasına rağmen, 15 iş ve 7 makine problemini çözmekte bile yetersiz kalmıştır. Problemin *NP-zor* doğası, daha büyük boyutlu problemleri çözebilmek için sezgisel yöntemleri kullanmayı gerektirmektedir.

Bu çalışmada, *Fm* | *prmu*, ST_{sd} | C_{enb}, $\sum T_i$, $\sum E_i$ probleminin çözümü için SST-u, EDD ve ATCS-u, sıralama kuralları, bir GA ve bir NSGA-II algoritması önerilmiştir. Önerilen algoritmaların başarısını sınamak amacıyla, küçük, orta ve büyük boyutlu test problemleri kullanılmıştır. Küçük ve büyük boyutlu problemlerde Dhingra ve Chandna'nın (2009) sezgisellerinin çözümlerinin tamamı, GA ve NSGA-II algoritmalarının çözümleri tarafından bastırılmıştır. Orta boyutlu problemlerde ise Dhingra ve Chandna 'nın sezgisellerinden HGA-1 ve HGA-2 pareto-etkin çözümler elde edemezken HGA-3 ve HGA-4 algoritmaları sadece problemlerin çok az bir bölümünde başarılı olabilmiştir. GA ve NSGA-II algoritmaları problemlerin tamamında, baskın çözümler elde edebilmiştir. NSGA-II algoritmasının türettiği baskın çözüm sayısının diğer algoritmalara kıyasla çarpıcı bir şekilde fazla olması dikkat çekicidir.

Gerçek hayat problemlerinin yapısı itibariyle, gelecekte, çok sayıda süreç özelliğini dikkate alan, çok amaçlı, kullanıcının tercihlerini göz önünde bulunduran ve hesaplama etkinliği yüksek algoritmaların geliştirilmesi ve uygulanması çok amaçlı çizelgeleme problemleri adına dikkate değer çalışmalar olacaktır. Bu tür çalışmaların artmasıyla, gerçek hayat problemleriyle baş etme konusunda da önemli adımlar atılmış olacaktır.

Araştırmacıların Katkısı

Bu çalışmada; Tuğba Saraç, matematiksel modelin kurulması, çözüm yaklaşımlarının geliştirilmesi, deneysel sonuçların yorumlanması ve makalenin yazılması, Nilay Bilgiçer, bilimsel yayın taramasının yapılması, test problemlerinin türetilmesi, çözüm yaklaşımlarının geliştirilmesi ve kodlanması, konularında katkı sağlamıştır.

Çıkar Çatışması

Yazarlar tarafından herhangi bir çıkar çatışması beyan edilmemiştir.

Kaynaklar

Behnamian, J., Fatemi Ghomi, S.M.T & Zandieh, M. (2009). A multi-phase covering Pareto-optimal front method to multi-objective scheduling in a realistic hybrid flowshop using a hybrid metaheuristic. *Expert Systems with Applications*, 36(8), 11057-11069. doi: http://doi.org/10.1016/j.eswa.2009.02.080

Ciavotta, M., Minella, G. & Ruiz, R. (2013). Multi-objective sequence dependent setup times permutation flowshop: A new algorithm and a comprehensive study. *European Journal of Operational Research,* 227(2), 301–313. doi: http://doi.org/10.1016/j.ejor.2012.12.031

Ciavotta, M., Ruiz, R. & Minella, G. (2009). New results for the multi-objective sequence dependent setup times flowshop problem. *Multidisciplinary International Conference on Scheduling : Theory and Applications (MISTA 2009) Proceeding Book*, 808-810. Erişim adresi: http://www.mistaconference.org/2009/abstracts/808-810-019-A.pdf

Deng J. & Wang L. (2017). A competitive memetic algorithm for multi-objective distributed permutation flow shop scheduling problem. *Swarm and Evolutionary Computation*, (32), 121-131. doi: http://doi.org/10.1016/j.swevo.2016.06.002

Dhingra, A. & Chandna, P. (2009). Hybrid genetic algorithm for multicriteria scheduling with sequence dependent setup time. *International Journal of Engineering*, 3(5), 510-520. Erişim adresi: https:/www.cscjournals.org/manuscript/Journals/IJE/Volume3/Issue5/IJE-112.pdf

Hatami, S., Ebrahimnejad, S., Tavakkoli-Moghaddam, R. & Maboudian, Y. (2010). Two meta-heuristics for three-stage assembly flowshop scheduling with sequence-dependent setup times. *The International Journal of Advanced Manufacturing Technology* 50(9), 1153–1164. doi: http://doi.org/10.1007/s00170-010-2579-5

Huang R.H. & Yang C.L. (2009). Solving a multi-objective overlapping flowshop scheduling. *The*

International Journal of Advanced Manufacturing Technology, 42(9), 955–962. doi: http://doi.org/10.1007/s00170-008-1652-9

Jiang E. & Wang L. (2019). An improved multi-objective evolutionary algorithm based on decomposition for energy-efficient permutation flow shop scheduling problem with sequence-dependent setup time. *International Journal of Production Research*, 57(6), 1756-1771. doi: http://doi.org/10.1080/00207543.2018.1504251

Kahraman, C., Engin, O. & Yilmaz, M.K. (2009). A New Artificial Immune System Algorithm for Multiobjective Fuzzy Flow Shop Problems. *International Journal of Computational Intelligence Systems*, 3, 236-247. doi: http://doi.org/10.1080/18756891.2009.9727656

Karimi, N. & Davoudpour, H. (2014). A high performing metaheuristic for multi-objective flowshop scheduling problem. *Computers & Operations Research*, 52, 149–156. doi: http://doi.org/10.1016/j.cor.2014.01.006

Karimi, N., Zandieh, M. & Karamooz, H.R. (2010). Bi-objective group scheduling in melez flexible flowshop: A multi-phase approach. *Expert Systems with Applications*, 37(6), 4024-4032. doi: http://doi.org/10.1016/j.eswa.2009.09.005

Lacoste , J., Ibanez, M.L. & Stützle, T. (2011). A hybrid TP+PLS algorithm for bi-objective flow-shop scheduling problems. *Computers & Operations Research*, 38(8), 1219–1236. doi: http://doi.org/10.1016/j.cor.2010.10.008

Mokotoff, E. (2009). Minimizing the Makespan and Total Flow Time on the Permutation Flowshop Scheduling Problem. *Multidisciplinary International Conference on Scheduling : Theory and Applications (MISTA 2009 Proceeding Book*, 479-506. Erişim adresi: http://www.mistaconference.org/2009/papers/479-506-069-P.pdf

Mokotoff, E. (2011). Algorithms for bicriteria minimization in the permutation flow shop scheduling problem. *Journal of Industrial and Management Optimization*, 7(1), 253-282. doi: http://doi.org/10.3934/jimo.2011.7.253

Naderi, B., Tavakkoli-Moghaddam, R. & Khalili, M. (2010). Electromagnetism-like mechanism and simulated annealing algorithms for flowshop scheduling problems minimizing the total weighted tardiness and makespan. *Knowledge-Based Systems,* 23(2), 77–85. doi: http://doi.org/10.1016/j.knosys.2009.06.002

Naderi, B., Zandieh, M., Balagh, A.K.G. & Roshanaei, V. (2009). An improved simulated annealing for hybrid flowshops with sequence-dependent setup and transportation times to minimize total completion time and total tardiness. *Expert Systems Applications*, 36(6), 9625–9633. doi: http://doi.org/10.1016/j.eswa.2008.09.063

Pan, Q.K., Wang, L. & Qian, B. (2009). A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems. *Computers & Operations Research*, 36(8), 2498 – 2511. doi: http://doi.org/10.1016/j.cor.2008.10.008

Qian, B., Wang, L., Huang, D., Wang, W. & Wang X. (2009). An effective hybrid DE-based algorithm for multi-objective flow shop scheduling with limited buffers. *Computers & Operations Research,* 36(1), 209 – 233. doi: http://doi.org/10.1016/j.cor.2007.08.007

Qian, B., Wang, L., Huang, D.X. & Wang X., (2009). Mutli-objective no-wait flow-shop scheduling with a memetic algorithm based on differential evolution. *Soft Comput,* 13, 847–869. doi: http://doi.org/10.1007/s00500-008-0350-8

Rajendran, C. & Ziegler, H. (2003). Scheduling to Minimize the Sum of Weighted Flowtime and Weighted Tardiness of Jobs in A Flowshop with Sequence-Dependent Setup Times. *European Journal of Operational Research,* 149 (3), 513–522. doi: http://doi.org/10.1016/S0377-2217(02)00485-X

Robert, R.B.J. & Rajkumar, R. (2019). A hybrid algorithm for multi-objective optimization of minimizing makespan and total flow time in permutation flow shop scheduling problems. *Journal of Information Technology and Control*, 48 (1), 47-57. doi: http://doi.org/10.5755/j01.itc.48.1.20909

Schaller, J. & Valente, J.M.S. (2013). An evaluation of heuristics for scheduling a non-delay permutation flow shop with family setups to minimize total earliness and tardiness. *Journal of the Operational Research Society,* 64(6), 805–816. doi: http://doi.org/10.1057/jors.2012.94

Seif J., Yu A.J.& Rahmanniyay F. (2018). Modelling and optimization of a bi-objective flow shop scheduling with diverse maintenance requirements. *International Journal of Production*

Research, 56(9), 3204-3225. doi: http://doi.org/10.1080/00207543.2017.1403660

Sha, D.Y. & Lin, H.H. (2009). A particle swarm optimization for multiobjective flowshop scheduling. *The International Journal of Advanced Manufacturing Technology*, 45(7-8), 749–758. doi: http://doi.org/10.1007/s00170-009-1970-6

Vahed, A.R., Dangchi, M., Rafiei, H. & Salimi, E. (2009). A novel hybrid multi-objective shuffled frog-leaping algorithm for a bi-criteria permutation flow shop scheduling problem. *The International Journal of Advanced Manufacturing Technology*, 41(11), 1227-1239. doi: http://doi.org/10.1007/s00170-008-1558-6

Valente, J.S.M. (2003). Using instance statistics to determine the lookahead parameter value in the ATC dispatch rule: Making a good heuristic better. *FEP Working Papers,* Universidade do Porto, Faculdade de Economia do Porto. Erişim adresi: http://wps.fep.up.pt/wps/wp127.pdf

Xu, J. & Zhou, X. (2009). A class of multi-objective expected value decision-making model with birandom coefficients and its application to flow shop scheduling problem. *Information Sciences*, 179(17), 2997–3017. doi: http://doi.org/10.1016/j.ins.2009.04.009

Yagmahan, B. & Yenisey, M.M. (2010). A multi-objective ant colony system algorithm for flow shop scheduling problem. *Expert Systems with Applications,* 37(2), 1361–1368. doi: http://doi.org/10.1016/j.eswa.2009.06.105

Yuan F., Xu X. & Yin M. (2019). A novel fuzzy model for multi-objective permutation flow shop scheduling problem with fuzzy processing time. *Advances in Mechanical Engineering*, , 11(4), 1–9. doi: http://doi.org/10.1177/1687814019843699