KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

DOKTORA TEZİ

GELİŞTİRİLMİŞ EL BOMBASI PATLATMA YÖNTEMLERİ İLE AKIŞ TİPİ ÇİZELGELEME PROBLEMLERİNİN ÇÖZÜMÜ

TURGAY ÖZTÜRK

KOCAELİ 2023

KOCAELİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

ENDÜSTRİ MÜHENDİSLİĞİ

ANABİLİM DALI

DOKTORA TEZİ

GELİŞTİRİLMİŞ EL BOMBASI PATLATMA YÖNTEMLERİ İLE AKIŞ TİPİ ÇİZELGELEME PROBLEMLERİNİN ÇÖZÜMÜ

TURGAY ÖZTÜRK

Dr. Öğr. Üyesi Celal ÖZKALE		
Danışman,	**Kocaeli Üniv.**	**.....**
Doç. Dr. Kasım BAYNAL		
Eş Danışman,	**Kocaeli Üniv.**	**.....**
Prof. Dr. Semra BORAN		
Jüri Üyesi,	**Sakarya Üniv.**	**.....**
Doç. Dr. Alparslan Serhat DEMİR		
Jüri Üyesi,	**Sakarya Üniv.**	**.....**
Dr. Öğr. Üyesi Hatice ESEN		
Jüri Üyesi,	**Kocaeli Üniv.**	**.....**
Dr. Öğr.Üyesi Pınar YILDIZ KUMRU		
Jüri Üyesi,	**Kocaeli Üniv.**	**.....**
Dr. Öğr. Üyesi Mehlika KOCABAŞ AKAY		
Jüri Üyesi,	**Kocaeli Üniv.**	**.....**

Tezin Savunulduğu Tarih: 05.07.2023

ETİK BEYAN VE ARAŞTIRMA FONU DESTEĞİ

Kocaeli Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım bu tez/proje çalışmasında,

- Bu tezin/projenin bana ait, özgün bir çalışma olduğunu,

- Çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı,

- Bu çalışma kapsamında elde edilen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi,

- Bu çalışmanın Kocaeli Üniversitesi'nin abone olduğu intihal yazılım programı kullanılarak Fen Bilimleri Enstitüsü'nün belirlemiş olduğu ölçütlere uygun olduğunu,

- Kullanılan verilerde herhangi bir tahrifat yapmadığımı,

- Tezin/Projenin herhangi bir bölümünü bu üniversite veya başka bir üniversitede

başka bir tez/proje çalışması olarak sunmadığımı,

beyan ederim.

☒ Bu tez/proje çalışmasının herhangi bir aşaması hiçbir kurum/kuruluş tarafından maddi/alt yapı desteği ile desteklenmemiştir.

☐ Bu tez/proje çalışması kapsamında üretilen veri ve bilgiler .....tarafından ..... no'lu proje kapsamında maddi/alt yapı desteği alınarak gerçekleştirilmiştir.

Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçları kabul ettiğimi bildiririm.

Turgay ÖZTÜRK

YAYIMLAMA VE FİKRİ MÜLKİYET HAKLARI

Fen Bilimleri Enstitüsü tarafından onaylanan lisansüstü tezimin/projemin tamamını veya herhangi bir kısmını, basılı ve elektronik formatta arşivleme ve aşağıda belirtilen koşullarla kullanıma açma izninin Kocaeli Üniversitesi'ne verdiğimi beyan ederim. Bu izinle Üniversiteye verilen kullanım hakları dışındaki tüm fikri mülkiyet hakları bende kalacak, tezimin/projemin tamamının ya da bir bölümünün gelecekteki makale, kitap, tebliğ, lisans, patent gibi çalışmalarda kullanımı, danışmanımın isim hakkı saklı kalmak koşuluyla ve her iki tarafın bilgisi dâhilinde bana ait olacaktır.

Tezin/projenin kendi özgün çalışmam olduğunu, başkalarının haklarını ihlal etmediğimi ve tezimin/projenin tek yetkili sahibi olduğumu beyan ve taahhüt ederim. Tezimde yer alan telif hakkı bulunan ve sahiplerinden yazılı izin alınarak kullanılması zorunlu metinlerin yazılı izin alarak kullandığımı ve istenildiğinde suretlerini Üniversiteye teslim etmeyi taahhüt ederim.

Yükseköğretim kurulu tarafından yayınlanan "***Lisanüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge***" kapsamında tezim aşağıda belirtilen koşullar haricinde YÖK Ulusal Tez Merkezi/ Kocaeli Üniversitesi Kütüphaneleri Açık Erişim Sisteminde erişime açılır

☐ Enstitü yönetim kurulu kararı ile tezimin/projemin erişime açılması mezuniyet tarihinden itibaren 2 yıl ertelenmiştir.

☐ Enstitü yönetim kurulu gerekçeli kararı ile tezimin/projemin erişime açılması mezuniyet tarihinden itibaren 6 ay ertelenmiştir.

☒ Tezim/projem ile ilgili gizlilik kararı verilmemiştir.

Turgay ÖZTÜRK

ÖNSÖZ VE TEŞEKKÜR

Genellikle doğadaki olay veya varlıkların hareketlerinden esinlenerek geliştirilen meta sezgisel yöntemler, optimum veya optimuma yakın çözümler elde edilebilmesi için çözüm uzayının aranmasına rehberlik eden yaklaşımlardır. Yapay zeka teknikleri olarak da bilinen metasezgisel yöntemlerin üretim ve hizmet sektörlerinde çok sayıda uygulaması bulunakta olup her gün bunlara yenileri eklenmektedir. Metasezgisel yöntemler, atama montaj hattı dengeleme, tesis yerleşim, sırt çantası, gezgin satıcı, çizelgeleme vb. optimizasyon problemlerinin çözümünde yaygın şekilde kullanılmaktadır.

Çözümünde metasezgisel yöntemlere sıklıkla başvurulan problemlerden biri de permütasyon akış tipi iş çizelgeleme problemidir (PATÇP). Teknik kısıtlar nedeniyle çok sayıda uygulaması bulunan PATÇP, yarım asırdan üzün süredir araştırmacıların ilgisini çekmektedir. Küçük boyutlu PATÇP'ler tamsayılı programlama vb. klasik optimizasyon yöntemler ile çözülebilirken, büyük boyutlu problemler için makul sürelerde optimum çözümü garantileyen bir yöntem henüz bulunmamaktadır. Bu nedenle bu tez çalışmasında PATÇP'nin çözümü için yeni metasezgiseller geliştirilmeye çalışılmıştır. Bu amaçla sürekli optimizasyon problemleri için önerilen orijinal El Bombası Patlatma Metodu (EBPM), PATÇP'nin çözümü için uyarlanmıştır. Daha sonra Ayrık EBPM (AEBPM) ve yerel arama eklenerek melez $AEBPM_{RYA}$ yöntemleri geliştirilmiştir.

Çalışmanın başından itibaren gösterdikleri emek, destek, ilgi ve sağladıkları motivasyonla çalışmanın gerçekleşmesini ve ilerlemesini sağlayan değerli danışman hocalarım Sayın Doç. Dr. Kasım BAYNAL ve Dr. Öğr. Üyesi Celal ÖZKALE'ye; çalışma süresince fikirleri ile beni destekleyen ve bana yol gösterin tez izleme komitesindeki hocalarım Sayın Dr. Öğr. Üyesi Hatice ESEN ve Dr. Öğr. Üyesi Alpaslan Serhat DEMİR'e sonsuz teşekkür ederim.

Ayrıca uzun çalışma süreci boyunca destekleri ve sağladıkları huzurlu çalışma ortamı için sevgili eşime ve biricik kızma teşekkürü bir borç bilir, ömrüm boyunca beni destekleyen aile büyüklerime minnet duygularımı sunarım.

Temmuz - 2023 Turgay ÖZTÜRK

İÇİNDEKİLER

ETİK BEYAN VE ARAŞTIRMA FONU DESTEĞİ i
YAYIMLAMA VE FİKRİ MÜLKİYET HAKLARI ii
ÖNSÖZ VE TEŞEKKÜR iii
İÇİNDEKİLER iv
ŞEKİLLER DİZİNİ vi
TABLOLAR DİZİNİ vii
SİMGELER VE KISALTMALAR DİZİNİ viii
ÖZET xi
ABSTRACT xii
1. GİRİŞ 1
2. İŞ ÇİZELGELEME PROBLEMLERİ 3
2.1. Çizelgeleme Problemlerinin Notasyonu ve İlgili Kavramlar 6
2.2. Çizelgeleme Problemlerinin Grafiksel Gösterimi 12
2.2.1. Gantt Şeması 13
2.2.2. Graf Gösterimi 14
2.3. Çizelgeleme Problemlerinin Sınıflandırılması 15
2.4. Çizelgeleme Problemlerinin Karmaşıklığı ve Çözüm Yaklaşımları 22
2.4.1. Optimal Çözümü Garantileyen Yöntemler 24
2.4.2. Optimum Çözümü Garanti Etmeyen Yöntemler 25
3. PERMÜTASYON AKIŞ TİPİ ÇİZELGELEME PROBLEMİ 30
3.1. Permütasyon Akış Tipi Çizelgeleme Problemlerinin Notasyonu ve Matematiksel Modeli 31
3.2. PATÇP'lerde Kullanılan Sezgisel Yöntemler 33
3.2.1. NEH Algoritması 35
3.3. PATÇP'lerde Kullanılan Metasezgisel Yöntemler 36
3.3.1. Diferansiyel Gelişim Algoritması 36
3.3.2. PATÇP'nin Çözümünde Metasezgisel Kullanılan Çalışmalar 38
4. EL BOMBASI PATLATMA METODU 40
4.1. Optimal Arama Yönlü EBPM 45
4.2. Parametre Seçimi İçin Öneriler 49
4.3. Optimal Arama Yönlü EBPM'nin Grafiksel Analizi 50
4.4. Gerçek Sayılardan Sıralama Vektörüne Dönüşüm 53
4.5. Ayrık El Bombası Patlatma Metodu 55
4.6. El Bombası Patlatma Metodu ile İlgili Literatür Çalışmaları 59
5. BULGULAR VE TARTIŞMA 62
5.1. Ajan Bölgesinin ve Optimal Arama Yönünün Etkisinin Değerlendirilmesi 64
5.1.1. Normallik Testi 66
5.2. EBPM'nin Doğrulanması 69
5.3. EBPM'nin Performansının DGA ile Karşılaştırılması 71
5.4. NEH ile Üretilen Başlangıç Çözümlerin Katkısı 74
5.5. EBPM'nin İterasyonlar Süresince Yakınsamasının Grafiksel Analizi 77
5.6. Melez DGA_{EBPM} Yöntemi 79
5.7. Ayrık EBM ile Yapılan Deneysel Çalışmalar 82
5.8. AEBPM'nin Performansın Kabul Görmüş Metasezgisellerle Karşılaştırılması 85
6. SONUÇLAR VE ÖNERİLER 91

KAYNAKLAR 96
EKLER 107
KİŞİSEL YAYIN VE ESERLER 176
ÖZGEÇMİŞ 177

ŞEKİLLER DİZİNİ

Şekil 2.1. İmalat Sisteminde Bilgi Akış Şeması 5
Şekil 2.2. Üçlü sistemde çizelgeleme problemlerinin gösterimi..... 9
Şekil 2.3. İş odaklı Gantt şeması..... 13
Şekil 2.4. Makine odaklı Gantt şeması 14
Şekil 2.5. Ayrık Graf..... 14
Şekil 2.6. Uygun bir çizelge 15
Şekil 2.7. Klasik akış tipi yerleşimde iş akışı 18
Şekil 2.8. Genel akış tipinde iş akışı..... 18
Şekil 2.9. Atölye tipi üretim sistemi örnek gösterim 19
Şekil 2.10. Çizelgeleme problemlerinin sınıflandırılması 21
Şekil 2.11. Çizelgeleme problemlerinin genel sınıflandırılması..... 22
Şekil 2.12. Problemlerin karmaşıklığı 23
Şekil 2.13. Çözüm yaklaşımları..... 24
Şekil 2.14. Metasezgisel Algoritmaların Sınıflandırılması..... 28
Şekil 2.15. Çizelgeleme problemlerinde yaygın olarak kullanılan Metasezgiseller..... 29
Şekil 3.1. Yerleşim tipine göre çizelgeleme problemleri ve PATÇP 30
Şekil 3.2. NEH yönteminin işleyiş adımları 35
Şekil 3.3. DGA' da yeni vektörün oluşturulması işlemi..... 37
Şekil 4.1. Patlama alanı ve ajan bölgesi çapı..... 41
Şekil 4.2. Olurlu bölge dışındaki noktaların olurlu bölgeye taşınması..... 42
Şekil 4.3. EBPM akış şeması..... 44
Şekil 4.4. OEBPM akış şeması 48
Şekil 4.5. De Jong fonksiyonun formülü ve yüzey alanı 50
Şekil 4.6. Her iterasyonda optimum noktaya yakınsama durumu 51
Şekil 4.7. Her bir iterasyonda ziyaret edilen en iyi noktalar..... 52
Şekil 4.8. Ziyaret edilen bütün noktalar (a), ilk (b), 16. (c) ve son (d) iterasyonda ziyaret edilen noktalar..... 53
Şekil 4.9. Komşuluk operatörleri (Li ve Yin, 2012)..... 56
Şekil 4.10. AEBPM akış şeması 58
Şekil 5.1. Metotların ortalama bağıl sapma yüzdelerine (Δ_{ort}) ait histogram grafikleri 67
Şekil 5.2. Taillard'ın 1. Probleminim çözümünde ilk 50 iterasyon boyunca elde edilen bağıl sapma yüzdeleri (Δ) 78
Şekil 5.3. Taillard'ın 1.,11.,21.41.ve 51. problemlerinin çözümünde ilk 100 iterasyon boyunca elde edilen ortalama bağıl sapma yüzdeleri (Δ_{ort}) 79

TABLOLAR DİZİNİ

Tablo 3.1. PATÇP İçin Geliştirilen Sezgisel Teknikler 34
Tablo 3.2. PATÇP'nin çözümünde kullanımı yaygın olan metasezgiseller 39
Tablo 4.1. Konum bilgileri ve EKPD yöntemi ile hesaplanan iş sıraları 54
Tablo 4.2. EBPM, OEBPM ve AEBPM için başlangıç parametreleri 57
Tablo 5.1. Çalışmada yaygın kullanılan EBPM başlangıç parametre değerleri 64
Tablo 5.2. Kullanılan parametre seviyeleri ve metot isimleri 65
Tablo 5.3. Farklı R_t ve Nq-ory sevilerine ait ortalama bağıl sapma yüzdeleri (Δ_{ort}) 66
Tablo 5.4. Metotların Δ_{ort} değerlerine ait çarpıklık ve basıklık istatistikleri 68
Tablo 5.5. Kolmogorov-Smirnov ve Shapiro-Wilk testi sonuçları 68
Tablo 5.6. Mann-Whitney U test istatistikleri 69
Tablo 5.7. Problem gruplarına ve yöntemlere göre Δ_{ort}, Δ_{std}, Δ_{min} ve Δ_{max} değerleri 70
Tablo 5.8. Taillard'ın test (1. grup) problemlerine ait sonuçlar 72
Tablo 5.9. Diğer test problemlerine (2. grup) ait sonuçlar 73
Tablo 5.10. EBPM ile DGA karşılaştırması Mann-Whitney U test istatistikleri 74
Tablo 5.11. $EBPM_{NEH}$ ve DGA_{NEH} ile Taillard'ın problemlerinde elde edilen sonuçlar 76
Tablo 5.12. NEH, $EBPM_{NEH}$ ve DGA_{NEH}'nin ortalama bağıl sapma yüzdeleri (Δ_{ort}) 77
Tablo 5.13. DGA, DGA_{NEH} ve DGA_{EBPM} ile Taillard'ın problemlerinde elde edilen sonuçlar 80
Tablo 5.14. DGA, DGA_{NEH} ve DGA_{EBPM} ile 2. grup test problemlerinde elde Δ_{ort} değerleri 81
Tablo 5.15. DGA ile DGA_{EBPM} karşılaştırması Mann-Whitney U test istatistikleri 82
Tablo 5.16. AEBPM ve $AEBPM_{RYA}$'de kullanılan başlangıç parametre değerleri 83
Tablo 5.17. AEBPM ve $AEBPM_{RYA}$ ile Taillard'ın problemlerinde elde edilen sonuçlar 83
Tablo 5.18. AEBPM ve $AEBPM_{RYA}$ ile 2. grup test problemlerinde elde Δ_{ort} değerleri 84
Tablo 5.19. Karşılaştırılan metotlara ait bilgiler 85
Tablo 5.20. Taillar'ın Problemlerine Ait Ortalama Bağıl Yüzde Sapma (Δ_{ort}) Değerleri 86
Tablo 5.21. CDABC'nin Karşılaştırıldığı Çalışmalar (Li ve Yin, 2012) 87
Tablo 5.22. CDBC ve 4 metot ile AEBPM ve $AEBPM_{RYA}$'nin Δ_{ort} değerleri 87
Tablo 5.23. CDBC, TLBO, HTLBO ve VNS yöntemlerine ait Δ_{ort} değerleri 89
Tablo 6. 1. Taillar'ın Problemlerinde NEH, AEBPM ve $AEBPM_{RYA}$'nin Δ_{ort} değerleri 93

SİMGELER VE KISALTMALAR DİZİNİ

a_j : Müsait zaman
block : Bloklanma
brkdwn : Arızalar
C_j : Tamamlanma zamanı
C_{max} : Maksimum tamamlanma zamanı (makespan)
d_j : j işinin teslim zamanı
E_j : Erken tamamlanma
F : Flow time (Akış süresi)
FFc : Esnek akış tipi
F_j : Akış zamanı
FJc : Esnek atölye tipi
Fm : Akış tipi
F_{max} : Toplam akış süresi
fmls : İş aileleri
H_i : i. örnek için test edilmiş herhangi bir sezgisel metot tarafından üretilen çözüm
I_i : Atıl zaman (idle time)
J_m : Atölye tipi
L_j : Geç kalma (lateness)
L_{max} : Maksimum pozitif geç kalma süresi
M_j : Paralel özdeş makine
m_{min} : Patlama yarıçapını güncellemede kullanılan sabit
m_{OSD} : Optimal arama yönünün ağırlığı
Np : Bir bombanın fırlatacağı şarapnel sayısı
Nq : El bombası sayısı
Nq-oay : Optimal arama yönünü belirlemekte kullanılacak el bombası sayısı
nwt : Beklemesi
Om : Açık atölye tipi
P : Araştırma yoğunluğu sabiti
p_{ij} : J işinin i. makinedeki işlem süresi
P_m : Paralel m adet özdeş makine
prec : Öncelik kısıtları
prmp : Bölünebilme
prmu : Permütasyon
P_{sin}, : Optimal aram yönünün ağılığını belirlemekte kullanılan başlangıç parametresi
P_{TS} : Ajan Bölgesi Araştırma Yoğunluğu
Q_m : Farklı işleme hızlarındaki paralel m adet makine
R : Tekrar sayısı
R_e : Patlama yarıçapı
recrc : Yeniden dolaşım
r_j : Geliş (hazır olma) zamanı
r_j : j işinin geliş zamanı
r_m : [-1,1] aralığında düzgün dağılıma göre üretilmiş rassal bir sayı
R_m : İlişkisiz paralel makineler
R_{rd} : $R_{t\text{-}initial}$ değerinin son R_t değerine oranını

R_t : Ajan bölgesi yarıçapı
$R_{t\text{-initial}}$: Başlangıçtaki ajan bölgesi yarıçapı (R_t)
S_{jk} : Sıra bağımlı hazırlık süreleri
Tj : Gecikme (tardiness)
U_i : i.test problemi için bilinen üst sınır değerleri
U_j : Geciken iş sayısı
W_{ij} : Bekleme süresi
w_j : j işinin ağırlığı
W_j : Toplam bekleme süresi
α : Makine ortamı
β : İş karakteristikleri, kısıtlar
γ : Performans ölçütü
Δ : Bağıl sapma yüzdesi
Δ_{max} : Maksimum bağıl sapma yüzdesi
Δ_{min} : Minimum bağıl sapma yüzdesi
Δ_{ort} : Ortalama bağıl sapma yüzdesi (relative percentage deviation)
Δ_{std} : Bağıl sapma yüzdelerinin standart sapması

Kısaltmalar

AATÇ : Açık Atölye Tipi Çizelgeleme
AEBPM : Ayrık El Bombası Patlatma Metodu
$AEBPM_{RYA}$: Refaranslanmış Yerel Arama Eklenmiş Melez Ayrık El Bombası Patlatma Metodu
ATÇ : Atölye Tipi Çizelgeleme
CDABC : Composite Discrete Artifical Bee Colony
CDS : Campbell, Dudek Smith Algoritması
DDE : Discrete Differential Evolution (Ayrık Diferansiyel Gelişim)
DDE_{RLS} : Discrete Differential Evolution with Referenced Local Search (Refaranslanmış Yerel Arama Eklenmiş Melez Ayrık DGA)
DGA : Diferansiyel Gelişim Algoritması
DGA_{EBPM} : Başlangıç Çözümlerin Bir Kısmı EBPM ile Oluşturulan DGA
DGA_{NEH} : Başlangıç Çözümlerin Bir Kısmı NEH ile Oluşturulan DGA
DGEM : Discrete Grenade Explosion Method (Ayrık El Bombası Patlatma Metodu)
$DGEM_{RLS}$: Discrete Grenade Explosion Method with Referenced Local Search (Refaranslanmış Yerel Arama Eklenmiş Melez Ayrık El Bombası Patlatma Metodu)
EBPM : El Bombası Patlatma Metodu
$EBPM_{NEH}$: Başlangıç Çözümlerin Bir Kısmı NEH ile Oluşturulan EBPM
EKPD : En Küçük Pozisyon Değeri
FIFO : First In First Out(İlk Gelen İlk İşlem Görür)
GA : Genetik Algoritma
GEM : Grenade Explosion Method (El Bombası Patlatma Metodu)
HC : Ho ve Chang Yöntemi
HTLBO : Hybrid Teaching–Learning-Based Optimization Algorithm (Melez Öğretme- Öğrenme Tabanlı Optimizasyon Algoritması)
IPSO : Improved Particle Swarm Optimization

KKO	: Karınca Kolonisi Optimizasyonu
M-MMAS	: First Proposed Ant-Colonyalgorithm
MOPF	: Çok Amaçlı Optimal Güç Akışı
$NEGA_{VNS}$	: Genetik Algoritma ile Değişken Komşuluk Arama Metotlarının Birlikte Kullanıldığı Melez Bir Sezgisel
NEH	: Nawaz, Enscore ve Ham Yöntemi
NP	: Non Polinomial
NPSO	: Novel Particle Swarm Optimization
OEBPM	: Optimal Arama Yönlü El Bombası Patlatma Metodu
OSD	: Optimal Search Direction (Optimal Arama Yönü)
PACO	: Posed Ant-Colony Algorithm
PATÇP	: Permütasyon Akış Tipi Çizelgeleme Problemi
PFSP	: Permutation Flowshop Scheduling Problem (Permütasyon Akış Tipi Çizelgeleme Problemi)
PSO	: Parçacık Sürü Optimizasyonu
PSO_{VNS}	: Değişken Komşuluk Arama (Variable Neighborhood Search, VNS) İle Geliştirilmiş Sürü Optimizasyonu
RLS	: Referenced Local Search (Refaranslanmış Yerel Arama)
RYA	: Refaranslanmış Yerel Arama
TA	: Tabu Arama
TB	: Tavlama Benzetimi
TLBO	: Teaching–Learning-Based Optimization Algorithm (Öğretme- Öğrenme Tabanlı Optimizasyon Algoritması)
VNS	: Variable Neighborhood Search (Değişken Komşuluk Arama)
YAKA	: Yapay Arı Kolonisi Algoritması

GELİŞTİRİLMİŞ EL BOMBASI PATLATMA YÖNTEMLERİ İLE AKIŞ TİPİ ÇİZELGELEME PROBLEMLERİNİN ÇÖZÜMÜ

ÖZET

Günümüz rekabet koşullarında işletmeler varlıklarını ancak müşteri taleplerini zamanında ve en az maliyetle karşılayarak sürdürebilirler. Bu nedenle kaynakların verimli kullanılması ve işlerin en iyi şekilde çizelgelenmesi gerekmektedir. Çizelgeleme işlerin hangi kaynakta ne zaman yapılacağının belirlendiği karar sürecidir. Çizelgeleme problemleri farklı sınıflara ayrılmaktadır. Teknik kısıtlar nedeniyle gerçek hayatta çok sayıda uygulaması bulunan permütasyon akış tipi çizelgeleme problemi (PATÇP) yarım asırdan uzun süredir araştırmacıların ilgisini çekmektedir.

El Bombası Patlatma Metodu (EBPM) Ahrari ve arkadaşları tarafından el bombalarının patlamalarından esinlenerek geliştirilmiş evrimsel bir algoritmadır. EBPM'yi diğer metasezgisellerden ayıran temel özellik ajan bölgesi yarıçapıdır. Bu özelik sayesinde ajanlar (el bombalarının) birbirine belirli bir mesafeden daha fazla yaklaşması önlenerek çözüm uzayının homojen bir şekilde taranması sağlanır.

Bu çalışmada öncelikle EBPM, permütasyon akış tipi çizelgeleme problemlerinin çözümü için uyarlanmış ve ajan bölgesi yarıçapının metot performansına etkisi araştırılmıştır. Daha sonra metodun performansını geliştirmek için bir dizi çalışmalar yapılmış, bu kapsamda Ayrık EBPM (AEBPM) ve yerel arama eklenerek melez $AEBPM_{RYA}$ geliştirilmiştir. Önerilen metotların maksimum tamamlanma zamanı performans ölçütüne göre performansları literatürde kabul görmüş yöntemler ile karşılaştırılmıştır. Sonuç olarak önerilen metasezgisellerin makul sürelerde kabul edilebilir sonuçlara ulaşabildiği ve PATÇP'lerin çözümünde kullanılabileceği görülmüştür.

Anahtar Kelimeler: El Bombası Patlatma Metodu, Metasezgisel Yöntemler, Permütasyon Akış Tipi Çizelgeleme Problemi.

SOLUTION OF FLOW SHOP SCHEDULING PROBLEMS WITH ENHANCED GRENADE EXPLOSION METHODS

ABSTRACT

In today's competitive conditions, businesses can only maintain their existence by meeting customer demands on time and at the least cost. For this reason, it is necessary to use resources efficiently and schedule the work in the best way. Scheduling is the decision process that determines when the work will be done in which source. Scheduling problems are divided into different classes. The permutation flowshop scheduling problem (PFSP), which has many applications in real life due to technical constraints, has been attracting the attention of researchers for more than half a century.

Grenade Explosion Method (GEM) is an evolutionary algorithm proposed by Ahrari et al., inspired by the explosions of grenades. The special feature that distinguishes GEM from other metaheuristics is the radius of the agent region. Thanks to this feature, agents (grenades) are prevented from approaching each other more than a certain distance, and the solution space is searched homogeneously.

In this study, firstly, GEM was adapted to solve permutation flowshop scheduling problems and the effect of agent region on the method performance was analysed. Then, a series of studies were carried out to improve the performance of the method, in this context, Discrete GEM (DGEM) and hybrid GEM_{RLS} has local search was developed. The performances of the proposed methods regard the criterion of maximum completion time were compared with the methods accepted in the literature. As a result, it has been seen that the proposed metaheuristics can produce acceptable results in reasonable time and they can be used in the solution of PFSPs.

Keywords: Grenade Explosion Method, Metaheuristics, Permutation Flowshop Scheduling Problem.

1. GİRİŞ

Günümüzün artan rekabet koşulları üreticileri; müşteri isteklerini eksiksiz yerine getirmeye, müşterinin talep ettiği hizmet veya ürünü, istenen kalite düzeyinde, düşük maliyetle ve talep edilen zamanda teslim etmeye zorlamaktadır. İstekleri eksiksiz karşılamak ancak çok iyi bir üretim planlaması ile mümkündür. Özellikle müşteri taleplerindeki değişkenlik kısa dönem ayrıntılı üretim planlarının ve iş çizelgelerinin yapılmasını güçleştirmektedir (Sağir ve Saraç, 2007). Bu durum çizelgeleme faaliyetlerinin önemini artırmaktadır.

Çizelgeleme hangi işin hangi makinede hangi sıra ile ne zaman yapılacağının belirlenmesi sürecidir. Bu sürçte belirli bir takım işleri yapmak için hangi kaynakların ne zaman ve nasıl kullanılacakları kararlaştırılır. Etkin bir çizelgeleme sayesinde işlerin daha az kaynak tüketimiyle veya daha kısa zamanda yapılma olanağı ortaya çıkmaktadır (Cerit ve diğ., 2005).

Akış tipi çizelgeleme; birbirinden farklı, m makine ve n işin bulunduğu; her bir işin m operasyondan oluştuğu, her bir işin farklı makinelerde yapıldığı ve bütün işlerin operasyonlarının aynı sıra ile yapıldığı problemlere denir ve bu problemler NP (non polinomial) problemler kapsamındadır (Engin ve Fıglalı, 2002).

Bu problemlerin her iş seri makinelerde tam olarak aynı sırayla işlem gördüğü özel hali ise permütasyon akış tipi çizelgeleme problemi (PATÇP) olarak adlandırılmaktadır (Kaya ve diğ., 2020).

Çizelgeleme problemleri genellikle iş ve makine sayıları, problem tipi ve performans ölçütü ile tanımlanır. Çizelgeleme problemlerinin gösteriminde kabul görmüş üçlü ve dörtlü olmak üzere 2 yöntem bulunmaktadır. Örneğin dörtlü gösterim ile $n/m/P/C_{maks}$ şeklinde formüle edilen bir problemde n iş sayısını, m makine sayısını, P problemin akış tipi olduğunu ve C_{maks} performans ölçütünün maksimum tamamlanma zamanı olduğunu belirtmektedir.

Akış tipi çizelgeleme problemleri üzerine bilinen ilk yöntem Johnson'a (1954) aittir. Johnson, sonlu n sayıda iş ve 2 makineden oluşan akış tipi çizelgeleme problemlerinde en küçük maksimum tamamlanma zamanını hesaplayan ve kendi adıyla bilinen bir

yöntem önermiştir. Daha sonra dal-sınır algoritmaları, beam arama vb. kesin çözümler oluşturan algoritmalar önerilmiştir (Yağmahan ve Yenisey, 2006).

Tamsayılı programlama gibi optimum çözümü garantileyen klasik yöntemlerin kullanımı küçük çaplı problemler için uygundur. Fakat geleneksel yöntemlerle optimum çözümü bulmak çok zaman aldığından büyük çaplı problemlerde optimum ya da optimuma yakın çözümleri elde edebilmek için sezgisel algoritmalar tercih edilir (Şevkli ve Yenisey, 2006).

Sezgisel algoritmalar, herhangi bir amacı gerçekleştirmek veya hedefe varmak için çeşitli alternatif hareketlerden etkili olanlara karar vermek amacıyla tanımlanan kriter veya bilgisayar metotlarıdır. Bu algoritmaların çözüm uzayında optimum çözüme yakınsaması ispat edilemez. Bu tür algoritmalar yakınsama özelliğine sahiptir. Fakat kesin çözümü garanti edemezler; sadece kesin çözüme yakın bir çözümü garanti edebilirler. Bu algoritmalar, problemin matematiksel modelinin kurulamaması ya da modelin kurulmasının zor olması, kesin çözüme ulaşmanın uzun zaman alması gibi nedenlerle tercih edilmektedir (Karaboğa, 2002).

Yapay Sinir Ağları, Tavlama Benzetimi (TB), Tabu Arama (TA), Genetik Algoritma (GA), Karınca Kolonisi Optimizasyonu (KKO), Parçacık Sürü Optimizasyonu (PSO) ve Diferansiyel Gelişim Algoritması (DGA) kullanımı yaygın olan sezgisel yöntemler arasındadır (Cura, 2006):

Bu çalışmanın ikinci bölümünde iş çizelgeleme problemlerine, üçüncü bölümünde permütasyon akış tipi çizelgeleme problemine, dördüncü bölümde el bombası patlatma metoduna, dördüncü bölümünde bulgular ve tartışmaya, beşinci bölümünde ise sonuç ve önerilere yer verilmiştir.

2. İŞ ÇİZELGELEME PROBLEMLERİ

Çizelgeleme günlük hayatta hepimizim karşılaştığı ve farkında olarak ya da olmadan bir şekilde çözdüğü bir karar problemidir. Gün içinde yapacağımız işlerin sıralarını, zamanlamalarını ve yerlerini bir şekilde kartlaştırırız. En azından evden çıkarken yapacağımız ilk işe karar vermiş oluruz. Bu kararı işlerin önem düzeyleri, bitirilmesi gereken tarihleri, zorluk dereceleri, yakınlığı vb. bilgileri bilinci veya bilinçsiz şekilde analiz ederek alırız. Bu bilgilerin bir kısmı vergilerin ödenme tarihîleri gibi yasal şartlardan, toplumsal kurallardan, hava durumu veya seyahat süresi gibi kontrolümüz dışındaki kaynak ve teknik kısıtlardan sağlanırken bir kısmı ise zamanını hata kendisini tamamen seçebildiğimiz veya değiştirebildiğimiz eylemlerimizden sağlanmaktadır.

Çizelgeleme faaliyetlerin zamanlamasına yönelik bir plan oluşturur ve her şey plana uygun ilerlerse, belli bir faaliyet ne zaman gerçekleşecek sorusuna cevap arar. Ne zaman sorusunun cevapları genellikle zamanlamayla ilgili bir bilgiyle gelir (Baker ve Trietsch, 2009).

Günlük yaşantımızın dışında üretim ve hizmet sektöründe de geniş bir uygulama alanı bulunan ve çok sayıda çalışmaya konu olan çizelgeleme, genel olarak görevler ve kaynakların eşleştirilmesidir.

Çizelgeleme, belirli bir dizi görevi yerine getirmek amacıyla kaynakların zaman içinde tahsisidir. Bu tanımda görevler ve kaynaklar içinde bulunulan ortama bağlı olarak değişik şekillerde adlandırılabilir (Pinedo, 2008). Görevler genel olarak, belirli ürün veya hizmeti meydana getirmek için gerekli olan faaliyetlerdir; atölyedeki işlemler, havalimanındaki iniş veya kalkışlar, inşaat projesindeki aşamalar, okulda verilmesi gereken dersler, hastanede uygulanacak tedaviler vb. olabilir. Kaynaklar ise bir atölyedeki makineler, bir havalimanındaki pistler veya hastanedeki hemşirelerdir.

Çizelgeleme problemleri hem üretim planlamada hem de hizmet planlamada işletmelerin karşısına çıkan önemli bir problemdir. Çizelgeleme problemleri, üretim planlamada üretim çizelgeleme, hizmet planlamada ise zaman çizelgeleme adı ile anılmakla birlikte (Palamutçuoğlu, 2022); çizelgeleme ile ilgili ilk çalışmalar çoğunlukla imalat sektöründe olduğu için kullanılan terimlerde genellikle bu sektörden gelmektedir. Bu sebeple bir

çizelgeleme problemi tanımlanırken genellikle; kaynaklar makineler, görevler ise işler olarak ifade edilir.

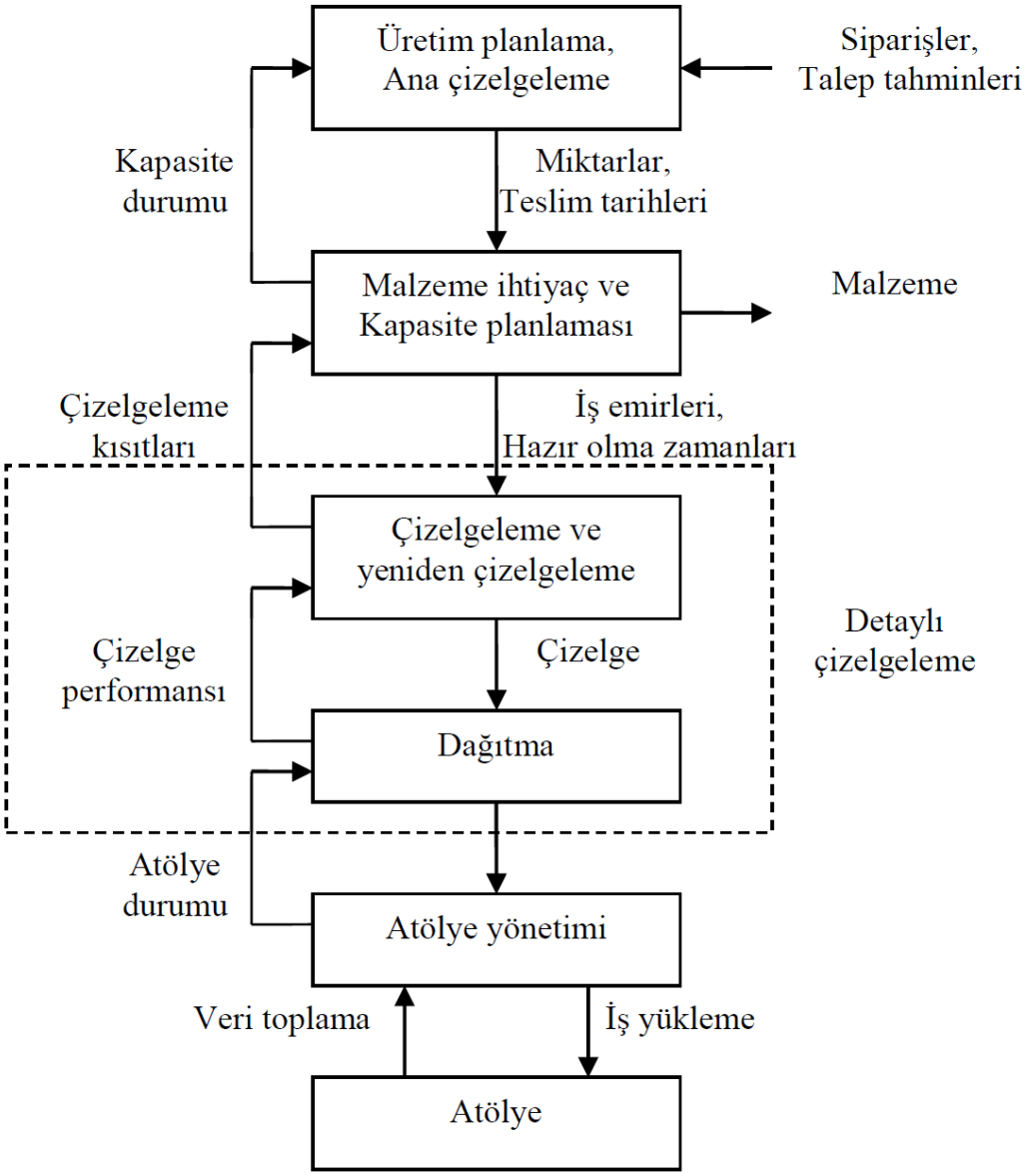
Çizelgeleme probleminde makine kapasitesi kısıtları, teknolojik kısıtlar olmak üzere iki tür kısıt vardır. Çizelgeleme problemi çözülürken, bu iki kısıt birlikte karşılanmalıdır. Eğer elde edilen çözüm, her bir iş için hangi kaynağın atanacağını ve her bir işin ne zaman yapılacağını gösteriyorsa geçerli bir çözümdür. Bu bağlamda, çizelgeleme problemi kısıtlara bağlı optimizasyon problemi olarak görülmektedir (Dağ, 2012).

Çizelgeleme kapasite ve önceliklendirme olmak üzere iki temel problemden oluşur. Kaynak belirli bir dönemde matematiksel veya sezgisel yöntemler kullanarak görevlere tahsis edilir. Sınırlı kaynakların uygun şekilde atanması, bir veya çoğu kez birbiri ile çelişen daha fazla amacı optimize ederek hedeflere en iyi şekilde ulaşmasını sağlar (Pinedo, 2008).

İşletmelerin müşteri taleplerine hızla cevap verebilmeleri, günümüzün ekonomik koşullarında rekabet edebilmeleri, hatta varlıklarını sürdürebilmeleri için bir zorunluluktur. Siparişlerin zamanında teslim edilememesinden kaynaklanan müşteri kayıpları işletmeler açısından önlenmesi gereken önemli bir sorundur. Bu doğrultuda işletmeler tarafından faklı çizelgeleme amaçları belirlenebilir. Son işin tamamlanma süresini veya teslim tarihinden sonra tamamlanan başka bir ifadeyle geciken işlerin sayısını azaltmak birer amaç olabilir. Benzer şekilde iş yükü dengesini sağlanmak da önemli bir hedeftir. İşler makinelere dengeli bir şekilde dağıtılarak, darboğazlar ortadan kaldırabilir. İşlerin dengeli dağıtımı üretim kapasitesini artırabilir, bitmiş ürün stoklarını ve üretim maliyetlerini azaltabilir (Rajakumar ve diğ., 2004) (Küçük, 2010).

Üretim çizelgeleme, kısa vadeli ana üretim programlarının konuları olan işlerin sıralanması, rotalanması, tezgahlara ve personele işlerin yüklenmesi şeklindeki bir planlama faaliyetidir. İşlerin nasıl sıralanacağı, hangi işin hangi makinalarda işlem göreceği, makina ve personellere hangi zaman periyodunda hangi işlerin atanacağı önceden belirlenen bir amaca veya amaçlara göre belirlenir. Bu amaç işlerin en kısa zamanda tamamlanması, siparişlerin teslim tarihlerinin en az ölçüde aşılması, makinaların ve personelin atıl zamanlarının minimize edilmesi şeklinde olabilir (Palamutçuoğlu, 2022).

Bir organizasyondaki çizelgeleme işlevi, Şekil 1.1'de de görüldüğü gibi, atölye düzeyinin yanı sıra orta ve uzun dönemli üretim planlama işlevinden de etkilenir. Üretim planlama işlevi, kaynak ihtiyaçları, talep tahminleri ve stok seviyelerini dikkate alarak uzun dönemli kaynak tahsisini ve firmanın ürün karışımını optimize etmeyi amaçlar. Bu yüksek planlama seviyesinde alınan kararlar çizelgeleme işlevini doğrudan etkiler. (Pinedo, 2008).



Şekil 2.1. İmalat Sisteminde Bilgi Akış Şeması (Aslan, 2022; Pinedo, 2008)

Çizelgeleme problemleri genellikle işlerin geliş şekli, makine sayısı, iş akış tipi ve performans ölçütü dikkate alınarak sınıflandırılmaktadır. Fakat farklı sınıflandırmalar da bulunmaktadır.

2.1. Çizelgeleme Problemlerinin Notasyonu ve İlgili Kavramlar

Çizelgeleme problemlerinin sonlu sayıda işten ve makineden oluştuğu varsayılmakta ve genellikle işlerin sayıları n ile makinelerin sayıları ise m ile gösterilmektedir. Araştırmacılar çoğunlukla makinler için i alt indisini, işlerin için j alt indisini kullanmaktadır. Eğer bir iş belli sayıda işlem basamağı veya operasyondan oluşuyorsa, (i, j) çifti j işinin i makinesindeki operasyonunu ifade eder. J işi ile ilgili bazı terim ve gösterimleri şu şekildedir (Pinedo, 2008; Aslan, 2022):

İşlem Süresi (p_{ij}): Makine i de, j işinin işlem süresi gösterir. Eğer j işinin işlem süresi, makineye bağlı değilse veya j işi sadece bir makinede işleniyorsa i indisi gösterilmeyebilir.

Geliş zamanı (r_j): j işinin hazır olma zamanını gösterir. j işinin işlenmeye başlanabileceği en erken zamandır.

Teslim zamanı (d_j): j işinin teslim zamanı, tamamlanma veya ürünün teslim zamanını gösterir. İş teslim zamanından sonra tamamlanırsa ceza oluşur. Bir işin teslim tarihi kesinlikle aşılamıyorsa bu zorunlu teslim tarihi (deadline) olarak bilinir ve $\bar{d}_j$ sembolüyle gösterilir.

Ağırlık (w_j): j işinin ağırlığı, j işinin sistemdeki diğer işler arasındaki oransal önemini gösterir. Örneğin, işin ağırlığı, işin maliyetinin işlerin toplam maliyetine oranı veya işe yüklenen değerin bir göstergesi olabilir.

Müsait zaman (a_j): j işinin geliş zamanı ile teslim tarihi arasında, işlenmeye uygun olduğu zamandır. Eşitlik (2.1) ile hesaplanır.

$$a_j = d_j - r_j \tag{2.1}$$

Bekleme süresi (W_{ij}): Bir işin önceki işlerin tamamlanmasını beklemesi veya bakım vb. nedenlerle makinenin meşgul olması durumunda ortaya çıkar. W_{ij}, j işinin i makinesinde işlem göremeye başlamadan beklediği süreyi gösterir.

Toplam bekleme süresi (W_j): j işinin bütün makinelerdeki toplam bekleme süresini ifade eder ve Eşitlik (2.2) ile hesaplanır.

$$W_j = \sum_{i=1}^{m} W_{ij} \tag{2.2}$$

Tamamlanma zamanı (C_j): Eşitlik (2.3) ile hesaplanan tamamlanma zamanı j işinin bittiği zamanı ifade eder.

$$C_J = r_j + \sum_{i=1}^{m} (W_{ij} + p_{ij}) \tag{2.3}$$

Akış zamanı (F_j): Eşitlik (2.4) de görüldüğü üzere tamamlanma zamanı ile geliş zamanı arasındaki fark olan F_j, j işinin atölyede geçirdiği süredir.

$$F_j = C_j - r_j \tag{2.4}$$

Geç kalma (lateness (L_j)): Bir iş teslim zamanından önce veya sonra bitebilir. İşin bitiş ve teslim zamanı arasındaki fark geç kalma olarak ifade edilir ve J işinin geç kalma süresi Eşitlik (2.5) deki gibi hesaplanır.

$$L_j = C_j - d_j \tag{2.5}$$

Gecikme (tardiness (T_j)): Bir işin tamamlanma zamanı teslim tarihinden geç ise gecikme ortaya çıkar. Eşitlik (2.6) da görüldüğü üzere gecikme geç kalmanın pozitif olduğu durumdur.

$$T_j = \max(L_j, 0) \tag{2.6}$$

Erken tamamlanma (E_j): Bir işin tamamlanma zamanı teslim tarihinden erken olmasıdır ve Eşitlik (2.7) ile hesaplanır. Büyük E_j değerleri yüksek stok maliyeti oluşturabilir.

$$E_j = \max(-L_j, 0) \tag{2.7}$$

Maksimum tamamlanma zamanı (makespan (C_{max})): Son işin bittiği zamandır ve Eşitlik (2.8) ile hesaplanır.

$$C_{max} = \max(C_j) \tag{2.8}$$

Geciken iş sayısı (U_j): Teslim tarihinden sonra biten işlerin sayısıdır.

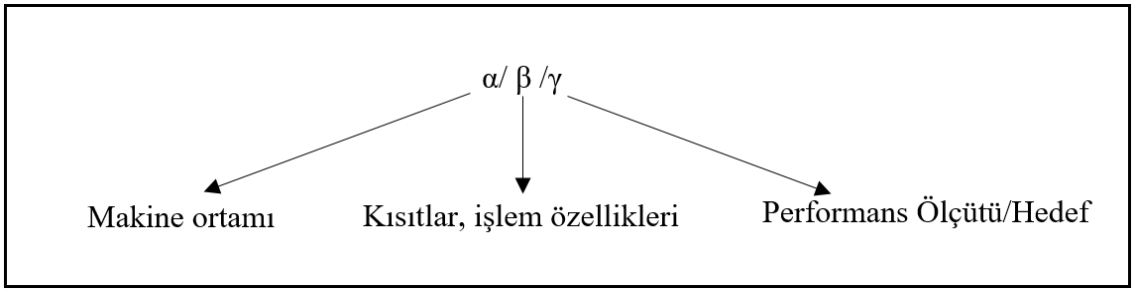
Atıl zaman (idle time (I_i)): i makinesinde boş geçen toplam süreyi ifade eden I_i, Eşitlik (2.9) kullanılarak hesaplanır.

$$I_i = C_{max} - \sum_{j=1}^{n} p_{ij} \tag{2.9}$$

Literatürde iş çizelgeleme problemlerinin gösteriminde üç parametreli ve dört parametreli olmak üzere iki farklı sistem kullanılmaktadır.

Dörtlü sistemde çizelgeleme problemleri α / β / γ / δ şeklinde dört parametre ile ifade edilir. Bu sistemde, işlerin geliş sürecini (işler arasındaki sürenin olasılık dağılımını) gösteren α parametresi, çizelgeleme probleminin karmaşıklığını ve belirsizliğini yansıtan önemli bir faktördür. Sadece n bulunduğunda sonlu sayıda makineye sahip statik bir problemi ifade eder. İkinci parametre, β, makine sayısını, üçüncü parametre atölyedeki işlerin akış düzenini (akış tipi, atölye tipi, vb.) gösterir. Son olarak δ parametresi ise performans ölçütünü belirtir. Örneğin; n/2/F/F_{max} Johnson'ın çok bilinen n işli ve 2 makineli, toplam akış süresinin minimize edildiği akış tipi çizelgeleme problemini tanımlar (Conway ve diğ., 1967).

Dörtlü sistem temel problemler için yeterlidir, fakat karmaşık (bağımlı işler, işlem bölme vb.) problemler için üçlü sistem daha uygundur (Aslan, 2022). Bu sistemde çizelgeleme problemleri Şekil 2.2'deki gibi α/β/γ şeklinde üç parametre ile gösterilir. Atölyenin düzenini (makine ortamını) α parametresi gösterir ve bu parametre tek bir girdiyle ifade edilir. β parametresi, işleme özellikleri ve kısıtlarıyla ilgili bilgi verir ve bu parametre ile problemin bir veya birden çok özelliği belirtilebilir. γ parametresi ise optimize edilecek performans ölçütünü başka bir ifadeyle amaç fonksiyonunu gösterir (Pinedo, 2008).



Şekil 2.2. Üçlü sistemde çizelgeleme problemlerinin gösterimi

Üçlü sistemde α parametresinin alabileceği bazı değerler şu şekildedir (Pinedo, 2008; Aslan, 2022):

Tek makine (1): Sadece bir makine bulunur, bu makine üzerinde optimum iş süresi bulunmaya çalışılır.

Paralel m adet özdeş makine (P_m): Aynı özelliğe sahip paralel m adet makine bulunur. j işi tek işlem gerektirir ve makinelerden herhangi birinde veya verilen alt kümeye ait herhangi bir makinede işlenebilir.

Farklı işleme hızlarındaki paralel m adet makine (Q_m): Farklı işlem hızlarına sahip m adet paralel makine vardır. Makine i'de j işinin süresi p_{ij}, p_j/v_i ye eşittir.

İlişkisiz paralel makineler (R_m): Bu ortam bir önceki ortamın genelleştirilmiş halidir. Paralel m adet farklı özelliklere ve işlem hızına sahip makine bulunur. Makine hızının işe göre değişmesi Q_m'den farklı olan yönüdür. j makinesi i işini v_{ij} hızıyla yapabilmekte olup harcanan süre p_i / v_{ij}'dir.

Akış tipi (Fm): Seri sıralanmış m adet makine bulunur. Her bir iş aynı rotayı izleyerek bütün makinelerde işlem görür. Eğer ilk gelen ilk çıkar kuyruk disiplini geçerli ise permütasyon akış tipi çizelgeleme oluşur ve bu özellik β alanında belirtilir. Bu durumda kuyrukta bekleyen işlerin sırası değişmez.

Esnek akış tipi (FFc): Akış tipi ve paralel makine ortamının genelleştirilmiş halidir. Bu makine ortamında, toplam c adet seri aşama ve her bir aşamada m adet benzer özelliklere sahip paralel makine bulunur. Her iş, belirli bir rotayı izleyerek aşamalardan geçer ve aşamaların her birinde sadece bir makinede işlenir. İşlerin her biri sırasıyla 1. aşama,

sonra 2. aşama ve daha sonra diğer aşamalarda işlenir. İşlerin önceliği bulunmaz. Bu özelliği nedeniyle çok işlemcili akış tipi veya melez akış tipi olarak da anılmaktadır.

Atölye tipi (J_m): m makine bulunan bir atölyeyi ifade eder. Her iş başlangıçta belirlenmiş bir rotaya sahiptir. İşlerin bir makineyi sadece bir kez veya daha çok ziyaret etmesine göre alt sınıflara ayrılır.

Esnek atölye tipi (FJc): Her aşamada c adet özdeş makine bulunan atölye tipidir. Aynı zamanda çok sayıda paralel makine bulunan, atölye tipinin (J_m) genelleştirilmiş halidir.

Açık atölye tipi (Om): Bu atölye tipinde m adet makine söz konusu olup her bir iş her bir makinede işlem görür. Fakat bazı işlerin bazı makinelerdeki işlem süreleri sıfır olabilir. İşlerin rotalarıyla ilgili herhangi bir kısıt yoktur. Her bir iş farklı rotaya sahip olabilir.

İş karakteristikleri ilgili olarak β parametresinin alabileceği bazı değerler şunlardır (Pinedo, 2008; Aslan, 2022):

Geliş (hazır olma) zamanı (r_j): işin başlama zamanı ile ilgili bir kısıtın bulunup bulunmadığını gösterir. Eğer (r_j) sembolü β alanında yer alıyorsa j işinin işlemi herhangi bir zamanda başlayabilir. Fakat bu sembol β alanında yer alıyorsa, j işinin geliş zamanından önce işleme başlayamaz.

Sıra bağımlı hazırlık süreleri (S_{jk}): Eğer çizelgede j işi k işinden önce geliyorsa S_{jk}, k işin başlayabilmesi için gereken hazırlık süresini gösterir. Eğer j ve k işleri arasındaki hazırlık süresi makineye de bağımlı ise o zaman S_{ijk} ile ifade edilir. Eğer β alanında hiç S_{jk} yoksa tüm hazırlık süreleri sıfır veya kuyruğun bağımsız olduğu anlamına gelir.

Bölünebilme (prmp): İşlerin tamamlanana kadar makinede kalmasının zorunlu olmadığı anlamına gelir. Her hangi bir zamanda her hangi bir işin işlenmesine ara verilebilir ve makineye farklı bir iş yerleştirilebilir. İşlemi yarıda kesilen iş ilgili makineye tekrar yerleştirildiğinde sadece kalan süre kadar işlem görmesi yeterlidir.

İş aileleri (fmls): Benzer özelliklere sahip işler bir ailede toplanır. Bu sembol aynı aileden işlerin sırayla hazırlık zamanına ihtiyaç duyulmadan işlenebileceğini ifade eder. Fakat bir iş ailesinden diğerine geçilirken makinenin ayarlanması, kalıpların değişmesi vb. nedenlerle hazırlık zamanına ihtiyaç duyulur.

Öncelik kısıtları (prec): Belirli işin işlenmeye başlanması için önce bir veya birden çok işin tamamlanması gerekir. Bu durum teknik kısıtlardan kaynaklanabilir veya öncelikli müşterilerin işleri bitmeden diğer işlerin başlamasına izin verilmeyebilir. Bir işin hemen başka bir işten sonra başlaması da gerekebilir.

Arızalar (brkdwn): Makinenin arıza nedeniyle uygun olmadığını ifade eder.

Makine atama kısıtları (M_j): Bu sembolün β alanında yer alması paralel özdeş makine bulunduğunu fakat her makinenin j işini işleme kapasitesine sahip olmadığını anlatır. M_j j işini işleme kapasitesine sahip makine setini ifade eder. Eğer β alanında M_j bulunmazsa, j işi herhangi makinede işlem görebilir.

Permütasyon (prmu): Akış tipinin özel bir halidir. Bu sembol, makineler arasındaki kuyruk disiplininin ilk gelen ilk işlem görür (FIFO) olduğunu gösterir. Bu durumda işler tüm makinelerde ilk makineye girdiği sıra ile işlenmek zorundadır.

Bloklanma (block): Makineler arasındaki bekleme alanının kapasitesi sınırlı olduğunda akış tipinde bu durum ortaya çıkabilir. Bu durumda, makine tamamladığı işi sonraki makineye gönderemez.

Beklemesiz (nwt): Teknik kısıtlar nedeniyle akış tipinde yer alan bu parametre, işlerin seri iki makine arasında bekleyemeyeceğini ve kuyruk disiplininin FIFO olduğunu ifade eder. Kuruma, bozulma gibi teknik nedenlerle işin makinleler arasında beklemesinin mümkün olmadığı üretim ortamlarında bu parametre kullanılır.

Yeniden dolaşım (recrc): atölye veya esnek atölye tipi problemlerde karşılaşılan bu parametre, her hangi bir işin her hangi bir makineyi birden çok ziyaret edebileceğini gösterir.

Çizelgeleme problemlerinde genellikle bir veya birden çok performans ölçütü en küçüklenmeye çalışılır. j işinin i makinesinde tamamlanma zamanı C_{ij} ve sistemde geçirdiği zaman C_j olarak gösterilir. Performans ölçütü aynı zamanda teslim zamanının (d_J) bir fonksiyonu da olabilir. Performans ölçütünü diğer bir ifadeyle amaç fonksiyonunu gösteren γ parametresinin alabileceği bazı değerler şunlardır (Pinedo, 2008; Aslan, 2022):

Maksimum tamamlanma süresi (C_{max}): Sistemdeki son işin bitiş zamanıdır. Bitiş zamanın minimum olması makinelerin kullanımının yüksek olduğu anlamına gelir.

Maksimum pozitif geç kalma süresi (L_{max}): L_{max}, $\max(L_1,....,L_n)$ olarak tanımlanır. L_{max} ihlal edilen bitiş tarihinin en kötüsünü ölçer.

Toplam ağırlıklı tamamlanma zamanı ($\Sigma w_j C_j$): İşlerin önem dereceleri ve tamamlanma zamanlarının bir fonksiyondur ve Eşitlik (2.10) kullanılarak hesaplanır. İşlerin neden oldukları elde bulundurma ve stok maliyetleri eşit olmayabilir. İşlere bu vb. maliyetlere göre ağırlıklar verilebilir. Başka bir ifadesiyle ağırlıklı akış zamanı stok veya elde bulundurma maliyetleri hakkında bilgi verir.

$$\sum w_j \,(C_j - r_j = \sum w_j C_j - \sum w_j r_j \qquad (2.10)$$

Toplam ağırlıklı gecikme ($\Sigma w_j T_j$): Sistemdeki her iş aynı önem düzeyine sahip olmayabilir. Bazı işlerin gecikmesi tolere edilebilirken bazı işlerin gecikmesi çok daha büyük kayıplara neden olabilir. Toplam ağırlıklı gecikme; işlerin teslim tarihlerine göre gecikme sürelerinin işlerin önem dereceleri ile çarpımlarının toplamıdır. Bu sayede sadece gecikme süreleri değil işlerin önem dereceleri de değerlendirmeye alınır.

Geciken işlerin ağırlıklı sayısı ($\Sigma w_j U_j$): Geciken işlerin önemlerinin yani ağırlıklarının toplamıdır.

Yukarıda verilenler çizelgeleme problemlerinde yağın olarak kullanılan amaç fonksiyonlar olup bunların sayılarını artırmak mümkündür. Ayrıca literatürde birden çok amacı en iyilemeye çalışan çok amaçlı çizelgeleme problemlerine yönelik çalışmalar da yaygındır.

2.2. Çizelgeleme Problemlerinin Grafiksel Gösterimi

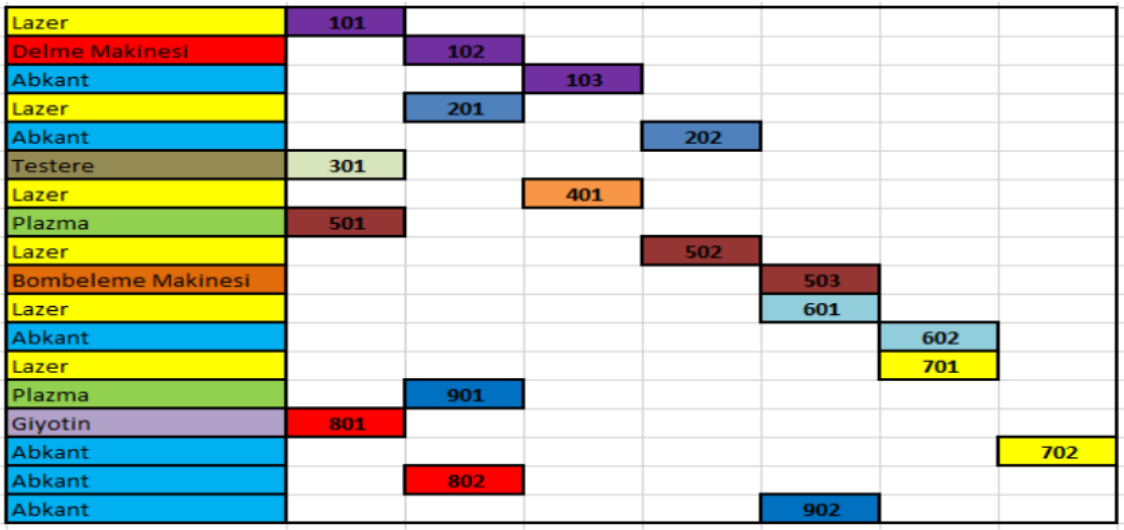
Çizelgeleme problemlerinin görselleştirilmesi oluşturulan çizelgelerin analizi, performansının ölçülmesi ve karşılaştırılmasında önemli bilgiler sağlanmaktadır. Problemlerinin grafiksel gösterimi ile kaynakları işlere nasıl tahsis edilgi ve boşta kalan kaynak ve zamanların daha kolay şekilde anlaşılmaktadır. Çizelgeleme problemlerinin

grafiksek gösterimi genellikle Gantt şemaları veya ayrık graflar kullanılarak yapılmaktadır.

2.2.1. Gantt Şeması

Henry Gantt tarafından 1900'lerin başında geliştirilen Gantt şemaları çizelgelerin gösteriminde kullanılan en yaygın ve basit modelleridir. Birçok türü olan Gantt şeması temel şekliyle kaynakların zamana bağlı dağılımını göstermektedir. Bu şemada kaynaklar dikey eksende, zaman ölçeği ise yatay eksende verilir. Temel Gantt şeması iş sürelerinin kesin olarak bilindiği varsayılarak çizilir (Baker ve Trietsch, 2009).

Esasen bir bar grafik olan Gantt Şemasında; yatay eksen süreyi, dikey eksen ise ilgili aktiviteler, makineler, çalışanlar ya da diğer kaynakları gösterir. Barlar yani çubuk diyagramlar, yükleme süreleri veya aktivitelerin başlama ve bitiş sürelerini göstermek için kullanılır. Gantt Şeması karmaşık çizelgelemeyi açıklayan görsel bir özettir (Nahmias, 2001).



Şekil 2.3. İş odaklı Gantt şeması (Kaçar ve Olgun, 2020)

Gantt diyagramı iş odaklı ve makine odaklı olmak üzere ikiye ayrılır. İş odaklı Gantt şemasında makineler yerine işlere odaklanır. Herhangi bir ürünün üretim aşamasındaki her işlemin üretim merkezlerinde ne kadar süre harcadıkları, üretim merkezleri arasındaki izledikleri rota ve hangi üretim merkezinde ne zaman işlem görmeye başlayıp

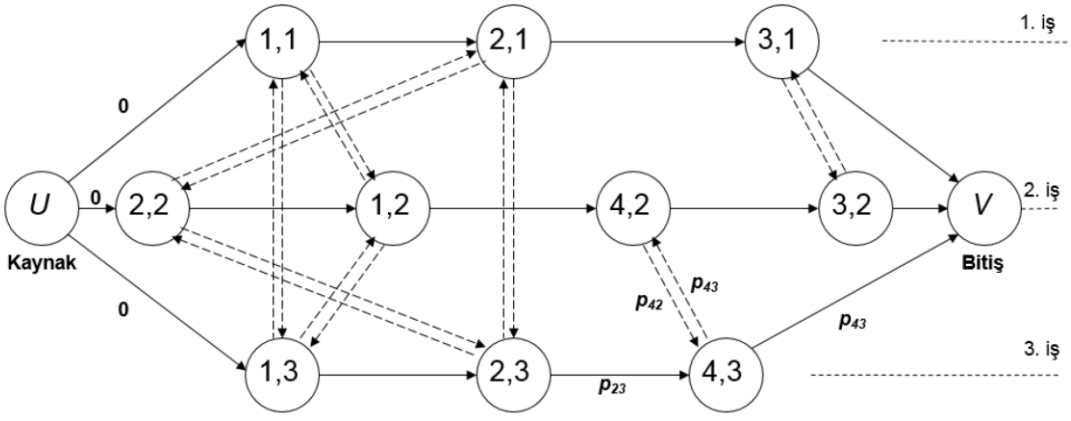
tamamlandığı incelenmektedir. Makine odaklı Gantt diyagramında ise işlemlerin yanı sıra makinelere odaklanılır. X ekseninde zaman, Y ekseninde ise makineler bulunur. Zamana göre makine yükleme işlemi gerçekleştirilir. Gantt şemalarının daha anlaşılabilmesi için Şekil 2.3'te iş odaklı ve Şekil 2.4'de makine odaklı Gantt şemaları temsili olarak gösterilmiştir (Kaçar ve Olgun, 2020).

Lazer	101	201	401	502	601	701	
Delme Makinesi		102					
Abkant		802	103	202	902	602	702
Testere	301						
Plazma	501	901					
Bombeleme Makinesi					503		
Giyotin	801						

Şekil 2.4. Makine odaklı Gantt şeması (Kaçar ve Olgun, 2020)

2.2.2. Graf Gösterimi

İş çizelgeleme problemleri graf teorisinden faydalanılarak da görselleştirilebilir. Graf teorisinde bağıntı şekle dönüştürülerek problem görselleştirilir ve anlaşılması kolaylaştırır. Görseldeki ilişkiler matris şeklinde ifade edilerek problem elle veya bilgisayar yardımıyla çözülebilir. Graf teorisinde problem düğüm ve çizgiler kullanılarak ifade edilir. Köşelerin düğüm (D), kenarların ise ayrıt (A) olarak tanımlandığı grafın en genel matematiksel ifadesi G=(D, A) şeklindedir (Bayzan, 2009).

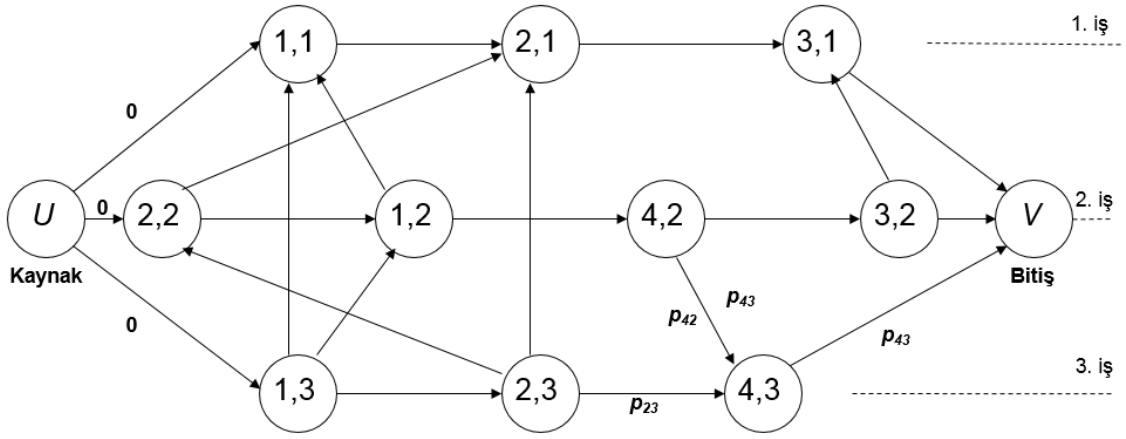


Şekil 2.5. Ayrık Graf (Aslan, 2022).

Çizelgeleme problemlerinde düğümler işleri, ayrıtlar ise rotaları temsil eder. İşlem süreleri (p_{ij}) ayrıtların ağırlığını gösterir. Ayrıca grafa gerçekte olmayan kaynak ve bitiş düğümleri eklenebilir. Problem türüne göre faklı graflar tercih edilmekte olup Şekil 2.5'te atölye tipi çizelgeleme probleminin gösteriminde kullanılan ayrık graf örneği verilmiştir

Ayrıt (çizgi) çiftlerinden biri çevrimsiz yönlendirilmiş bir graf oluşturacak şekilde seçilerek uygun bir çizelge oluşturulur. Yönlendirilmiş grafta ayrıtlar üzerindeki oklarla düğümler arasındaki sıra da gösterildiğinden bu seçimle, işlerin her bir makindeki işlem sıraları da belirlenmiş olur. Oluşan grafta kaynaktan bitişe olan en uzun yol maksimum tamamlanma zamanına (C_{max}) eşittir ve kritik yol olarak adlandırılır (Pinedo, 2008). Şekil 2.6'da uygun bir çizelge örneği gösterilmektedir.

Graflar kullanılarak iş çizelgelerinin görselleştirilmesi işlerin önceliklerinin ve akış üzerindeki etkilerinin daha kolay anlaşılmasını sağlar. Küçük boyutlu problemler graf teorisinden faydalanılarak optimize edilmektedir. Fakat büyük boyutlu problemlerde karmaşık ve çok büyük graflara ihtiyaç duyulduğundan grafların etkinliği azalmaktadır.



Şekil 2.6. Uygun bir çizelge (Aslan, 2022)

2.3. Çizelgeleme Problemlerinin Sınıflandırılması

Klasik atölye çizelgelemeden iş gücü ve hizmet planlamaya, ürün montaj hattı dengelemeden, lojistik kaynak ataması ve çizelgelemesine kadar birçok alanda uygulaması bulunan çizelgeleme problemleri çeşitli faktörlere göre değişik şekillerde sınıflandırılabilmektedir. İhtiyaç üretimi, üretim ortamı, işlem karmaşıklığı, makine

sayısı problemin doğası, stratejileri, performans ölçütleri ve parametrelerin belirliliği Çizelgeleme problemlerinin sınıflandırılmasında göz önünde bulundurulan başlıca faktörlerdir (Palamutçuoğlu, 2022).

İhtiyaç üretimine, diğer bir ifadeyle uygulanan stok politikasına göre çizelgeleme problemleri; açık atölye tipi çizelgeleme ve kapalı atölye tipi çizelgeleme şeklinde iki sınıfa ayrılır. Stoğa üretim yapılıyor kapalı atölye, müşteri talepleri stok yapmadan direkt üretimden karşılanırsa açık atölye olarak adlandırılır. Kapalı ve açık atölye tipleri iki ucu temsil etmektedir, bunların karışımından oluşan melez problemler de mevcuttur.

Çizelgeleme problemleri, işlerin geliş şekline göre statik ve dinamik problemler olarak iki sınıfa ayrılabilmektedir. Bu sınıflandırmada; çizelgelenecek iş kümesi zamana bağlı olarak değişmiyor ise problem statik, değişiyorsa dinamik olarak nitelendirilmektedir (Kaçar ve Olgun, 2020). Başka bir ifadeyle; işler sisteme aynı anda giriş yapıyorsa ve zamanla değişiklik göstermiyorsa statik, işler sisteme farklı zamanlarda parça parça giriş yapıyorsa ve zamanla yeni işler ortaya çıkıyorsa çizelgeleme dinamiktir (Conway ve diğ., 2003).

Diğer bir sınıflandırma faktörü ise problemin yapısı yani parametrelerin belirliliğidir. Bu faktöre göre çizelgeleme problemleri stokastik (olasılıksal) ve deterministik olarak iki gruba ayrılır. Eğer işlem süreleri, hazırlık süreleri, teslim tarihleri gibi parametreler önceden kesin olarak biliniyorsa deterministik problem, eğer parametreler kesin olarak bilinmiyor ve bir olasılık dağılımına uyuyor ise stokastik problem olarak nitelendirilir.

Çizelgeleme problemleri temelde makine sayılarına göre tek makine ve çok makine problemleri olarak ikiye ayrılmaktadır:

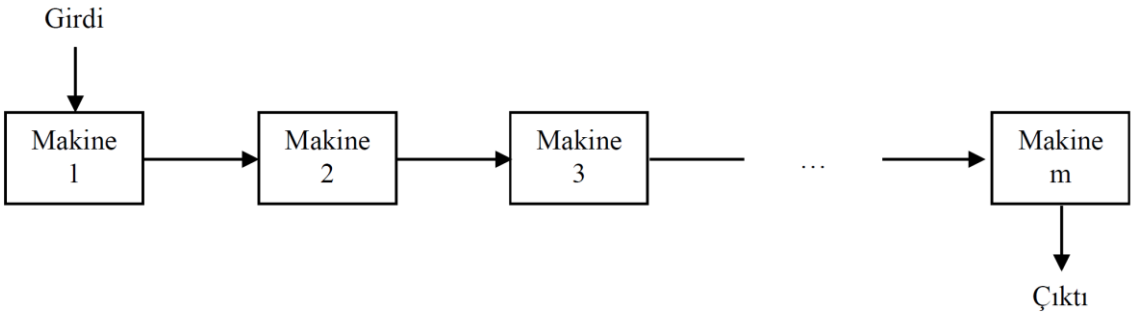
Tek makine problemi: Bütün işlerin tek makinede veya kaynakta gerçekleştirildiği en basit çizelgeleme problemidir. Tek makine modelleri çeşitli nedenlerle önemlidir. Tek makine problemi çok makinden oluşan problemlerin özel bir durumudur. Tek makine problemlerinde elde edilen sonuçlar, ilerleyen zamanlarda daha karmaşık problemlere uygulanabilecek sezgisel yöntemler için temel sağlayabilir. Uygulamada, daha karmaşık çok makineli çizelgeleme problemleri tek makineli alt problemlere bölünerek çözülebilir. Örneğin, sadece bir darboğaza sahip çok makineli bir problem, tek makine problemine

dönüşebilir (Pinedo, 2008). Bazen de boya fabrikalarında belirli bir zamanda tek bir rengin üretilmesi gibi büyük bir fabrika tekbir makineymiş gibi davranabilir.

Çok makine çizelgeleme problemi: Gerçek hayat çizelgeleme problemlerinde çoğunlukla birden fazla makine mevcuttur. Makineler paralel, seri ya da her ikisi şekilde konumlandırılabilir. Paralel çizelgelemede her bir iş herhangi bir makinede gerçekleştirilebilir ve işlem süresi makineden bağımsızdır, yani makineler özdeştir. Çizelgeleme; hangi işin hangi makinede ve hangi sırayla yapılacağı kararlarını içerir. (Kadı, 2020).

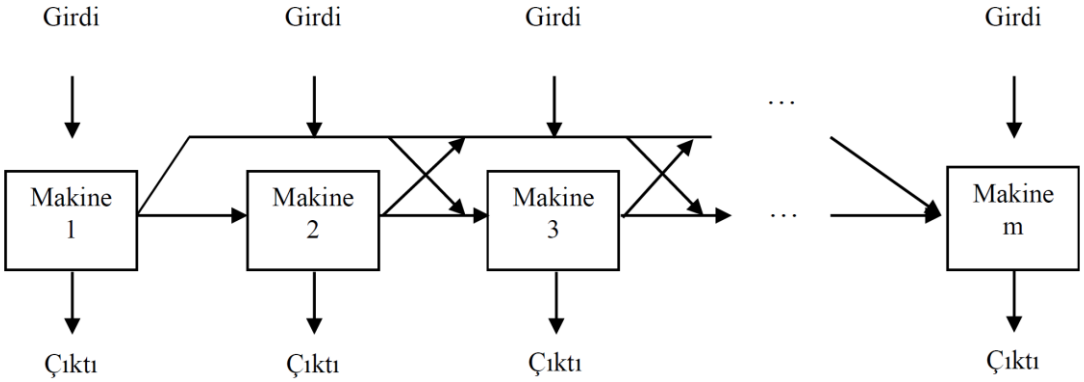
Çizelgeleme problemleri yerleşim tipilerine veya üretim ortamının yapısına göre akış tipi ve atölye tipi olarak iki gruba ayrılabilir. Esasen bu iki grup üretim sistemlerinin iki uç noktasıdır. Bu iki uç üretim ortamı arasında faklı yerleşim düzenleri oluşturulabilir ve bu yerleşim düzenlerinin özelliklerine uygun çeşitli çizelgeleme problemleri tanımlanabilir. Bu bağlamda çizelgeleme problemlerini; akış tipi üretim çizelgeleme (flow shop scheduling), atölye tipi üretim çizelgeleme (job shop scheduling), açık atölye tipi üretim çizelgeleme (open job shop scheduling), parti üretim çizelgeleme (batch shop scheduling), parti/akış tipi üretim çizelgeleme (batch/flow shop scheduling), imalat hücresi çizelgeleme (manufacturing cell scheduling), montaj atölyesi çizelgeleme (assembly shop scheduling), montaj hattı çizelgeleme (assembly line scheduling), iletim hattı çizelgeleme (transfer line scheduling) ve esnek iletim hattı çizelgeleme (flexible transfer line scheduling) olarak 10 sınıfta inceleyen çalışmalar bulunmaktadır (Palamutçuoğlu, 2022). Literatür incelendiğinde yerleşim tiplerine göre çizelgeleme problemlerinin genellikle; akış tipi çizelgeleme problemleri, esnek (hibrit) akış tipi problemleri, atölye tipi çizelgeleme problemleri ve açık atölye tipi çizelgeleme problemleri başlıkları altında incelendiği görülmektedir.

Akış tipi yerleşimde, birbirine seri bağlı m adet makine bulunur ve işler aynı rotayı izleyerek makinelerde işlem görür. Akış tipi yerleşime sahip üretim sistemleri genellikle tek bir ürünün veya birbirine benzeyen bir ürün grubunun üretimi için tercih edilir. İş istasyonları, ürünün üretimine uygun işlem basamaklarına göre yerleştirilir. Özel amaçlı makineler veya iş istasyonları, üretim akışına göre sıralanır ve iş akışı düzgündür (Kaya ve diğ., 2020). Şekil 2.7'de tüm işlerin makinelerde sadece bir defa işlendiği klasik akış tipi üretim ortamındaki iş akışı temsili olarak gösterilmiş.



Şekil 2.7. Klasik akış tipi yerleşimde iş akışı (Aslan, 2022)

Akış tipinin daha genel bir durumu Şekil 2.8'de gösterilmiştir. Şekilde de görüldüğü üzere işler tüm makinelerde işlenmek zorunda değildir. Bu aynı zamanda işlerin tüm ardışık makinelerde işlem görmesinin gerekmediği anlamına da gelmektedir. Bazı makinler atlanabilir, atlanan makine ilk veya son makine de olabilir. Ancak, iş akışı tek yöndedir, işler geri dönemez. Genel akış tipi problemi, bazı işlem sürelerinin sıfır olduğu klasik akış tipi çizelgeleme problemi şeklinde de ifade edilebilir (Baker ve Trietsch, 2009).



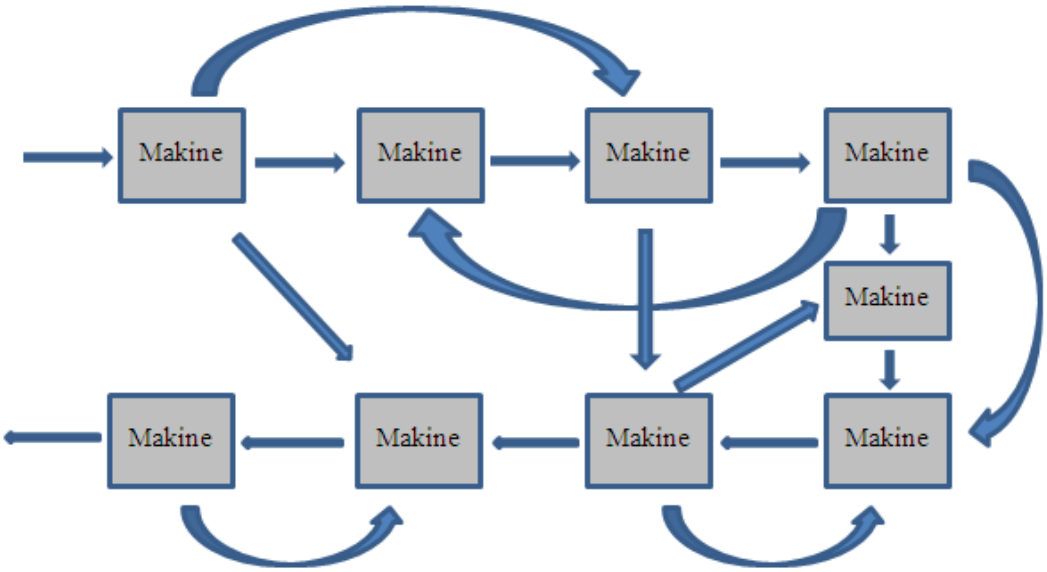
Şekil 2.8. Genel akış tipinde iş akışı (Aslan, 2022)

Akış tipi çizelgelemede problemlerinde, iş sıralarının her makinede farklı olduğu çizelgeler ve iş sıralarının her makinede aynı olduğu permütasyon çizelgeler olmak üzere iki farklı çizelgeleme türü bulunmaktadır. Bir iş birinci makineden başlayarak bütün makinelerde tam olarak aynı sıra ile işlem görüyorsa permütasyon akış tipi çizelgeleme problemi olarak adlandırılır (Çiçekli ve Bozkurt, 2016).

Akış tipi çizelgeleme problemlerinin sıralı akış tipi (ordered flow shop), ara stoklara izin veren tipi (flow shop with blocking) ve beklemeye izin vermeyen tipi (no-wait flow shop) gibi çeşitli alt sınıfları bulunmaktadır (Baker ve Trietsch, 2009; Göçmen ve diğ., 2010).

Esnek (hibrit) akış tipi çizelgeleme problemi: En az biri paralel olarak iki veya daha fazla makineden oluşan bir dizi üretim aşamasından oluşur. Tüm işler aynı işlem rotasını izler ve işler her aşamada yalnızca bir makine tarafından işlenerek bir aşamadan diğerine akarlar (González-Neir ve diğ., 2016). Üretim kapasitesini artırmak için üretim aşamalarına yeni makineler eklenmesi hibrit akış tipi problemlerinin oluşmasına neden olur (Madenoğlu, 2020). İlaç, bilgisayar, kağıt, kozmetik gibi birçok sektörde bu akış tipi ile bulunabilir.

Atölye tipi çizelgeleme (ATÇ) problemi: Genel atölye tipi çizelgeleme problemi, sonlu sayıda m adet makinede işlenmek üzere yine sonlu sayıda n tane işi, önceden belirlenen bir sıra ve kapasite kısıtlarını yerine getirerek, amaç fonksiyonunu optimum kılacak şekilde her bir işlemin başlama zamanını belirlemek olarak tanımlanabilir. Bu problemde her bir işin makinelere uğrayacağı sıra farklıdır ve iki tip kısıtlamadan söz edilebilir. Birincisi, bir işin bir operasyonu bitmeden diğer operasyonunun başlayamamasıdır. Aynı zamanda bir iş aynı anda sadece bir tezgah tarafından işlenebilir. İkincisi ise bir tezgah aynı anda sadece bir işin bir operasyonunu gerçekleştirir (Şevkli ve Yenisey, 2006).



Şekil 2.9. Atölye tipi üretim sistemi örnek gösterim (Kadı, 2020)

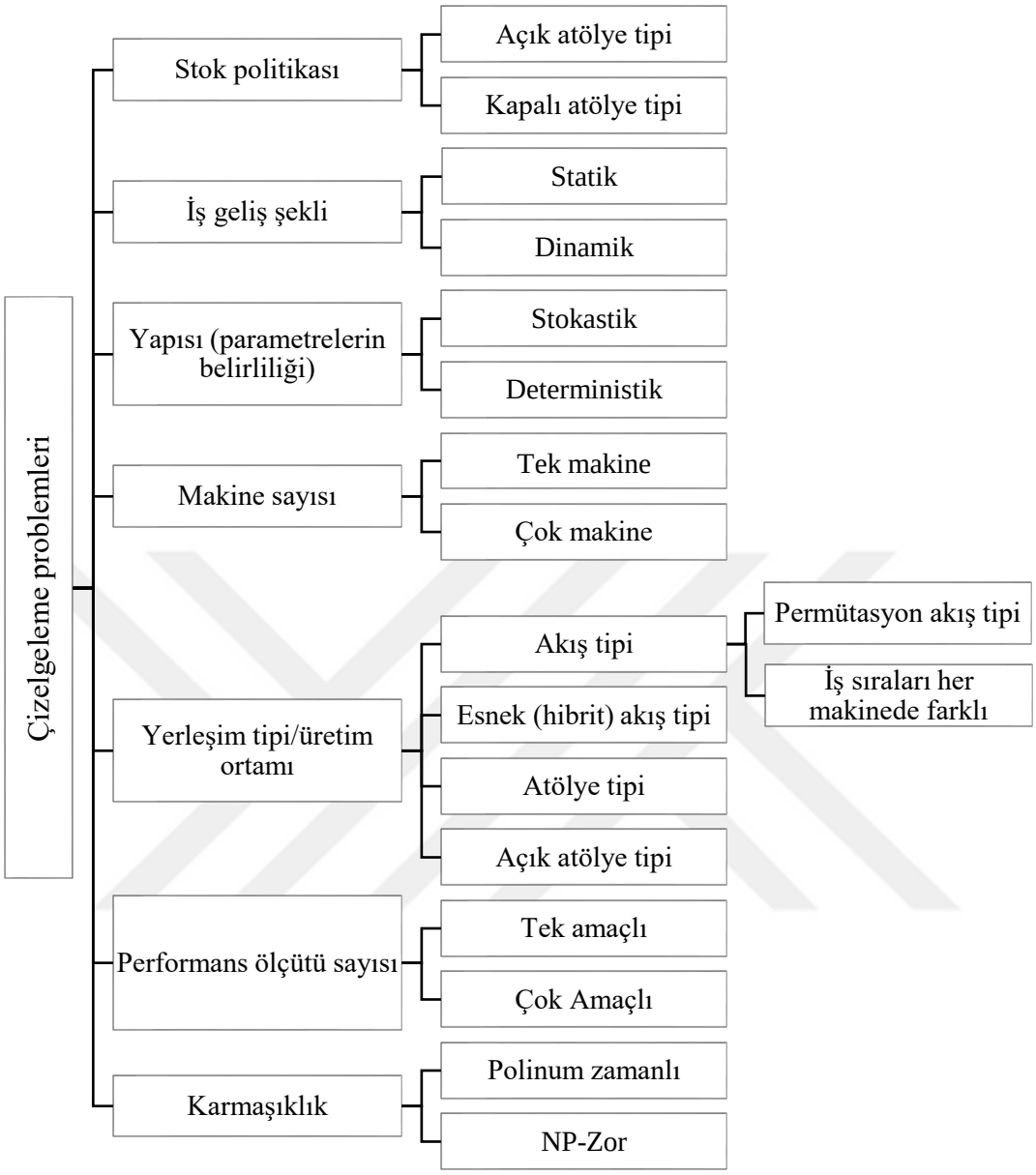
Atölye tipi üretim sistemi örnek gösterimi Şekil 2.9'da verilmiştir. Şekilde görüldüğü üzere işler aynı makinede birden çok kez işlem görebilir. Atölye tipi çizelgeleme problemi işlerin her makineye bir kez uğradığı ve birden çok uğradığı biçiminde iki sınıfa ayrılabilir. İkinci tür problemler tekrarlı (rcrc) akış tipi olarak bilinir.

Açık (serbest dolaşımlı) atölye tipi çizelgeleme (AATÇ) problemi: Atölye tipi çizelgeleme probleminin özel bir tipi olarak açıklanabilir. ATÇ problemlerinde, bir işin tamamlanması için belirli makine veya makine gruplarında gerçekleştirilmesi gereken işlemlerin, belirli bir sırası vardır. AATÇ problemlerinde ise işlem sırası önemsiz yani serbesttir (Baysal ve diğ., 2012). AATÇ'de işler arasında önceden belirlenmiş bir sıralama yoktur, bu yüzden de benzer şartlarda atölye tipi çizelgeleme problemine kıyasla AATÇ probleminde çok daha geniş bir çözüm uzayı mevcuttur (Kadı, 2020; Abraham ve Xhafa, 2008).

Çizelgeleme problemleri performans ölçütünün sayısına ve türüne göre de sınıflandırılabilir. Performans ölçütünün sayısına tek amaçlı veya çok amaçlı olarak iki gruba ayrılan çizelgeme problemlerinde maksimum tamamlanma zamanı ve toplam akış zamanı başta olmak üzere birçok performans ölçütü amaç fonksiyonu olarak kullanılabilmektedir. Son yıllarda yeşil çizelgeleme başlığı altında enerji tüketimini en küçüklmeyi hedefleyen çalışmaların sayısının artığı gürülmektedir.

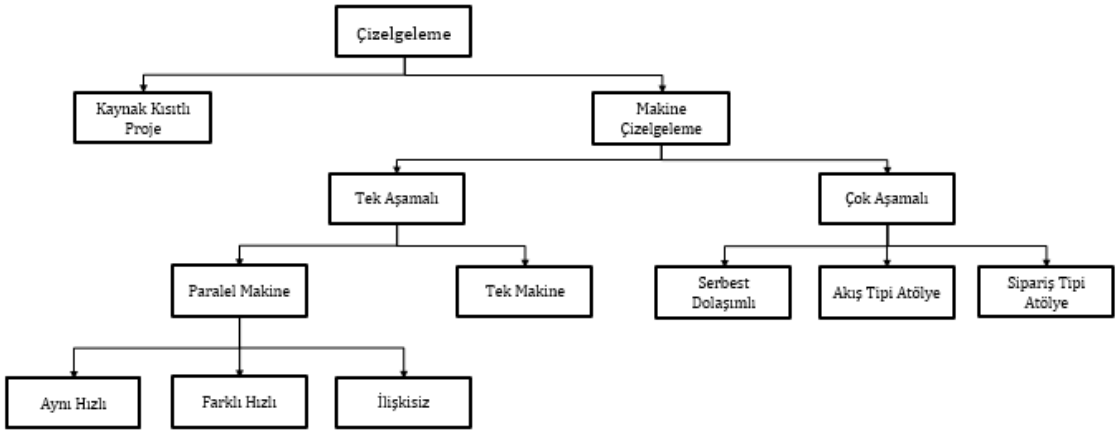
Çizelgeleme problemlerinin sınıflandırılmasında dolaylı olarak göz önüne alınan bir diğer faktör de problemin karmaşıklığıdır. Deterministik çizelgeleme problemleri karmaşıklığına göre polinom zamanlı çözülebilir ve polinom zamanlı çözülemeyen olarak iki ana sınıfa ayrılabilir. İkinci sınıftaki problemler NP-Zor (Non-deterministic Polinomial Hard) olarak tanımlanırlar (Pinedo, 2008). Problemin karmaşıklık bakımından hangi sınıfa girdiği problemin çözüm yöntemini belirlerken göz önünde bulundurulması gereken önemli bir faktördür.

Yukarıda verilen bilgiler ışında çizelgeleme probleminin sınıflandırılması Şekil 2.10'da özetlenmiştir. Şekil 2.10'da verilen sınıflandırma literatürde yaygın olarak kullanılan ortak özelliklere göre yapılmıştır fakat iteratürde yer alan tüm sınıflandırmaları içermemektedir. Literatürde genellikle bir veya birakç faktöre göre yapılmış farklı sınıflandırma şemaları bulunmaktadır.



Şekil 2.10. Çizelgeleme problemlerinin sınıflandırılması

Kaçar ve Olgun (2020) çizelgeleme problemlerini öncelikle kaynak kısıtlı proje çizelgeleme ve makine çizelgeleme şeklinde ikiye sınıfa ayırmışlardır. Daha sonra makine çizelgeleme problemlerini tek ve çok aşamalı problemler olarak iki gruba ayırmışlardır. Tek aşamalı problemlerde her işin tek bir işlemden oluştuğu kabul edilirken çok aşamalı makine çizelgeleme problemlerinde bir işin yapılması için çok sayıda işlemin yapılması gerektiği kabul edilmiştir. Bu şekilde sınıflandırmaya devam edilerek Şekil 2.11' de verilen hiyerarşik sınıflandırma oluşturulmuştur.



Şekil 2.11. Çizelgeleme problemlerinin genel sınıflandırılması (Kaçar ve Olgun, 2020)

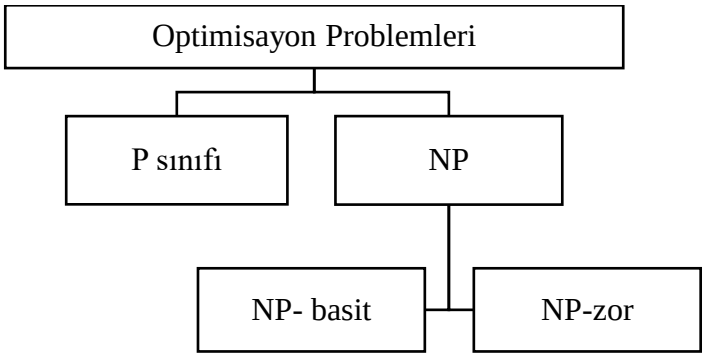
2.4. Çizelgeleme Problemlerinin Karmaşıklığı ve Çözüm Yaklaşımları

Karmaşıklık teorisindeki gelişmeler, çizelgeleme problemleri ve bunların çözüm yöntemleri arasındaki ilişkinin anlaşılmasına yönelik önemli bir bakış açısı sunar. Karmaşıklık kavramı, bir çözüm algoritmasının gerektirdiği hesaplama çabasını ifade eder (Baker ve Trietsch, 2009).

Hesaplama çabası, büyüklük sırası gösterimi ile tanımlanır. Örneğin, boyutu n (Teknik olarak n, sorunu belirlemek için gereken bilgi miktarını belirtir.) olan bir problemi çözmek için belirli bir algoritma kullandığımızı varsayalım. Algoritmada ihtiyaç duyulan hesaplama sayısı tipik olarak yukarıdan n'nin bir fonksiyonu ile sınırlanır. Karmaşıklık bakımından algoritmalar öncelikle polinom zamanlı olanlar (P) ve olmayanlar (NP) şeklinde ikiye ayrılır. NP sınıfı problemler de kendi içinde NP-zor ve NP- basit olarak iki başlıkta incelenebilir (Baker ve Trietsch, 2009).

Problemdeki girdilerin boyutu x ile gösterilirse ve y de sabit bir sayı ise; çözümü için $O(x^y)$ hesaba ihtiyaç duyulan bir problem polinom karmaşıklığa sahiptir. Eğer problemin çözümünde girdi sayısına bağlı, üstel olarak artan hesaplama adımlarına ihtiyaç duyuluyorsa problemin karmaşıklığı $O(y^x)$ şeklinde ifade edilir ve problem NP-zor sınıfındadır. Bazı NP sınıfı problemler tahmin edilebilecek cevaplara sahiptir. Bu grupta yer alan ve tahmin edilen cevabın doğruluğunu teyit etmek için girdi boyutuna bağlı olarak polinom mertebesinde hesaplama süresine sahip algoritmalar geliştirilebilen

problemler NP-basit olarak bilinir (Aslan, 2022). Şekil 2.12'de optimizasyon problemlerinin zorluk açısından sınıflandırılması gösterilmiştir.



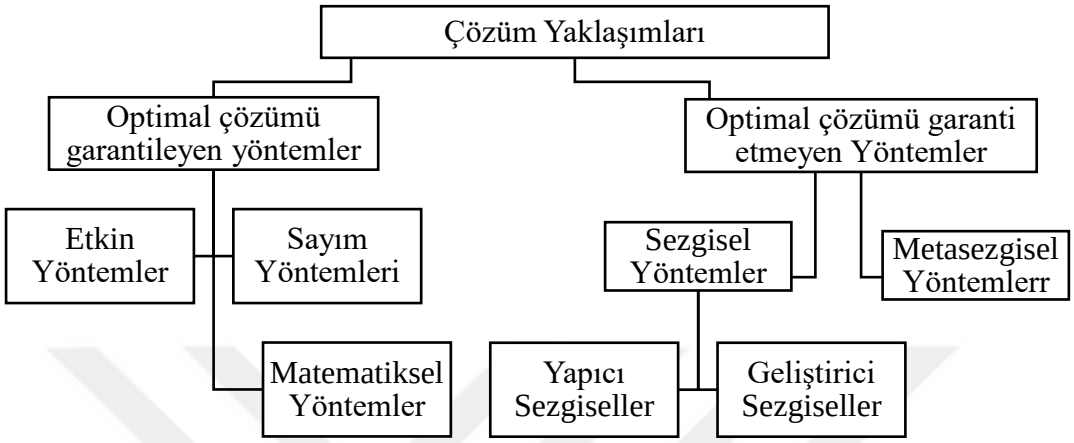
Şekil 2.12. Problemlerin karmaşıklığı (Özcan, 2013)

Çizelgeleme problemleri genellikle NP zor sınıfında yer alır ve bunlar için mevcut kolay çözüm yoktur. Çizelgeleme problemlerinin çözümünde kullanılacak yönteme karar vermek için öncelikle problemin NP-zor sınıfında olup olmadı belirlenmelidir. Bu amaçla daha önce problemin karmaşıklığının belirlenip belirlenmediği araştırılabilir. Bu amaçla Pinedo (2008) ve Aslan (2022) çalışmalarında NP-zor sınıfı problemlerin listelerini verilmişlerdir.

Problemleri P ve NP olarak ayırmanın amacı çözüm yönetimin kararlaştırılmasıdır. Büyük çapta bir NP-zor problemle karşılaşıldığında, eldeki teknikler ile optimum çözümlerin bulunamayabileceği bilinir ve kesin çözümü garanti etmeyen fakat daha az hesaplama ile yaklaşık çözümler üreten sezgisel yöntemler tercih edilebilir. Herhangi mantıklı bir çözümü kullanmanın, problemi eldeki en hızlı bilgisayarla kesin olarak çözmekten daha az zaman aldığı NP-zor örnekler mevcuttur. Bu nedenle sezgisel yöntemlere güvenmek genellikle pratikte istisnadan ziyade kuraldır (Baker ve Trietsch, 2009).

Çizelgeleme problemlerine çözümü için farklı yaklaşımlar içeren çeşitli yöntemler geliştirilmiştir. Bu yöntemler optimum çözümü garantileyen ve optimum çözümü garanti etmeyen fakat yaklaşık çözüm bulmaya çalışan yöntemler olarak iki grupta toplanabilir. Birinci grupta yer alan yöntemler klasik optimizasyon teknikleri olarak da bilinir. İkinci grupta ise probleme özgü geliştirilen sezgiseller ve yapay zekâ tabanlı metasezgiseller

bulunmaktadır. Şekil2.13'de çizelgeleme problemlerinde kullanılan çözüm yaklaşımları şematik olarak gösterilmiştir.



Şekil 2.13. Çözüm yaklaşımları

2.4.1. Optimal Çözümü Garantileyen Yöntemler

Tamsayılı doğrusal programlama, dinamik programlama ve hedef programlama, dal ve sınır algoritması vb. klasik optimizasyon teknikleri ile çizelgeleme problemlerini çözmek ve kesin sonuçlar elde etmek mümkündür. Fakat bu yöntemler büyük boyutlu problemlerde kabul edilebilir sürelerde çözüme ulaşmakta yetersiz kalmaktadır. Bu grupta yer olan yöntemler etkin yöntemler ve sayım yöntemleri olarak iki başlıkta incelenebilir.

Etkin yöntemler: Polinom zamanda optimum sonucu bulabilen çözüm süreçleridir. İlk ve en çok bilinen örneği iki makineli akış tipi çizelgeleme problemleri için geliştirilmiş olan Johnson yöntemidir (Johnson, 1954). Aker (1956) tarafından geliştirilen yöntem, iki iş ve m makine veya iki makine ve n işlik çizelgeleme problemleri için kullanılmaktadır. Szwarc (1960) tarafından ise iki iş ve m makine problemi genişletilerek üç iş ve on makineli bir problemin çözümü için uyarlanmıştır. Hefetz ve Adiri (1982) tarafından ise iki makineli ve birim işlem zamanlı atölye tipi çizelgeleme problemi için etkin bir yöntem önerilmiştir (Aslan, 2022).

Sayım yöntemleri: Temel sayım stratejisi olan dal-sınır algoritması; en iyi çözümden daha kötü olan çözümlerden veya şimdiye kadar elde edilen çözümlerden tahmin edilebilen optimum maliyetin üst sınırına neden olacak dalları keserek bir kısmi karar ağacı oluşturmaktadır (Palamutçuoğlu, 2022). Bu dinamik karar ağacı ile olası çizelgelerden oluşan çözüm uzayı araştırılır. Bu yöntem, ağacın büyük kısımlarını araştırmadan çıkarılmayı amaçlayan prosedürler ve kurallardan oluşmaktadır. Dal, büyük bir problemin daha yönetilebilir alt problemlere bölünmesidir. Sınır ise alt problemlerin optimum sonuçları için sınırların hesaplanması işlemleridir. Dal sınır yöntemiyle küçük boyutlu problemler uygun bir sürede çözülebilir fakat büyük boyutlu problemlerde yöntem yetersiz kalmaktadır. Bu durumun esas nedeni, erken aşamalarda olası tüm çizelgelerin büyük bir kısmını eleyecek kadar etkili kuralların henüz oluşturulamamış olmasıdır (Aslan, 2022).

Matematiksel yöntemler: Dinamik programlama, tamsayılı doğrusal programlama, ve hedef programlama gibi yöneylem araştırması teknikleri bu başlık altında incelenebilir. Çizelgeleme problemlerinin bu tür klasik matematiksel yöntemler ile çözümünde çoğunlukla çok miktarda değişken ve kısıda ihtiyaç duyulur. Bu durum yöntemlerin etkinliğini problem boyutu artıkça düşürmektedir.

2.4.2. Optimum Çözümü Garanti Etmeyen Yöntemler

Önceki başlıklarda açıklandığı gibi çizelgeleme problemleri genellikle NP zor sınıfında yer alır ve çoğunlukla kesin sonuç veren teknikler bulunmaz. Bu durumda optimum sonucu garanti etmeyen fakat makul bir süre içerisinde optimum sonuca yakın sonuçlar üretebilen sezgisel yöntemlere başvurulur.

Sezgisel yöntemler, herhangi bir amacı gerçekleştirmek veya hedefe varmak için çeşitli alternatif hareketlerden etkili olanlara karar vermek amacıyla tanımlanan kriter veya bilgisayar metotlarıdır (Karaboğa, 2002). Sezgisel yöntemler, kabul edilebilir hesaplama maliyetleri ile uygunluğu veya optimumu garanti etmeksizin iyi (optimuma yakın) çözümü arayan algoritmik süreçlerdir. Bu yöntemlerde çoğu zaman optimale ne kadar yakın olunduğu dahi belirlenemez (Cura, 2006).

Kesin optimum çözüm teknikleri yerine sezgisel yöntemlerin kullanılmasının çok sayıda sebebi vardır. Birincisi, model oluşturma aşamasında genellikle birçok varsayımda bulunulur ve oluşturulan model bu varsayımlar nedeniyle problemi tam olarak temsil etmez. Bu nedenle modelin çözümü ile elde edilen sonuç gerçek problemin en iyi çözümünden farklı olabilir. İkincisi, uygulamada karşılaşılan çizelgeleme problemlerinin çoğu NP-zor sınıfında olduğundan bu problemler gelişmiş iş istasyonlarında çözülse bile kesin çözüme ulaşmak için çok uzun zamana ihtiyaç duyulabilir. Üçüncü olarak, üretim ortamlarında çok sayıda belirsizlik ve rassallık bulunmaktadır. Bu nedenle birkaç saat gibi kısa bir süre içerisinde makinelerin arızalanması, yeni işlerin gelmesi, işlerin teslim tarihlerinin değişmesi vb. durumların meydana gelebileceği bir ortamda, çok uzun hesaplama zamanı harcayarak optimum bir çözüme ulaşmak kabul edilebilir bir yaklaşım değildir (Pinedo, 2008; Aslan, 2022).

Yaklaşık algoritmalar olarak da bilinen optimum çözümü garanti etmeyen yöntemler sezgisel yöntemler ve metasezgisel yöntemler olarak iki grupta incelenebilir. İki grup arasındaki belirleyici fark; sezgiseller genellikle tek çözüm üreten, daha basit kurallardan oluşması, metasezgisellerin ise daha karmaşık ve zeki yapılarıyla başlangıç çözümü her adımda geliştirerek daha iyi çözümler bulmaya çalışmalarıdır. Zeki yapılara sahip olmaları nedeniyle metasezgisel yöntemler, yapay zeka teknikleri olarak da bilinmektedir (Aslan, 2022).

Eski Yunanca'da "problem çözme sanatı" veya "problemleri çözmek için yeni yöntemler geliştirme" anlamlarına sahip "heuriskein" kelimesinden türetilen sezgisel terimi günümüzde yöneylem araştırması ve optimizasyon alanlarında yaygın olarak kullanılmaktadır. Sezgisel terimi, tam çözümü bulunması zor hatta imkansız olan problemler için makul çözümler bulmada kullanılan yöntemleri kapsamaktadır. Sezgisel yöntemler, optimum ya da optimumuma yakın bir çözüme götüren fakat bunu garanti etmeyen, hızlı ve pratik stratejilerdir (Kundakçı, 2013). Sezgisel yöntemlerin zayıf tarafı probleme özgü olarak geliştirilmeleri ve farklı problem türleri için uygulanamamalarıdır (Özcan, 2013).

Genel olarak sezgisel yöntemler iki grupta sınıflandırılır: Çözüm kurucu/yapıcı sezgiseller (constructive algorithm) ve çözüm geliştirici sezgiseller (improvement

algorithm) (Dağ, 2012; Ruiz ve Maroto, 2005). Literatürde sezgisellerin farklı şekilde sınıflandırıldığı çalışmalarda mevcuttur.

Temel önceliklendirme kuralları en basit sezgisel yöntemlerdir. Bu yapısal tekniklerde işler; süreleri, geliş zamanları vb. ölçütlere göre sıralanır ve öncelikleri belirlenir. Daha sonra işler belirlenen öncelik sırasına göre seçilerek makinelere gönderilir. Bugüne kadar çok sayıda önceliklendirme kuralı üzerinde çalışılmıştır fakat herhangi bir kural öne çıkmamaktadır. Bunun temel nedeni çok problem bağımlı olmalarıdır. Bununla birlikte en uzun işlem süreli işin en erken işlenmesi, en kısa işlem süreli işin en erken işlenmesi, en kısa hazırlık süreli işin en erken işlenmesi, teslim zamanı en erken olan işin en önce işlenmesi, ilk gelen işin en erken işlenmesi yaygın önceliklendirme kuralları arasında yer almaktadır. Temel önceliklendirme başka bir ifadeyle dağıtma kurallarının uygulanması çok kolaydır ve çözüm için harcanan süreler kısadır (Aslan, 2022).

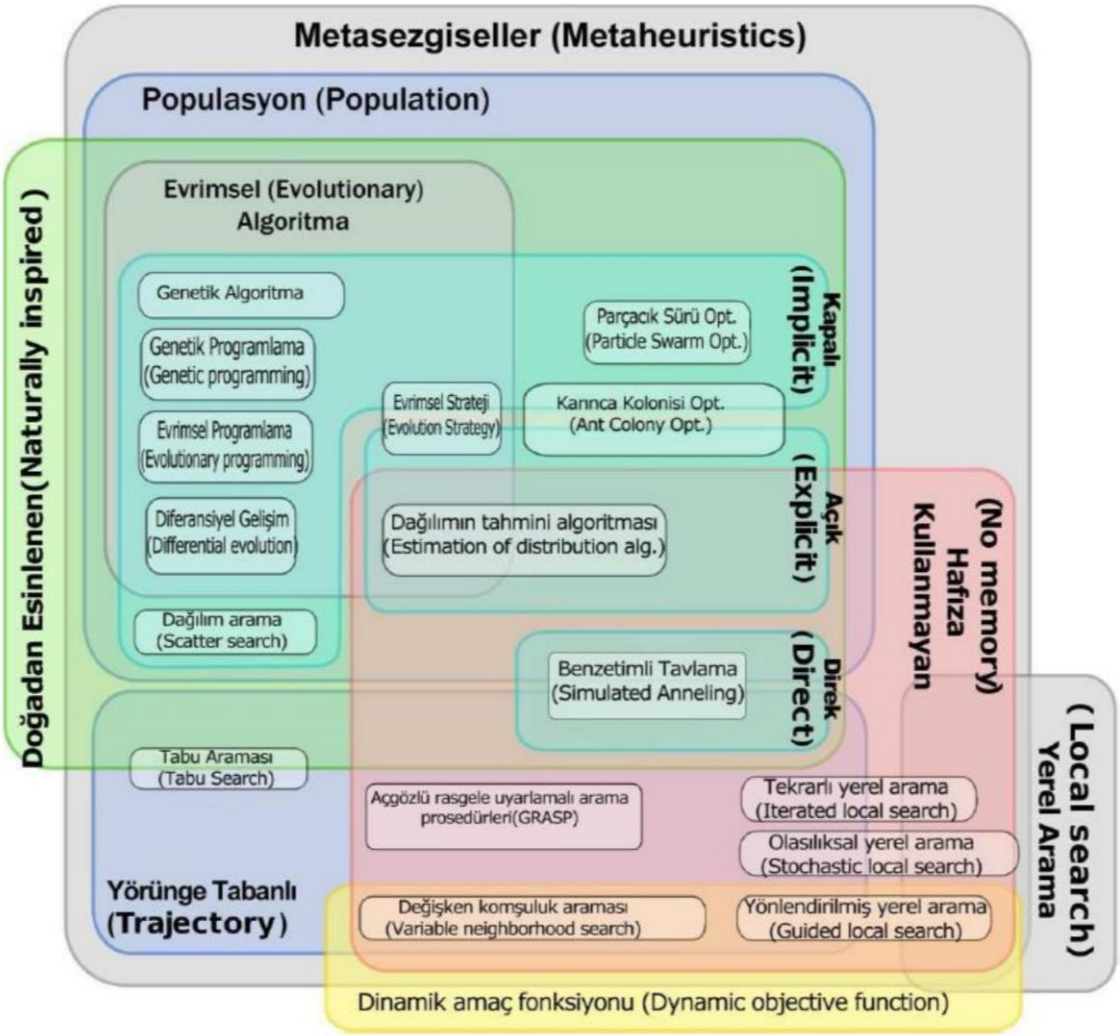
Palmer tarafından geliştirilen Eğim Dizisi Yöntemi (Palmer, 1965), Campbell, Dudek Smith (CDS) Algoritması (Campbell vd., 1970); Gupta tarafından önerilen sezgisel algoritma (Gupta, 1971), Dannenbring Yöntemi (Dannenbring, 1977) ve Nawaz, Enscore ve Ham (NEH) Yöntemi (Nawaz ve diğ., 1983) çözüm kurucu sezgisel arasında yer almaktadır.

Çözüm geliştirici sezgisel algoritmalar, eklenen bir takım yeni süreç ve kurallarla mevcut sezgisellerle elde edilen çözümleri iyileştirmeye çalışırlar. Ho ve Chang (HC) Yöntemi (Ho ve Chang, 1991), HAMC (Ravindran ve diğ., 2005) ve NEH-D (Dong ve diğ., 2008) geliştirici sezgisel örnekleridir.

Günümüzde temel sezgisel metotların birleşerek etkili kullanımı sonucunda metasezgisel algoritmalar geliştirilmiştir. Metasezgisel kelimesi bulmak anlamına gelen "sezgisel" ile daha üst bir seviye anlamına gelen "meta" adındaki iki Yunanca kelimeden gelmektedir. Literatürde yer alan metasezgisel terimine ilişkin çeşitli tanımların ortak noktaları aşağıda verilmiştir (Tosun Ö. , 2012; Blum ve Roli, 2003):

- Metasezgiseller araştırma sürecine rehberlik eden stratejilerdir,
- Amaç optimal/ optimuma yakın çözümler bulabilmek için araştırma uzayını etkili bir biçimde keşfetmektir,

- Metasezgisel teknikler basit yerel araştırma tekniklerinden karmaşık öğrenme süreçlerine kadar çeşitlilik gösterir,
- Metasezgiseller yaklaşık sonuçlar verir ve genellikle deterministik değillerdir,
- Araştırma uzayındaki kimi noktalarda (yerel minimum) tuzağa düşmeyi önleyecek mekanizmalar içerebilirler,
- Problem tabanlı değildirler,
- Araştırmayı kolaylaştırmak için bilgi tabanlı olabilirler (TA'da tabu listesi kullanımı vb.).



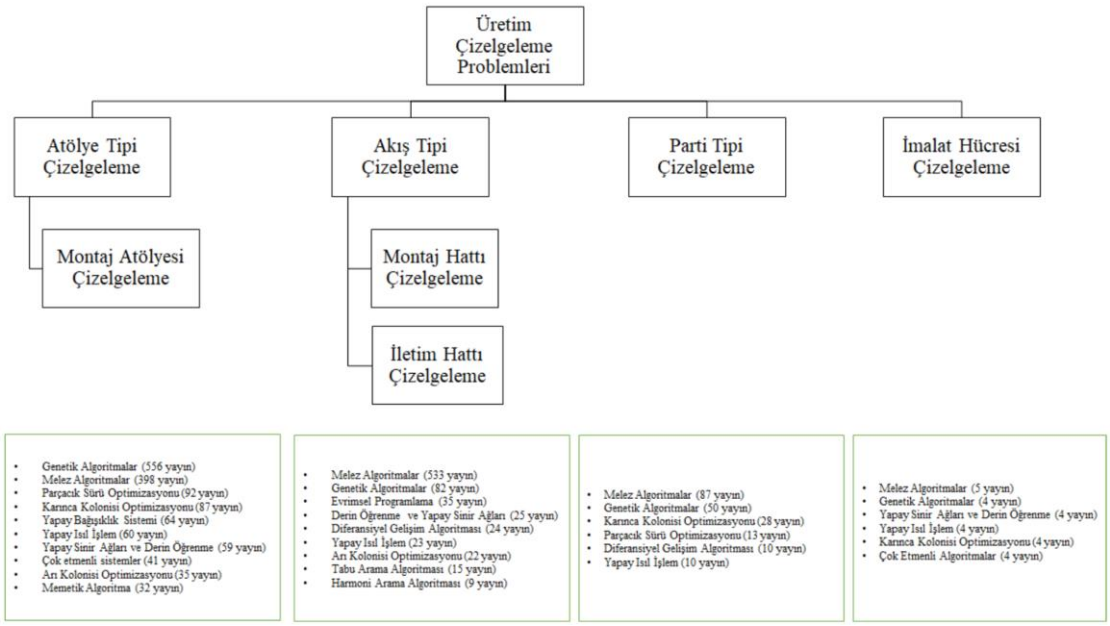
Şekil 2.14. Metasezgisel Algoritmaların Sınıflandırılması (Çelik ve diğ., 2019)

Metasezgisel algoritmalar, yüksek seviyeli çalışma ortamında, verimli arama işlemleri kullanarak çözüm uzayındaki optimum çözüme daha hızlı şekilde ulaşmaktadır.

Metasezgisel algoritmalarının en önemli avantajlarından biri de, yerel optimum noktalara takılmadan global sonuca ulaşabilme yeteneğidir (Çelik ve diğ., 2019).

Literatürde Genetik Algoritma, Diferansiyel Gelişim Algoritması, Karınca Kolonisi Optimizasyonu, Parçacık Sürü Optimizasyonu, Yapay Arı Kolonisi Algoritması (YAKA), Tavlama Benzetimi başta olmak üzere birçok kabul görmüş metasezgisel bulunmaktadır ve bunlara sürekli yenileri eklenmektedir. Metasezgisel yöntemlerin sınıflandırılması faklı şekillerde yapılmaktadır. Doğadan esinlenme, popülasyon tabanlı olma, evrimsel olma en çok kullanılan sınıflandırma ölçütlerdir. Şekil 2.14'de Çelik ve diğerleri (2019) tarafından yapılan sınıflandırma şematik olarak verilmiştir.

Palamutçuoğlu (2022); metasezgisel yöntemlerinin çizelgeleme problemlerinde uygulanmasına ilişkin SCOPUS ve Web of Science indekslerinde taranan ve 1996-2021 yıllarında yayınlanan dergilerde yaptığı literatür taramasında, bu yöntemlerin kullanıldığı 854 akış tipi, 1.410 atölye tipi, 36 parti tipi, 28 üretim hücresi tipi, 30 montaj atölyesi, 34 montaj hattı tipi ve 4 iletim hattı çizelgeleme yayınını incelemiştir. İnceleme sonucunda hangi çizelgeleme problemlerinde hangi algoritmaların yaygın olduğu belirlenerek Şekil 2.15'te gösterilmiştir.

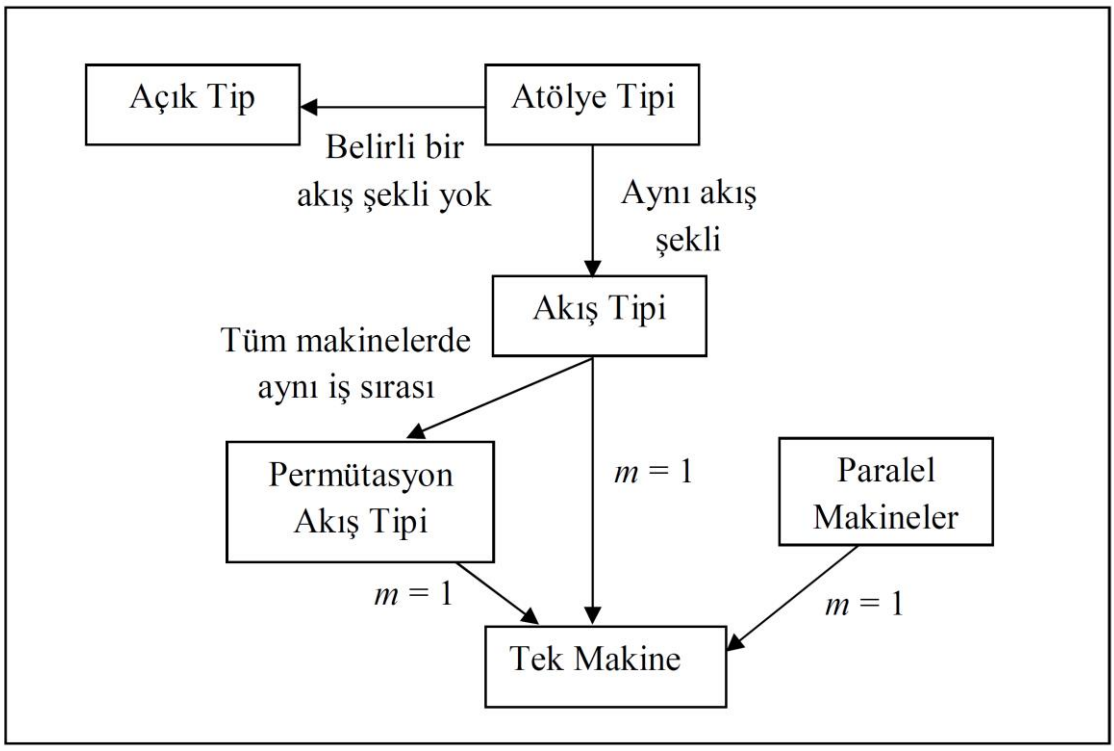


Şekil 2.15. Çizelgeleme problemlerinde yaygın olarak kullanılan Metasezgiseller (Palamutçuoğlu, 2022)

3. PERMÜTASYON AKIŞ TİPİ ÇİZELGELEME PROBLEMİ

Akış tipi üretim, teknik kısıtlar ve seri üretim hatlarının artması nedeniyle birçok endüstri dalında yaygın şekilde uygulanmaktadır. Bu sebeple akış tipi çizelgeleme problemi, çok sayıda araştırmaya konu olmuştur. ATÇP'nin bir alt sınıfı olan permütasyon akış tipi çizelgeleme probleminde her bir işin işlem sıraları bütün makinelerde birebir aynıdır (Yağmahan ve Yenisey, 2006; İşler ve diğ., 2009).

Akış tipi yerleşimde, iş sıralarının her makinede farklı olduğu çizelgeler ve iş sıralarının her makinede aynı olduğu permütasyon çizelgeler olmak üzere iki farklı çizelgeleme türü bulunmaktadır. Bir iş birinci makineden başlayarak bütün makinelerde tam olarak aynı sıra ile işlem görüyorsa permütasyon akış tipi çizelgeleme problemi olarak adlandırılır (Çiçekli ve Bozkurt, 2016). Şekil 3.1'de yerleşim tipine göre sınıflandırmada PATÇP'nin yeri şematik olarak gösterilmiştir.



Şekil 3.1. Yerleşim tipine göre çizelgeleme problemleri ve PATÇP (Kundakçı, 2013)

ATÇP'de birden fazla makine bulunur ve bütün işler makinelerden aynı sıra ile geçer. PATÇP'de ise ilk başta iş sırası sabitlenir ve tüm makinelerde aynı iş sırası kullanılır (Göçmen ve diğ., 2010). Genel ATÇP'de özüm aralığı $(n!)^m$ adet çizelgeden oluşurken PATÇP'de makineler için aynı permütasyon kullanıldığından problem basitleşir ve olası çizelge sayısı (n!) adete düşer (Kurnaz ve Kart, 2010).

Akış tipi çizelgeleme problemi, m tane makinede aynı sırayla işlenecek olan n işin en iyi şekilde sıralaması ile ilgilenir. Bu problemler için genel kabuller ve kısıtlar şu şekildedir (Kurnaz ve Kart, 2010; Bacha ve diğ., 2022):

- ✓ Tüm işler bütün makinelerde aynı sıra ile işlenir.
- ✓ Tüm işler birbirinden bağımsız ve başlangıçta işlenmeye hazırdır.
- ✓ Makineler her zaman uygundur.
- ✓ Her bir makine yalnızca bir işi yapabilir ve her bir iş yalnızca bir makinede işlenebilir.
- ✓ Makinede işlenen işin yarıda kesilmesi mümkün değildir.
- ✓ Eğer bir işin gireceği makinede başka bir iş varsa, diğer iş, makinenin boşalmasını kuyrukta beklemek zorundadır.

3.1. Permütasyon Akış Tipi Çizelgeleme Problemlerinin Notasyonu ve Matematiksel Modeli

Önceki bölümde açıklanan üç parametreli gösterimde çizelgeleme problemleri α/ β/ γ şeklinde üç parametre ile ifade edilmektedir. Bu gösterimde α parametresi iki makine problemleri için F2, m makineli problemler için ise Fm' dir. PATÇP'de β alanına "prmu" yazılır ve bu problemin permütasyon kısıtlı olduğunu ifade eder. Performans ölçütünü ifade eden γ parametresinin alabileceği değerlere göre PATÇP'nin notasyonu değişmektedir. Sık karşılaşılan PATÇP notasyonları şunlardır:

- F2/prmu/C_{max}: İki makineli maksimum tamamlanma zamanı ölçütlü PATÇP,
- Fm/prmu/C_{max}: m makineli maksimum tamamlanma zamanı ölçütlü PATÇP,
- Fm/prmu/ ΣF: m makineli toplam akış zamanı ölçütlü PATÇP,
- Fm/prmu/ ΣF_{max}: m makineli maksimum akış zamanı ölçütlü PATÇP,
- Fm/prmu/ ΣI: m makineli toplam boş makine zamanı ölçütlü PATÇP,
- Fm/prmu/ ΣT: m makineli toplam geç bitirme zamanı ölçütlü PATÇP,

- Fm/prmu/ C_{max}, ΣF: m makineli maksimum tamamlanma zamanı ve toplam akış zamanı ölçütlü (çok amaçlı) PATÇP,
- Fm/prmu/ C_{max}, ΣT: m makineli maksimum tamamlanma zamanı ve toplam geç bitirme zamanı ölçütlü (çok amaçlı) PATÇP,
- Fm/prmu/ C_{max}, ΣF, ΣT: m makineli maksimum tamamlanma zamanı, toplam akış ve toplam geç bitirme zamanı ölçütlü (çok amaçlı) PATÇP,

Bu örnekleri artırmak mümkündür. Akış tipi çizelgeleme problemlerinde performans ölçütü olarak genellikle maksimum tamamlanma zamanı (makespan) veya toplam akış zamanı kullanılmaktadır (Pan ve diğ., 2008a; Pan ve diğ., 2008b).

Bekleme süresinin olmadığı ve hazırlık sürelerinin işlem sürelerine dahil edildiği kabul edildiğinde, p_{ij} j. işin i makinesindeki işlem süresi olmak üzere verilen $\pi=\{\pi_1, \pi_2,\ldots, \pi_n\}$ iş sırasında tamamlanma süreleri Eşitlik (3.1-3.4) ile şu şekilde hesaplanmaktadır (Tasgetiren ve diğ., 2007):

$$C(1,\pi_1) = P_{1.\pi_1} \tag{3.1}$$

$$C(1,\pi_j) = C(1,\pi_{j-1}) + P_{1,\pi_j}, \qquad j = 2,\ldots,n, \tag{3.2}$$

$$C(i,\pi_1) = C(i-1,\pi_1) + P_{i,\pi_1}, \qquad i = 2,\ldots,m, \tag{3.3}$$

$$C(i,\pi_j) = Maks\{C(i,\pi_{j-1}) | C(i-1,\pi_j)\} + P_{1,\pi_j}, \quad i = 2,\ldots,m; \quad j = 2,\ldots,n. \tag{3.4}$$

Burada n iş sayısı, m makine sayısı ve π iş sırasını göstermekte olup işler bütün makinelerde aynı sıra ile yapılmaktadır. Bu durumda maksimum tamamlanma zamanı Eşitlik (3.5) ile hesaplanır ve son sıradaki iş, son makineden çıkıncaya kadar geçen süreyi ifade eder.

$$C_{maks}(\pi) = C(m,\pi_n) \tag{3.5}$$

Akış süresi (flow time); İşin işlenmeye hazır olduğu andan tamamlanmasına kadar geçen zamandır. Beklemenin olmadığı ve hazırlık sürelerinin işlem sürelerine dahil olduğu kabul edildiğinde, bir işin akış süresi o işin tamamlanma süresine eşit olmaktadır. Bu

durumda toplam akış süresi, işlerin akış sürelerinin yani işlerin tamamlanma sürelerinin toplamıdır ve Eşitlik (3.6) ile hesaplanır (Tasgetiren ve diğ., 2011):

$$\sum F = \sum_{=1}^{n} C(m, \pi_n) \tag{3.6}$$

Genel ATÇP'de optimum sonuca ulaşmak için $(n!)^m$ adet çözümün değerlendirilmesi gerekirken Conway'in önerdiği iki teoremle bazı çözümler elenebilir (Ying, 2008). PATÇP' de ise tüm makinelerde iş sırası aynı olduğu için olası çözüm sayısı (n!) adettir. Tek veya iki makineli PATÇP'ler kolay sınıfına girmektedir. Makine ve iş sayısının büyümesi durumunda, olası çizelgelerin sayısı da büyüyeceği için en iyi çözümün tespiti için harcanacak zaman aşırı artar. Bu problemlerin çözümü için bir polinomsal zaman algoritması olmadığından, PATÇP'ler NP-zor olarak nitelendirilirler (Dağ, 2012).

Problemlerin büyüklüklerine ve zorluk derecelerine göre geliştirilmiş çeşitli yöntemler bulunmaktadır. Küçük ve polinomsal sürede çözülebilen problemler için tam sayılı programlama, dal-sınır algoritması, kesme düzlemi algoritmaları ile kesin sonuçlar bulunabilirken, çoğu zaman üstel hesaplama zamanı gerektiren büyük çaplı problemler için optimale yakın sonuçlar veren sezgisel ve metasezgisel yöntemler geliştirilmiştir (Palamutçuoğlu, 2022).

3.2. PATÇP'lerde Kullanılan Sezgisel Yöntemler

İki makineli permütasyon akış tipi çizelgeleme problemleri Johnson kuralı ile etkin bir şekilde çözülebilir. Üç makineli PATÇP'ler de bazı koşulları sağlaması durumunda yine Johnson kuralı etkin bir şekilde çözülebilirken m>3 makineli PATÇP'ler için etkin bir yöntem yoktur. Tamsayılı doğrusal programlama, dal-sınır algoritması gibi klasik optimizasyon yöntemleriyle küçük ölçekli problemler çözülebilmektedir. Fakat büyük boyutlu problemlerde bu yöntemler yetersiz kaldığı için sezgisel yöntemlere başvurulmaktadır. Tablo 3.1'de Permütasyon akış tipi çizelgeleme problemleri için geliştirilen sezgisel örnekleri verilmiştir.

Tablo 3.1. PATÇP İçin Geliştirilen Sezgisel Teknikler

Çalışma	Başlık/Konu	Tür	Notlar
Palmer (1965)	Minimum Toplam Sürede Çok Aşamalı Bir İş Sıralama—Optimuma Yakın Bir Hızlı Yöntem	Yapıcı	Eğim dizisi yöntemi
Campbell ve diğ. (1970)	n İş, m Makine Sıralama Problemi İçin Sezgisel Bir Algoritma	Yapıcı	CDS
Gupta (1971)	Akış Tipi Çizelgeleme Problemi İçin İşlevsel Bir Sezgisel Algoritma	Yapıcı	Eğim dizisine dayalı
Dannenbring (1977)	Akış Tipi Çizelgeleme Sezgisellerinin Değerlendirilmesi	Yapıcı	Johnson ve Palmer'e dayalı
Nawaz ve diğ. (1983)	m-makine, n-iş Akış Tipi Çizelgeleme Problemi için Sezgisel Bir Algoritma	Yapıcı	NEH
Widmer ve Hertz (1989)	Akış Tipi Çizelgeleme Problemi İçin Yeni Bir Sezgisel Yöntem	Yapıcı	Widmer ve Hertz Yöntemi
Ho ve Chang (1991)	n İş, m Makine Akış Tipi Çizelgeleme Problemi İçin Yeni Bir Sezgisel Algoritma	Geliştirici	HC) Yöntemi
Rajendran (1995)	Çok Amaçlı Akış Tipi Çizelgeleme Problemi için Sezgiseller	Yapıcı	
Agarwal ve diğ. (2006)	Akış Tipi Çizelgeleme Problemi için Geliştirici Sezgisel: Bir Uyarlamalı Öğrenme Yaklaşımı	Geliştirici	Ağırlıklandırılmış işlem süreleri
Göçmen ve diğ. (2010)	Permütasyon Akış Tipi Çizelgele Problemleri için CDS ve NEH Algoritmalarının Kıyaslanması ve Yeni bir Algoritma Önerisi	Geliştirici	
Kurnaz ve Kart (2010)	İş Akış Çizelgeleme Problemi Üzerinde NEH, FRB3 ve FRB4 Sezgisellerinin Karşılaştırılması	Yapıcı	

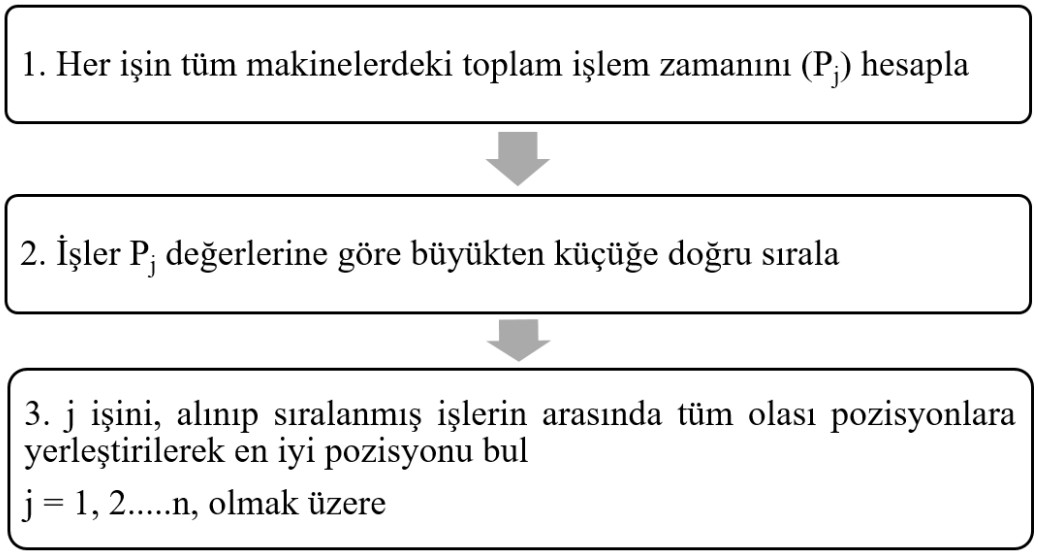
Sezgisel yöntemler tek başına kullanılabildikleri gibi metasezgisel yöntemlerde başlangıç çözümlerin oluşturulmasında da kullanılmaktadır. NEH algoritması bu amaçla yaygın

olarak kullanılan ve literatürde kabul görmüş yöntemlerdendir. Bu çalışmada da geliştirilen metasezgisel yöntemde başlangıç çözüm üretmek için NEH algoritması kullanıldığı için bu bölümde bu algoritma hakkında bilgi verilmiştir.

3.2.1. NEH Algoritması

Nawaz, Enscore ve Ham tarafından 1983 yılında önerilen NEH sezgiseli akış tipi çizelgeleme probleminin çözümünde yüksek performansa sahip ve en çok bilenen yöntemdir. Bu yöntemin performansının güncel ve daha karmaşık birçok sezgiselden daha iyi olduğunu gösteren çalışmalar mevcuttur. Son yıllarda geliştirilen yüksek performanslı sezgisel yöntemlerin birçoğu yine NEH tabanlıdır (Kurnaz ve Kart, 2010).

NEH yöntemi; tüm makineler üzerinde daha fazla işlem süresine sahip işlere daha az işlem süresine sahip işlerden daha yüksek öncelik verilmesi varsayımına dayanır. Buna göre; n iş arasından en yüksek işlem süresine sahip iki iş seçilir. Bu iki iş için en iyi sıralama belirlenir. Bu iki işin birbirine göre pozisyonları sabit tutulur. Daha sonra, üçüncü en yüksek işlem süresine sahip iş seçilir. Bu iş bir önceki adımda bulunan kısmi sıralama başına, ortasına ve sonuna yerleştirilerek oluşturulan üç kısmi sıralama değerlendirilir ve en iyi kısmi sıralama seçilir. Tüm işler çizelgelenene kadar süreç bu şekilde devam eder (Göçmen ve diğ., 2010). NEH yönteminin işleyişi üç adımda Şekil 3.2.'de verilmiştir.



Şekil 3.2. NEH yönteminin işleyiş adımları (Kurnaz ve Kart, 2010)

3.3. PATÇP'lerde Kullanılan Metasezgisel Yöntemler

Permütasyon akış tipi problemlerinin çoğunda çözüm uzayı bütün çözümlerin değerlendirilemeyeceği kadar büyüktür. Bu nedenle makul bir sürede çözümler değerlendirilerek kabul edilebilir bir çözüm bulunması gerekmektedir. Bu amaçla sezgisel ve metasezgisel yöntemler kullanılmaktadır. Bu yöntemlerde çözüm uzayının tamamı yerine bazı çözümler değerlendirilir. Hangi çözümlerin neye göre ve nasıl seçilerek değerlendirileceği kullanılan sezgisele göre değişmektedir. Maalesef seçilen çözümler arasında optimum çözümün de yer alması garanti edilemez. Bu nedenle sezgisel yöntemlerle elde edilen çözümler optimum değil, iyi çözümler olarak algılanmalıdır (Cura, 2006). Birçok araştırmacı meta sezgiselleri, sezgisel yöntemlerin probleme özgü olmayan ve ileri düzey bir sınıfı olarak değerlendirmektedir.

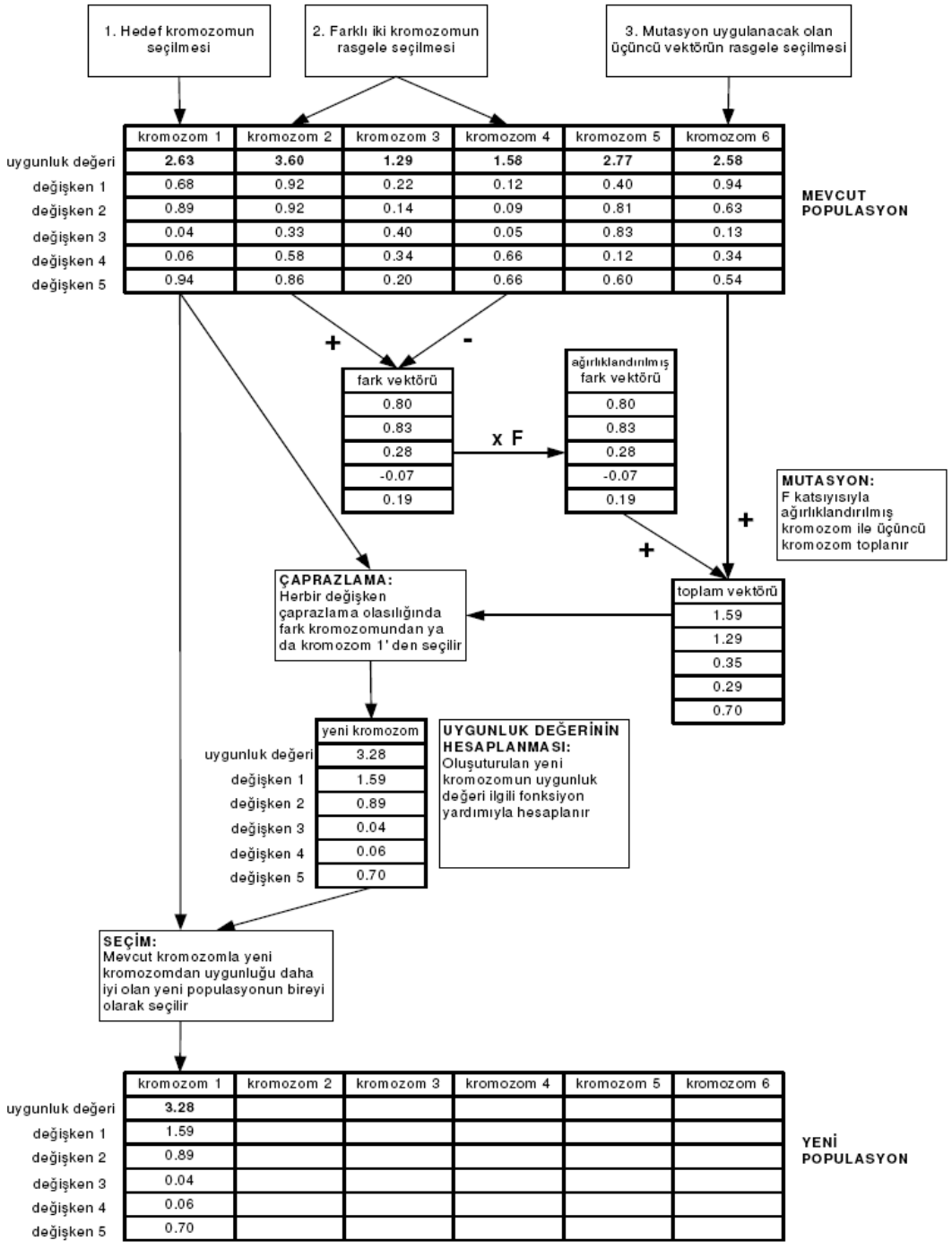
PATÇP'lerin çözümünde çok sayıda metasezgisel kullanılmıştır ve bunların sayısı her geçen gün artmaktadır. Çalışmanın bu bölümünde öncelikle yaygın olarak kullanılan metasezgisellerden biri olan ve bu çalışmada önerilen metotların performanslarını karşılaştırmak için kodlanan Diferansiyel Gelişim Algoritması hakkında kısa bilgi verilecektir. Daha sonra PATÇP'lerin çözümünde metasezgisel kullanılan çalışma örnekleri verilecektir.

3.3.1. Diferansiyel Gelişim Algoritması

Diferansiyel Gelişim Algoritması Price ve Storn tarafından 1996 yılında önerilmiş tekniği sağlam ve kullanımı kolay bir metasezgisel yöntemdir. Sürekli fonksiyonların optimizasyonunda yeni bir yaklaşım sunan DGA'nın diğer optimizasyon metotlarından daha hızlı ve daha kesin sonuçlara ulaştığı belirtilmiştir (Doğru ve Eren, 2020).

DGA, popülasyon tabanlıdır ve GA'ya benzer işleyiş ve operatörler sahihtir. Popülasyon tabanlı yapısı sayesinde aynı anda çözüm uzayında farklı noktalarda araştırma yapabilir. DGA, Genetik Algoritmadakilere benzer çaprazlama, mutasyon ve seçim operatörlerine sahip olmakla birlikte, GA'dan farkı her bir operatörün tüm popülasyona sırayla uygulanmasıdır. Her bir kromozom, tek tek ele alınarak rasgele seçilen diğer üç kromozomla birlikte kullanılarak yeni bir birey oluşturulur. Bu işlemler sırasında mutasyon ve çaprazlama operatörleri kullanılır. Yeni elde edilen kromozomun

uygunluğu, mevcut kromozomla karşılaştırılarak daha iyi olan birey bir sonraki nesile aktarılır. Bu şekilde, seçim operatörü de kullanılmış olur (Keskintürk, 2006). Diferansiyel Gelişim Algoritmasında yeni vektörlerin oluşturmasın için yapılan işlemler Şekil 3.3'te gösterilmiştir.



Şekil 3.3. DGA' da yeni vektörün oluşturulması işlemi (Keskintürk, 2006)

Öncelikle başlangıç popülasyonundan seçilen en iyi kromozomlara çaprazlama ve mutasyon operatörleri uygulanarak yeni bir birey elde edilir. Bunlar arasındaki en iyi birey seçim operatörü ile sonraki popülasyona aktarılır. Bu süreçte popülasyondaki tüm bireylere durdurma kriteri sağlanıncaya kadar söz konusu operatörler sıyrayla uygulanır (Dokur ve diğ., 2019).

İhtiyaç duyulan kod sayısının az olması DGA'ya diğer metasezgisellere karşı önemli bir üstünlük sağlamaktadır. DGA için yaklaşık 20 satırlık kod yeterli olmaktayken bazı metasezgiseller için binlerle ifade edilen satıra ihtiyaç duyulmaktadır (Keskintürk, 2006).

3.3.2. PATÇP'nin Çözümünde Metasezgisel Kullanılan Çalışmalar

Bu çalışma süresince permütasyon akış tipi çizelgeleme probleminin çözümünde metasezgisellerden faydalanılan yüzün üzerinde çalışma incelenmiştir. Başlangıçta Tavlama Benzetimi, Tabu Arama ve Genetik Algoritma genellikle tek başlarına kullanılırken zamanla bunlara diğer metasezgisel yöntemler eklenmiştir. İlk çalışmalarda yöntemler tek başlarına kullanılırken zamanla performanslarını geliştirilmesi amacıyla başlangıç çözümün diğer sezgisellerle oluşturulduğu, yerel arama tekniklerinin eklendiği, iki veya daha fazla metasezgisel yöntemin birlikte kullanıldığı çalışmalar yaygınlaşmıştır.

Bu alanda yapılan çalışmalarda göze çarpan başka bir değişimde performans ölçütlerinde yaşanmışıdır. Başlangıçta maksimum tamamlanma zamanı, toplam akış zamanı vb. tek amaçlı çalışmalar bulunurken özellikle son on beş yılda iki veya daha çok ölçütü birlikte iyileştirmeyi amaçlayan çok amaçlı çalışmaların sayısı artmıştır. Benzer şekilde performans ölçütlerindeki çeşitlilik de artmıştır. Örneğin son beş yılda yapılan çalışmalar arasında, enerji verimliğinin artırılmasını amaçlayan yeşil çizelgeleme kavramının önemli bir yere sahip olduğu görülmüştür. Tablo 3.2'de PATÇP'lerin çözümünde metasezgisel yöntemlerden faydalanılan örnek çalışmalar verilmiştir.

Tablo 3.2. PATÇP'nin çözümünde kullanımı yaygın olan metasezgiseller

Metasezgisel	Çalışma	Notlar
Genetik Algoritma	Reevest (1995)	İlk GA çalışması
	Iyer ve Saxena (2004)	Yerel arama
	Ruiz ve diğ. (2006)	İki yeni GA
	Tseng ve Lin, (2009)	Melez GA
	Li ve diğ. (2021)	Yeşil Çizelgeleme
Tabu Arama	Widmer ve Hertz 1989)	İki aşamalı-SPIRIT
	Taillard (1990)	NEH ile Başlangıç
	Daya ve Al-Fawzan (1998)	
	Grabowski ve Wodecki (2004)	Hızlı TA
	Eksioglu ve diğ. (2008)	Üç farklı değişim metodu
	Ali ve diğ. (2021)	Dağınık PATÇP
Tavlama Benzetimi	Osman ve Potts (1989)	İlk Metasezgisel
	Ishibuchi ve diğ. (1995)	Modifiyeli BT
	Nearchou A. (2004)	
Karınca Kolonisi Optimizasyonu	Stützle (1998)	İlk çalışma
	Rajendran ve Ziegler (2004)	Maksimum tamamlanma zamanı/toplam akış zamanı
	Ying ve Liao (2004)	
	Ahmadizar (2012)	Maksimum tamamlanma zamanı
	Yağmahan ve Yenisey 2006)	Parametre Eniyileme
Parçacık Sürü Optimizasyonu	Liu ve diğ. (2007)	Memetik Algortima bazlı PSO
	Pan ve diğ. (2008b)	Ayrık PSO
	Tasgetiren et al. (2007)	Maksimum tamamlanma zamanı/toplam akış zamanı
	Ramanan ve diğ. (2014)	Değişken komşuluk arama
	Amallynda (2019)	Çok amaçlı
Diferansiyel Gelişim Algoritması	Onwubolu ve Davendra (2006)	
	Pan ve diğ. (2008a)	Ayrık DGA
	Liu ve diğ. (2014)	
Yapay Arı Kolonisi Algoritması	Tasgetiren M. et al. (2011)	Ayrık YAKA
	Liu ve Liu (2013)	Melez YAKA
	Yu ve diğ. (2022)	Değişken komşuluk arama stratejisi
Dağınık Arama	Çiçekli ve Bozkurt (2016)	Permütasyon Akış Tipi Çizelgeleme Probleminin Dağınık Arama İle Optimizasyon
Ateşböceği	Kaya ve diğ. (2020)	Melez, PSO, başlangıç çözüm etkisi

4. EL BOMBASI PATLATMA METODU

El Bombası Patlatma Metodu (EBPM) el bombalarının patlama biçimlerinden esinlenerek geliştirilmiş evrimsel bir algoritmadır. EBPM'nin mantığı el bombalarının patlama şekline dayanmaktadır. El bombaları patladıkları bölgedeki nesneleri imha etmek için şarapnel parçaları fırlatır. Her bir şarapnelin neden olduğu kayıp hesaplanır. Şarapnel parçalarının neden olduğu kaybın yüksek olması o alanda değerli nesnelerin olduğunu gösterir. Kaybı artırmak için bir sonraki el bombası en çok kaybın oluştuğu noktaya atılır(Ahrari ve Atai, 2010).

El bombasına yakın olan nesneler daha çok zarar görür. Fakat her bir koordinat için patlama mesafesi, şarapnel parçalarının nesnelere zarar verebileceği uzaklık, olan Re'nin büyük seçilmesi ile uzaktaki nesnelerin de zarar görmesi mümkünüdür (Aguiar ve Junior, 2015). Bu süreçte nesnelerin tahribi ile oluşan zarar amaç fonksiyonun uygunluk değeri olarak düşünülür. Bu süreç sonunda, ilk tekrarlarda şarapnel parçaları bütün alanı araştıramasa dahi el bombasının atılacağı en iyi yer bulunacaktır. Bu süreçte nesnelerin tahribi ile oluşan zarar amaç fonksiyonun uygunluk değeri olarak düşünülür. El bombasının şu anki yeri X şu şekilde ifade edilir (Ahrari ve Atai, 2010):

$X = \{X_m\}, \qquad m = 1,2, \dots, n$

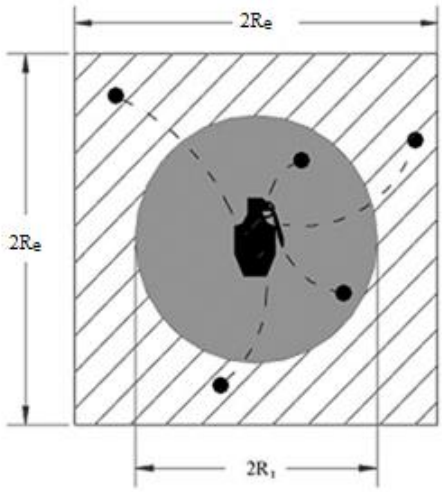
Burada n araştırılan uzayın boyutunu gösterir ve bağımsız değişken sayısına eşittir. Nq patlamada üretilen şarapnel sayısı ise imha edilen nesnelerin yerleri X'_j, Eşitlik (4.1) kullanılarak hesaplanır (Ahrari ve Atai, 2010).

$$X'_j = X_m + sign(r_m).|r_m|^p.\mathrm{Re} \tag{4.1}$$

Burada r_m [-1,1] aralığında düzgün dağılıma göre üretilmiş rassal bir sayı P ise sabit bir sayıdır. P sabiti araştırmanın yoğunluğunu belirlemektir. P'nin büyük değerleri, şarapnellerin el bombasının patladığı noktaya yakın olan bölgenin daha iyi taranmasını sağlarken P'nin küçük değerleri ise uzak noktaların taranmasına izin verir (Ahrari ve Atai, 2010). P; ajan bölgesi araştırma ihtimali olan P_{TS} değişkenine bağlı olarak, Eşitlik (4.2) kullanılarak hesaplanır. P_{TS} ilk iterasyonda 0.2 alınır daha sonra en büyük tahribatın meydana geldiği nokta baz alınarak yükseltilir veya azaltılır (Ahrari ve diğ., 2009).

$$P = \max\left(\frac{1}{n}, \frac{log(R_t/R_e)}{log(p_{TS})}\right) \quad (4.2)$$

EBPM'yi diğer metasezgisellerden ayıran tasarım özelliği ajan (bu algoritmada el bombası) bölgesi yarıçapıdır (R_t). Her hangi bir ajan diğer ajanlara belli bir mesafeden (R_t) daha fazla yaklaşamaz. Bu parametrenin büyük değerleri ile birkaç patlama gerçekleştirildikten sonra olurlu bölgenin düzgün şekilde tarandığından emin olunur. Bu parametrenin küçük değerlerinde ise el bombalarının birbirine yaklaşmasına izin verilerek belli bir bölgenin detaylı şekilde aranması sağlanır (Surender Reddy, 2016). Bu nedenle R_t değeri başlangıçta ($R_{t\text{-initial}}$) büyük seçilerek kademeli olarak düşürülür. R_{rd} parametresi, $R_{t\text{-initial}}$ değerinin son R_t değerine oranını ifade eder ve araştırmacı tarafından belirlenmelidir. Şekil 4.1'de patlama alanı ve ajan bölgesi çapı 2 boyutlu uzayda gösterilmiştir (Ahrari ve Atai, 2010).



Şekil 4.1. Patlama alanı ve ajan bölgesi çapı (Ahrari ve Atai, 2010)

Eşitlik (4.1)den açık bir şekilde anlaşılmaktadır ki; bir el bombası ile her bir kenarı $2R_e$ uzunluğuna sahip n boyutlu bir uzayda arama yapılmaktadır. Örneğin 3 boyutlu bir uzayda çalışılıyorsa arama yapılan bölge merkezinde el bombası bulunan, kenar uzunluğu $2R_e$ olan bir küp olacaktır (Ahrari ve Atai, 2010).

EBPM [-1,1] aralığında gerçek sayı kümesi ile çalıştığından metodun kullanılabilmesi için bağımsız değişkenlerin [-1,1] aralığında ölçeklendirmesi gerekmektedir. Ayrıca Eşitlik (4.1) kullanılarak üretilen şarapnel parçaları olurlu bölgenin dışına çıkabilir. Bu

durumda Eşitlik (4.3) kullanılarak şarapneller için olurlu bölge içinde yeni bir koordinat hesaplanır (Ahrari ve Atai, 2010).

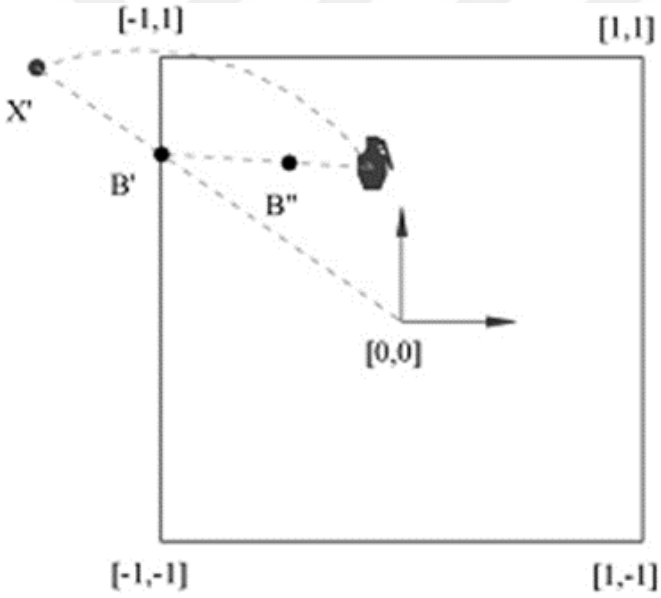
$$\text{Eğer } X'_j \notin [-1,1] \rightarrow \left(B'_j = \frac{X'_j}{\left| en\ büyük\ X'_j\ değeri \right|} \right)$$

$$\rightarrow B''_j = r'_j . \left(B'_j - X \right) \quad + X, \tag{4.3}$$

$j = 1,2, \ldots, Nq$ (şarapnel sayısı)

$0 < r'_j < 1$ (rassal sayı)

Burada X_j' olurlu bölge dışındaki çarpışma noktasını B'' ise olurlu bölgedeki yeni çarpışma noktasını göstermektedir. Bu işlem Şekil 4.2' de gösterilmiştir (Ahrari ve Atai, 2010).



Şekil 4.2. Olurlu bölge dışındaki noktaların olurlu bölgeye taşınması (Ahrari ve Atai, 2010)

Yukarıda metodun temel mantığı açıklanırken belirtiği üzere R_t değerinin başlangıçta ($R_{t\text{-}initial}$) büyük seçilerek kademeli olarak düşürülür. Böylelikle başlangıçta çözüm uzayı hızlıca taranırken ilerleyen iterasyonlarda optimum noktanın olabileceği bölgelerin detaylı taranması sağlanır. Aşağıda R_t değerinin kademeli olarak düşürülmesi için basit bir yöntem Eşitlik (4.4)de verilmiştir (Ahrari ve Atai, 2010):

$$R_t = \frac{R_{t-initial}}{(R_{rd})^{(iterasyon\ No./Toplam\ iterasyon\ sayısı)}} \quad (4.4)$$

R_{rd} değeri algoritma çalıştırılmadan önce seçilir. R_{rd} ilk iterasyondaki R_t değerinin son iterasyondaki R_t değerine oranını ifade eder. Ayrıca $2R_e$ değeri de aşağıda Eşitlik (4.5) göre güncellenir (Ahrari ve Atai, 2010):

$$R_e = (R_{e-initial})^{m}.(R_t)^{1-m}, \qquad 0 < m < 1 \quad (4.5)$$

Eşitlik (4.4) ve Eşitlik (4.5) ile Re değeri R_t değerine göre daha yavaş düşürülerek global arama yeteneğinin korunması sağlanır. Başlangıçta belirlenmesi gereken m değeri iterasyonlar boyunca sabit tutulur veya aşamalı olarak düşürülür (Ahrari ve Atai, 2010).

Yukarıda tanımlanan parametreler doğrultusunda EBPM algoritmasının adımları şu şekildedir (Ahrari ve Atai, 2010):

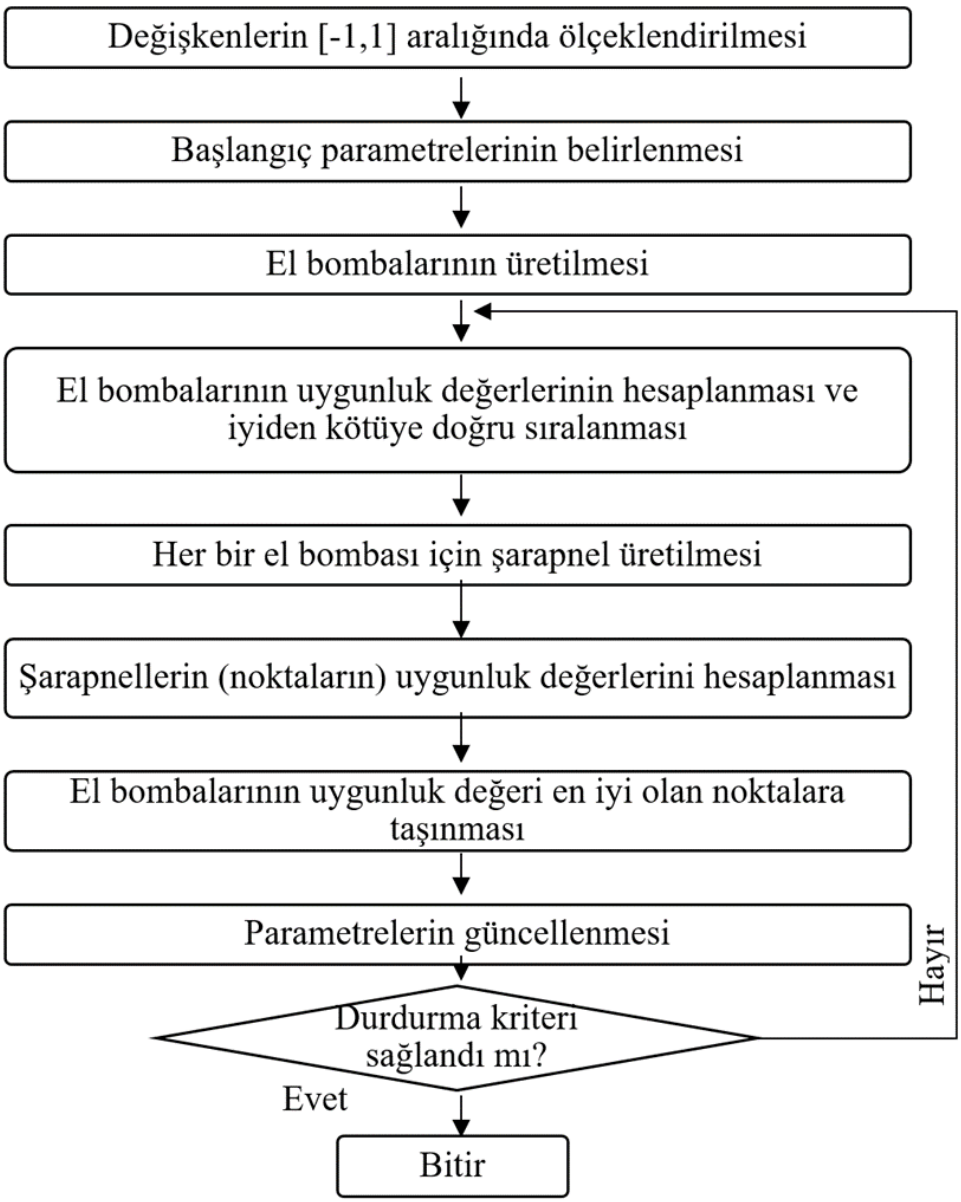
1. Bağımsız değişkenleri [-1,1] aralığında ölçeklendir.
2. Başlangıç parametrelerini (Np, Nq, $R_{t\text{-initial}}$, $R_{e\text{-initial}}$, R_{rd}, P_{TS}, maksimum iterasyon sayısı, m_{max}, m_{min} (veya m_{max}, m_{min} yerini sabit bir m değeri)) belirle; $R_e = R_{e\text{-initial}}$, $R_t = R_{t\text{-initial}}$ ve iterasyon no=1 atamalrını yap.
3. Ng adet el bombası üret. Bu bombalar n boyutlu uzayda rastgele üretilir. Fakat bombaların birbirine R_t mesafesinden daha yakın olmamasına dikkat edilir.

   (İterasyon sayısı < maksimum iterasyon sayısı olduğu sürece)

4. El bombalarını uygunluk değerlerine göre iyiden kötüye doğru sırala,
5. i=1 şeklinde ayarla,
6. i' inci el bombasının etrafında Eşitlik (4.1) kullanarak kabul edilebilir Nq adet nokta (X') üret. Eğer X' noktası uygun bölgenin dışında ise Eşitlik (4.3)yi kullanarak noktayı uygun bölgeye taşı. (X' noktası diğer el bombalarından en az R_t uzaklıkta ise kabul edilebilir.)
7. Şarapnellerin (noktaların) uygunluk değerlerini hesapla ve el bombasını uygunluk değeri en iyi olan noktaya taşı.
8. i=i+1; (bomba numarasını bir artır)
9. Eğer Ng>i ise adım 6' ya dön.

10. R_t değerini Eşitlik (4.4)ü kullanarak güncelle.
11. Eğer sabit alınmamış ise m değerini güncelle,
12. Eşitlik (4.5) göre R_e değerini güncelle.
13. Eşitlik (4.2) göre P değerini güncelle.
14. Maksimum iterasyon sayısına ulaşıldığında döngüden (adım 4-14) çık.

EBPM algoritmasının adımları Şekil 4.3'te özetlenmiştir.



Şekil 4.3. EBPM akış şeması

4.1. Optimal Arama Yönlü EBPM

Evrimsel yöntemlerde yeni çözümler komşuluk ilişkisi (fonksiyonu) ile belirlenir. Komşuluk ilişkisinin belirlenmesi metodun performansını önemli derecede etkilemektedir. Sezgisel yöntemlerin çoğunda komşuluk ilişkisinde rassal sayılara dayanan arama metotları kullanırlar ve rassallık yerel optimum noktalardan kurtulmak için korunur. EBPM'de kullanılan komşuluk ilişkisi de rassal sayılara dayanır.

Ahrari ve diğerleri kendi geliştirdikleri EBPM'ye olan güveni artırmak amacıyla EBPM'nin komşuluk ilişkisinde revizyona gitmişler ve ortogonal dizilerden faydalanarak yönteme optimal araştırma yönünü (optimal search direction) belirlemeye çalışan bir komşuluk ilişkisi eklemişlerdir (Ahrari ve diğ., 2009). Optimal arama yönlü El Bombası Patlatma Metodunda (OEBPM) temel mantık korunmaktadır. Fakat başlangıçta arama yönünü belirlemek için belirli sayıda şarapnel fırlatılarak hangi yönde hasarın daha çok olduğu belirlenmekte ve diğer şarapneller bu şarapnellerden elde edilen bilgiye dayanarak yanlı şekilde fırlatılmaktadır.

El bombasının bulunduğu yer X Eşitlik (4.1) kullanılarak hesaplanabilir. Fakat arama yönünün daha iyi anlaşılabilmesi için notasyonda küçük değişikliklere ihtiyaç duyulmuş ve Eşitlik (4.6) tercih edilmiştir (Ahrari ve diğ., 2009).

$$X' = X + r_1^p R_e \vec{d}\text{rnd} \tag{4.6}$$

Burada Eşitlik(4.1) ile aynı şekilde R_e patlama yarı çapı, r_1 [0,1] aralığında rassal sayı ve P bir sabittir. P sabiti aynı şekilde Eşitlik (4.2) ile güncellenmektedir. Eşitlik (4.1)den farklı olarak $\vec{d}_{rnd}$ rassal yönü ifade etmektedir (Ahrari ve diğ., 2009).

Patlama yönünü ($\vec{d}_{rnd}$) seçmek için standart normal dağılıma uygun rassal sayılar üretilir, sonra bu sayılar mutlak değerine bölünür. Olurlu bölgenin dışında kalan noktalar ise Eşitlik (4.3) kullanılarak olurlu bölgeye taşınır.

Optimal arama yönünü belirlemek için öncelikle her iterasyonda 2n şarapnel, her boyutta ve her yöne birer tane olmak üzere fırlatılır. Bu şarapnellerin meydana getirdiği tahribat analiz edilerek değişkenlerin uygunluk değeri üzerindeki etkisi tahmin edilir ve optimal arama yönü belirlenir. Daha iyi bir performans elde etmek için diğer şarapneller, optimal

arama yönüne yanlı olan rassal işlemlerle belirlenen yönlere fırlatılır. Şarapnellerin fırlatılacakları yönler Eşitlik (4.7) ile belirlenir (Ahrari ve diğ., 2009).

$$\vec{d} = m_{OSD}(r_1)\vec{d}_{OSD} + (1 - m_{OSD}(r_2)\,\vec{d}rnd, \quad 0 \leq m_{OSD} \leq 1 \tag{4.7}$$

$$\vec{d}_{GRS} = \frac{\vec{d}}{\|\vec{d}\|}$$

Burada; $\vec{d}_{GRS}$ şarapnellerin fırlatılacağı yön, $\vec{d}_{OSD}$ analiz sonucunda belirlenen optimal arama yönü, $\vec{d}_{rnd}$ rassal olarak belirlen yön ve r_1 ve r_2 $0 \leq r_1, r_2 \leq 1$ olmak üzere rassal sayılardır. M_{OSD} ise optimal arama yönünün ağırlığıdır. Sonuçta $\vec{d}_{GRS}$ yönündeki yeni noktalar Eşitlik (4.8) ile hesaplanır (Ahrari ve diğ., 2009).

$$X' = X + r_1^p R_e \vec{d}_{GRS} \tag{4.8}$$

Bu eşitlik, Eşitlik (4.6)'nın $\vec{d}_{rnd}$ yerine $\vec{d}_{GRS}$ konulmuş halidir. Böylelikle optimal arama yönü, yeni noktaların belirlenmesinde hesaba katılmış olur fakat diğer yönlerde de arama ihtimali devam etmektedir. Optimal arama yönünün ağırlığı olan M_{OSD}, ilk iterasyonlarda küçük seçilir ve sürekli olarak yükseltilir (Ahrari ve diğ., 2009).

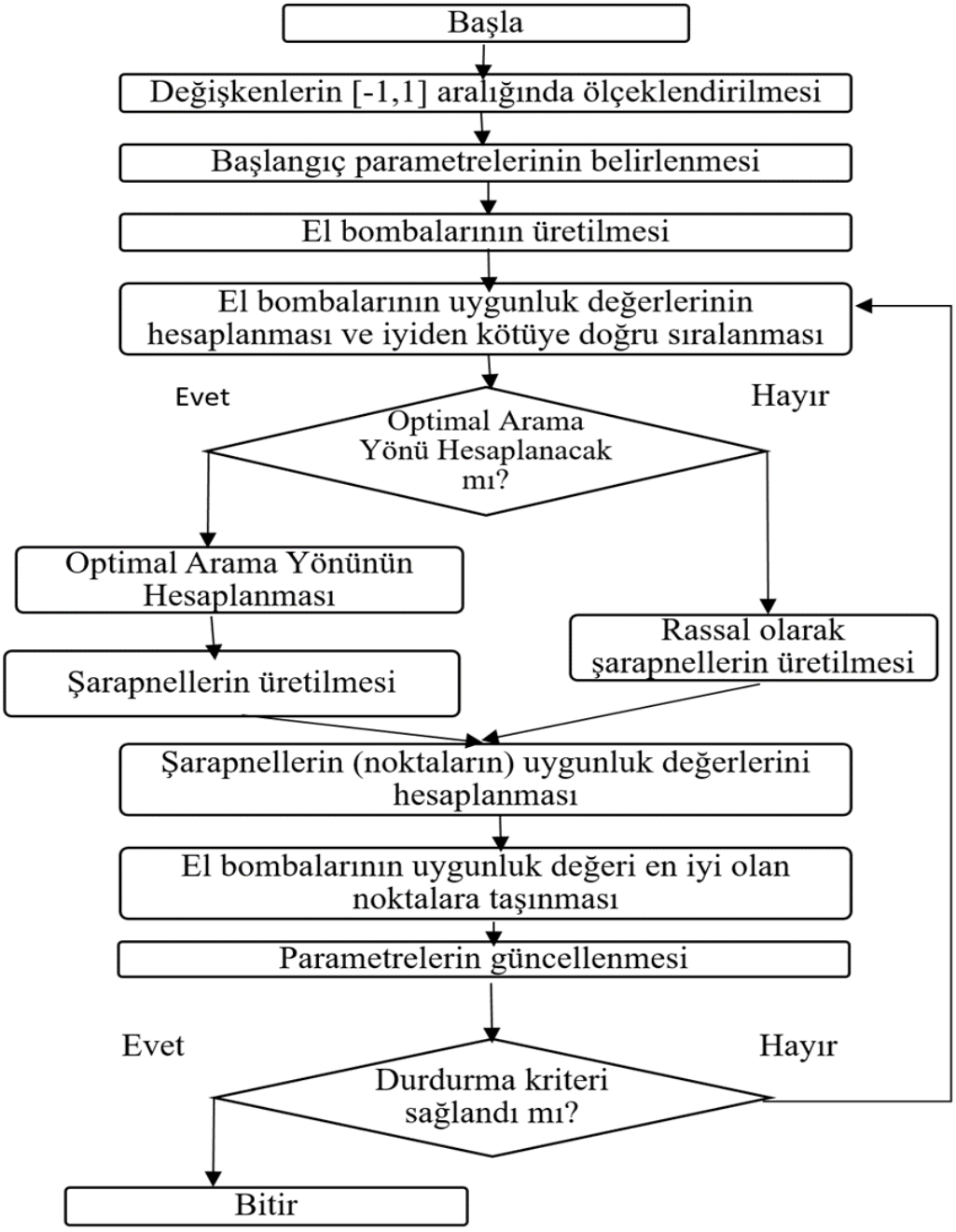
Bütün el bombalarının optimal yönü araştırma amacıyla ekstra 2n şarapnel fırlatması işlem yükünü artırmakta ve algoritmayı yavaşlatmaktadır. Bu nedenle bütün el bombaları için bu işlem yapılmaz; fonksiyonda kaç tane optimum nokta olduğu düşünülüyorsa, o kadar el bombası ek olarak 2n şarapnel fırlatılır. Yukarıdaki açıklamalar doğrultusunda OEBPM algoritmasının adımları şu şekilde sıralanmaktadır (Ahrari ve diğ., 2009):

1. Bağımsız değişkenleri [-1,1] aralığında ölçeklendir.
2. Başlangıç parametrelerini seç (Ng, Nq, $R_{t\text{-initial}}$, R_{rd}, P_{sin}, m_{min} ve maksimum iterasyon sayısı,)
3. $R_e = 2\sqrt{n}$, $R_t = R_{t\text{-initial}}$, $m_{OSD}=0$, iterasyon no=1 ve $P_{TS}=0.2$ (i=1,2…,n) atamalarını yap,
4. Ng adet el bombası üret. (Bu bombalar n boyutlu uzayda rasgele üretilir. Fakat bombaların birbirine R_t mesafesinden daha yakın olmamasına dikkat edilir.)
5. İterasyon sayısı < maksimum iterasyon sayısı olduğu sürece,
6. El bombalarını uygunluk değerlerine göre iyiden kötüye doğru sırala,

7. i=1 şeklinde ayarla,
8. Eğer, iterasyon no ≤0.1maksimum iterasyon ve i beklenen optimum nokta sayısından küçükse,
   a. 2n şarapnel fırlat (her boyutta ikişer ve her yönde birer tane),
   b. Bunlardan bu iterasyonda daha önce patlatılan bombalardan birinin ajan bölgesine girmeyen en iyi şarapneli "x'$_{OSD}$" şeklinde etiketle,
   c. Eğer bu bombanın konumu bu iterasyonda daha önce patlatılan bombalardan birinin ajan bölgesinde değilse, bu bombanın konumunu "x'$_{cur}$" şeklinde etiketle. Aksi takdirde x'$_{cur}$= [] olarak ata,
9. Bu bomba için, eğer hesaplanmış ise $\vec{d}_{OSD}$'yi kullanarak, $\vec{d}_{OSD}$ hesaplanmamış ise $\vec{d}_{rnd}$'yi kullanarak Nq adet şarapnel üret ve bunlardan en iyisini "x'$_{rnd}$" olarak etiketle,
10. Bombayı x'$_{OSD}$, x'$_{cur}$ ve x'$_{rnd}$ konumlarından en iyisine taşı;
    a. Eğer x'$_{rnd}$ en iyisi ise P_{TS} değerini bombanın şimdiki konumu ile eski konumu arasındaki farka göre güncelle
    b. Eğer uygunluk değerindeki iyileşme z iterasyon boyunca 10^{-6}/z sevisinin altında kalmış ise şarapnel sayısını yarıya düşür.
11. i=i+1, eğer i≤ Ng ise 8. Adıma git,
12. R_t ve m değerlerini düşür. Sonra Re değerini, $R_e = (2\sqrt{n})^m (R_t)^{1-m}$ eşitliği ile güncelle,
13. m_{OSD} değerini; $M_{OSD} = \sin(\frac{\pi}{2}(\frac{iterasyon\ \ no - 0.1Maks\ \ iterasyon}{0.9Maks\ \ iterasyon})^{p_{\sin}})$ ile güncelle,
14. İterasyonu bitir (adım 5).

OEBPM akış şeması Şekil 4.4'te özetlenmiştir. Yukarıdaki algoritma adımları incelendiğinde; belirli sayıda iterasyonda uygunluk değerinde istenilenin altında bir ilerleme olursa, bu bombanın ürettiği şarapnellerin sayısının yarıya düşürüldüğü görülmektedir. Bu EBPM'nin performansını artırmak için özellikle birkaç yerel optimum noktanın olduğu durumlarda gerçekleştirilen isteğe bağlı bir işlemdir. Eğer uygunluk değerinde z iterasyon süresince (10^{-6}/z) den daha az bir ilerleme sağlanmış ise o bombanın şarapnellerinin yarısı alınarak, bombaların tamamına rastgele dağıtılır. Burada z, Eşitlik (4.9) ile hesaplanır (Ahrari ve diğ., 2009).

$$z = \left[1.5 - \frac{\text{İterasyon No.}}{\text{Maksimum iterasyon sayısı}}\right] . \left[\frac{\text{Maks. iterasyon sayısı}}{20}\right] \quad (4.9)$$



Şekil 4.4. OEBPM akış şeması

4.2. Parametre Seçimi İçin Öneriler

Metasezgisel yöntemlerde başlangıç parametrelerinin seçiminin yöntemin performansı üzerinde büyük etkisi bulunmaktadır. Literatürde başlangıç parametrelerinin etkilerini araştırmaya yönelik çok sayıda çalışma (Engin ve Fıglalı, 2002; Yağmahan ve Yenisey, 2006), bulunmaktadır. Çalışmalarda en iyi parametrelerinin belirlenmesi için deney tasımı teknikleri başta olmak üzere farklı yaklaşımlar kullanılmıştır.

Ahrari ve diğerleri EBPM'de parametrelerin belirlenmesine yönelik aşağıdaki önerilerde bulunmuşlardır (Ahrari ve diğ., 2009):

EBPM'de el bombası sayısı yapay arı kolonilerindeki arı sayısı ile aynı rolü oynamaktadır. Sıradan optimizasyon problemleri için 2-3 el bombası ve her bomba için 10-15 şarapnel yeterli olmaktadır. Karmaşık problemler için el bombalarının sayısını artırmaktansa şarapnel sayısını artırmak tercih edilmedir. Gerek duyulması halinde her bir bomba için şarapnel sayısı 100 kadar artırılabilir. El bombası sayısının, beklenen optimal nokta sayısının 1.5 veya 2 katı alınması yerinde olacaktır.

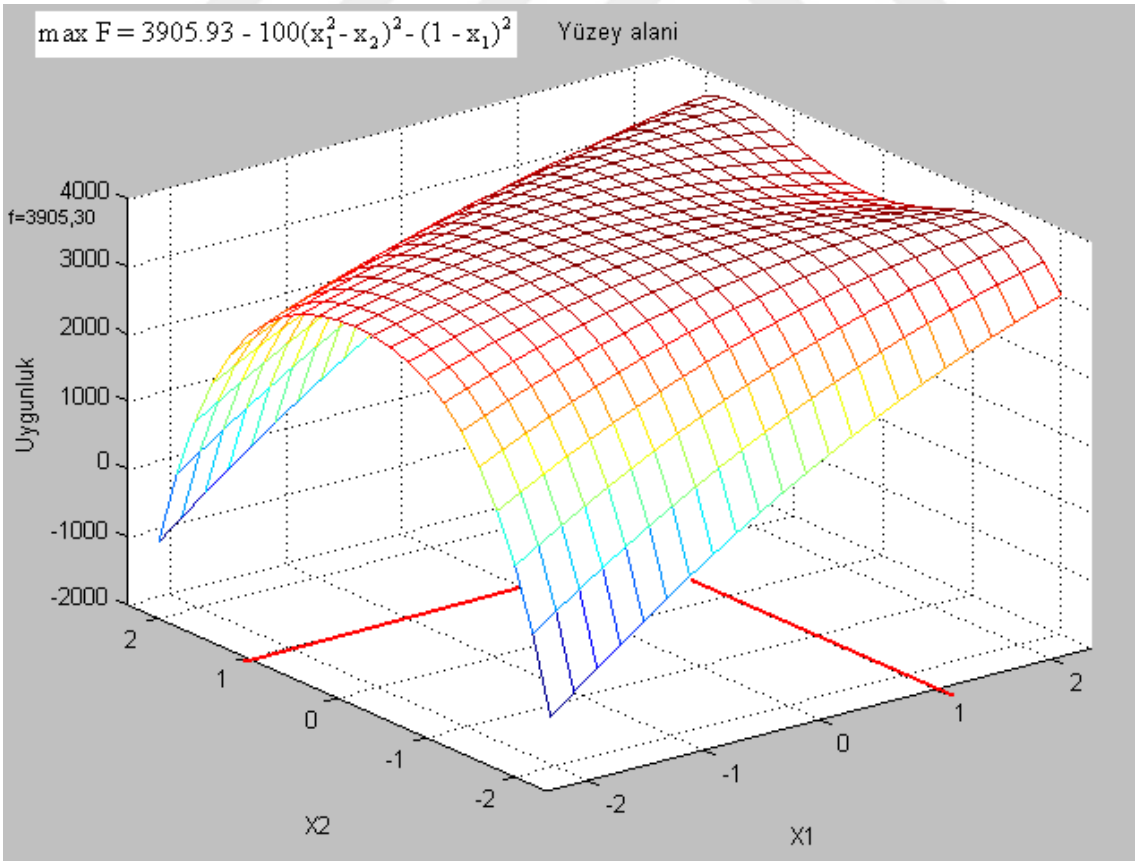
$R_{t\text{-}initial}$ değerinin yeterince büyük seçilmesi ve kademli olarak R_{rd} oranında küçültülmesi gerekmektedir. $R_{t\text{-}initial}$'nin büyük seçilmesi bombaların ilk iterasyonlarda belirli noktalarda toplamasını önlemekte, böylece yerel optimum noktalara takılma riskini azaltmaktadır. Fakat $R_{t\text{-}initial}$ çok büyük seçilirse bombaların çözüm uzayına yerleştirilmeleri uzun zaman alacaktır. R_{rd} oranı ise ajan bölgesin son iterasyondaki büyüklüğünü belirlemekte kullanılmaktadır. Eğer bu oran küçük seçilirse daha hassas sonuçlar edilebilmekte fakat yerel optimumlara takılma riski artmaktadır. Bu oran büyük seçilirse daha global bir arama yapılmakta fakat sonuçların hassasiyeti azalmaktadır. R_{rd} ve $R_{t\text{-}initial}$ parametreleri kullanılan bomba sayısına ve problemin yapısına göre seçilmedir. Bu parametreler için henüz kesin öneriler yoktur.

Yenilenmiş EBPM' de patlama çapı $R_e = 2\sqrt{n}$ olarak alınmakta ve kademeli olarak R_t büyüklüğüne bağlı olarak düşülmektedir. R_e değeri önceki başlıklarda verilen Eşitlik (4.5) kullanılarak güncellenmektedir. Eşitlik (4.5)de yer alan m değeri başlangıçta 1 alınıp kademeli olarak sıfıra yakın bir değere kadar (genellikle $m_{mim}=0.2$) düşürülebilir.

Optimal arama yönünün ağırlığı olan P_{sin} parametresi [0.1,10] aralığında fonksiyonun düzenli ve simetrik oluşana bağlı olarak seçilmelidir. Eğer fonksiyon simetrik birkaç parçadan oluşuyorsa P_{sin} değeri küçük seçilerek m_{OSD} ağırlığının hızlı bir şekilde yükselmesi sağlanır. Eğer fonksiyon simetrik parçalardan oluşmuyorsa ve düzensiz ise P_{sin} değeri büyük seçilerek rassal aramanın geliştirilmesi sağlanır.

4.3. Optimal Arama Yönlü EBPM'nin Grafiksel Analizi

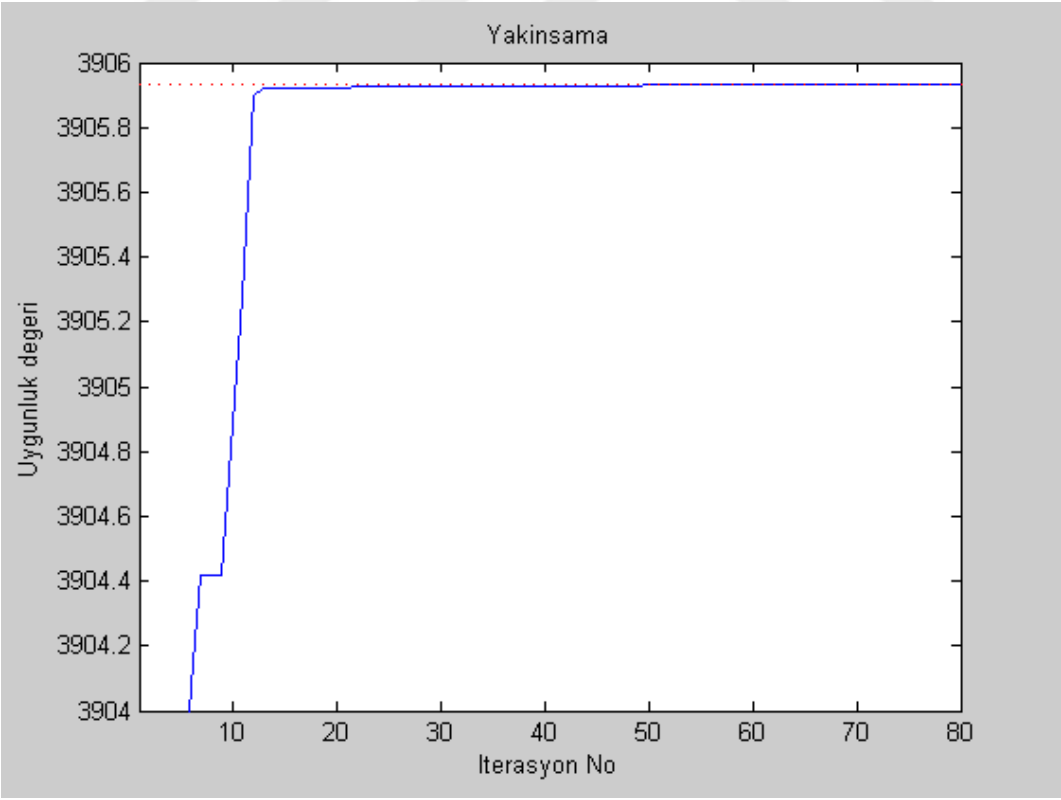
Ahrari ve diğerleri, EBPM'nin global optimuma ulaşma konusunda en az literatürde kabul görmüş diğer sezgisel yöntemler kadar başarılı olduğunu, hız açısından değerlendirildiğinde ise EBPM'nin diğer yöntemlerden daha iyi olduğunu; özellikle çok sayıda yerel optimuma sahip fonksiyonlarda EBPM'nin diğer yöntemlerden üstün olduğunu ve bu üstünlüğün sağlanmasında EBPM'ye özgü olan ajan bölgesi belirlenmesinin önemli bir rolünün olduğunu belirtmişlerdir (Ahrari ve diğ., 2009; Ahrari ve Atai, 2010).



Şekil 4.5. De Jong fonksiyonun formülü ve yüzey alanı

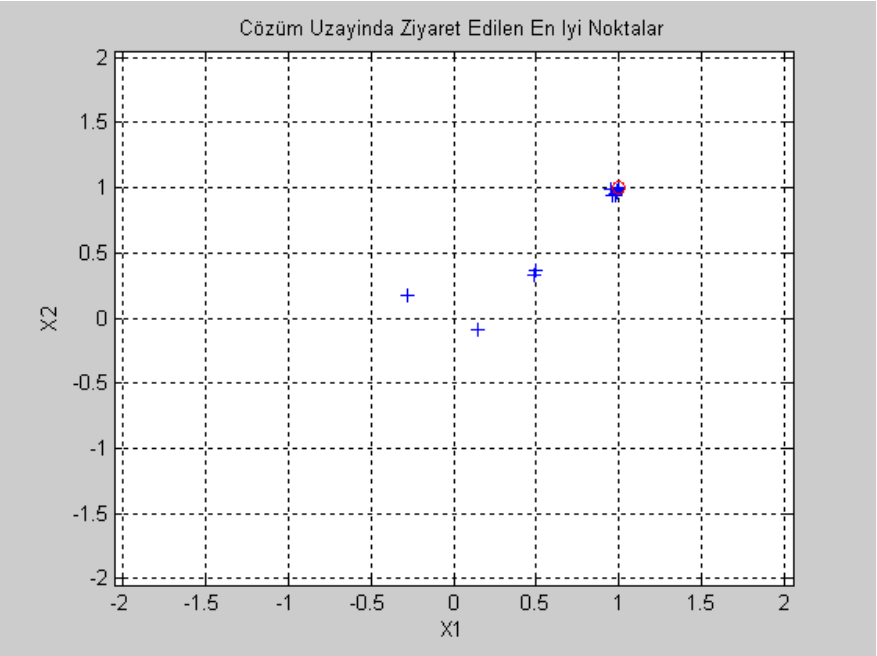
Bu başlık altında grafikler yardımıyla EBPM'nin çözüm uzayında ziyaret ettiği noktaların dağılımı ve optimum noktaya nasıl yaklaşıldığı incelenecektir. Bu amaçla OEBPM, Matlab programı kullanılarak yazılım haline getirilmiş olup yazılımda Sayın Ahrari'den elektronik posta yolu ile temin edilen kaynak kodlardan da faydalanılmıştır. Analiz için Şekil 4.5'te verilen [−2.048, 2.048] aralığında tanımlı De Jong kıyaslama fonksiyonu kullanılmıştır.

Şekil 4.6'da her iterasyonda optimum noktaya ne kadar yaklaşıldığı görülmek olup Şekilden de anlaşıldığı üzere uygunluk değeri ilk iterasyonlarda hızlı bir şekilde artmış daha sonra yavaşlamıştır. Bu EBPM genel özelliklerinden biri olup; ilk iterasyonlarda global arama yapılırken, ilerledikçe belirli bölge veya bölgelere yoğunlaşmıştır. Bu deneyde 475 nokta ziyaret edildikten sonra 27. iterasyonda hata 10^{-3} seviyesine inmiştir. Fakat algoritma maksimum iterasyon sayısına kadar durmayacak şekilde ayarlandığından çalışmaya devam etmiş ve son (80.) iterasyonda hesaplanan hata $2x10^{-6}$ seviyesine düşmüştür.



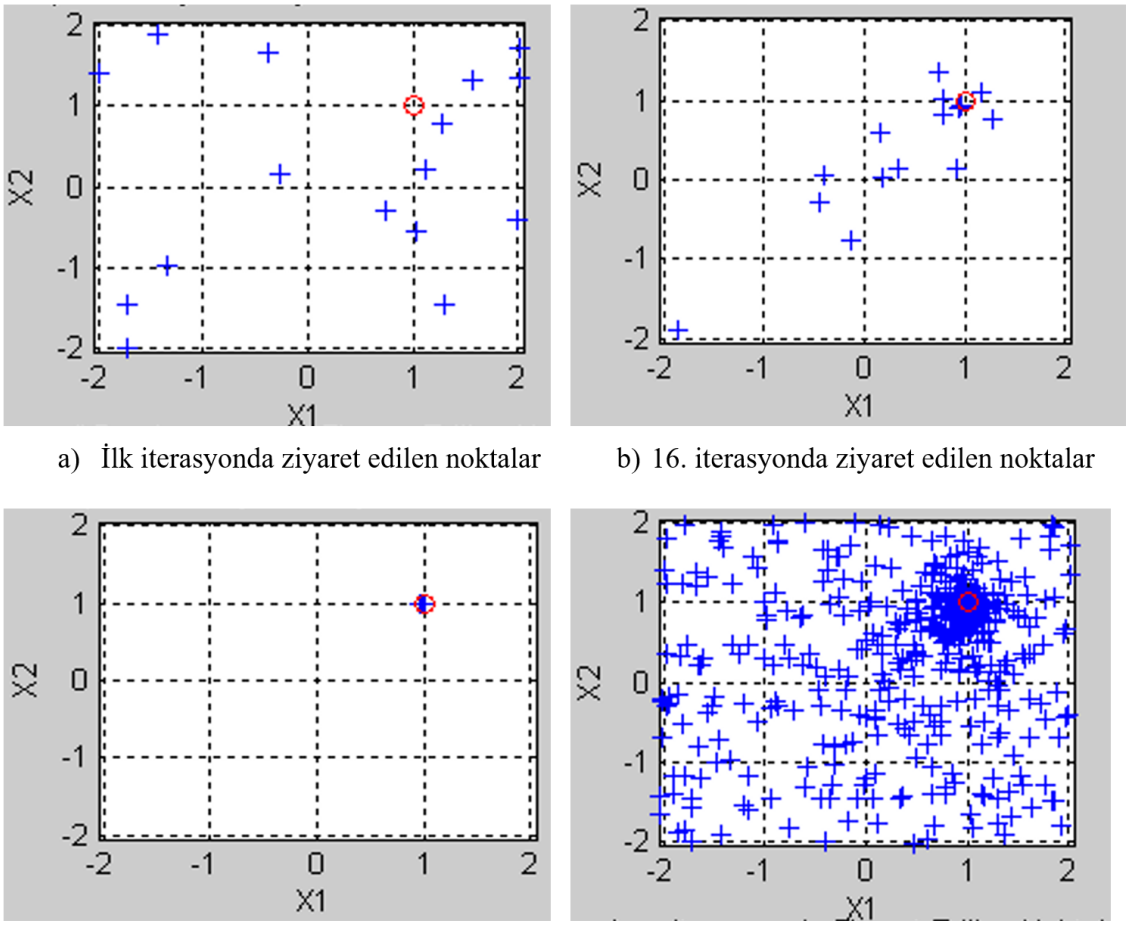
Şekil 4.6. Her iterasyonda optimum noktaya yakınsama durumu

Şekil 4.7'de her iterasyonda ziyaret edilen en iyi noktalar gösterilmiş olup nokta sayısı, noktaların çakışması nedeniyle az görülmektedir. Özellikle 27. iterasyonda optimal sonucun yakalanmasından sonra bütün noktalar tek bir nokta şeklinde görülmektedir.



Şekil 4.7. Her bir iterasyonda ziyaret edilen en iyi noktalar

Şekil 4.8'de ise ziyaret edilen bütün noktalar ile 1., 16. ve 80. iterasyonda ziyaret edilen noktalar gösterilmiştir. Şekil 4.8.a'da görüldüğü üzere ilk iterasyonda noktalar çözüm uzayına neredeyse homojen olarak dağılmıştır; opitimum noktaya en uzak koordinatlar dahil bütün çözüm uzayı kabaca taranmıştır. Daha sonra orijinden kuzeydoğu istikametine uzanan bir doğru şeklinde yoğunlaşmaya başlamıştır. Şekil 4.8.b'de 16. iterasyonda ziyaret edilen noktaların konumları verilmiştir. Bu doğru sürekli küçülerek zamanla optimum sonucun etrafında yoğunlaşmış ve giderek optimum sonuca yaklaşmıştır. Şekil 4.8.c'de görüldüğü üzere son iterasyonda ziyaret edilen bütün noktalar hedefin etrafında neredeyse tek bir nokta halinde birleşmiştir. Şekil 4.8.d'de maksimum iterasyon sayısı olan 80 iterasyon boyunca ziyeret edilen bütün noktalar gösterilmiştir. Noktalar çözüm uzayının tamamına dağılmakla birlikte opimum çözümün etrafında yoğunlaşmıştır.



a) İlk iterasyonda ziyaret edilen noktalar

b) 16. iterasyonda ziyaret edilen noktalar

c) Son iterasyonda ziyaret edilen noktalar

d) Zeyaraet edilen bütün noktalar

Şekil 4.8. Ziyaret edilen bütün noktalar (a), ilk (b), 16. (c) ve son (d) iterasyonda ziyaret edilen noktalar

Sonuç olarak ilave özelliklerle EBPM'nin performansının artmıştır, fakat ilave edilen özellikler yöntemin biraz daha karmaşık bir hal almasına neden olmuştur. Özellikle yöntemin yazılıma çevrilmesi, ilk haline göre çok daha zordur.

4.4. Gerçek Sayılardan Sıralama Vektörüne Dönüşüm

EBPM'nin iş çizelgeleme problemlerinde kullanılabilmesi için gerçek sayılardan oluşan EBPM vektörlerinin sıralama değişkenlerine dönüştürülmesi gerekmektedir. PATÇP'deki iş sıraları ile şarapnel parçaları arasında ilişki, n sayıda iş için n boyutlu konumlar kullanılarak sağlanır.

Konumların iş sıralarına dönüşümünde kullanılacak kodlama yönetiminin metodun başarısını etkilediği bilinmektedir. Daha kolay olması ve çizelgeleme problemleri ile daha

iyi performans vermesi nedeniyle bu çalışmada En Küçük Pozisyon Değeri (EKPD) yöntemi kullanılmıştır.

Bu yöntemde elde edilen gerçek çözüm vektörü küçükten büyüğe doğru sıralanır. En küçük değere sahip olan iş birinci sıraya atanır ve bütün işlerin sırası belirlenene kadar atama işlemine davam edilir. Örnek olarak Tablo 4.1'de X'= (-0.90, 0.50, -0.20, 0.80, 0.15,-0.75) konumunun Π=(1, 5, 3, 6, 4, 2) permütasyonuna nasıl dönüştürüldüğü gösterilmektedir. Şevkli ve Yenisey (2006), uygulanan bu yöntemi rasgele gösterim metodunun (Bean, 1994) permütasyon akış tipi çizelgeleme problemine uyarlanması olarak tanımlamaktadır. Bu yöntem Taşgetiren ve diğ.(2007), Liu ve diğ. (2007) ve Dasgupta ve Das, (2015) tarafından örnekleri ile açıklanmıştır.

Tablo 4.1. Konum bilgileri ve EKPD yöntemi ile hesaplanan iş sıraları

Boyut	1	2	3	4	5	6
Konum Değeri (X')	-0.90	0.50	-0.20	0.80	0.15	-0.75
İş sırası	1	5	3	6	4	2

Gerçek sayılardan sıralama vektörüne dönüşüm eklenmesi ile el bomsa patlatma algoritması aşağıdaki şekilde ifade edilebilir:

1. Parametreleri belirle,
2. Eğer ön görülmüşse optimal arama yönü için 2n şarapnel üret,
3. Gerçek çözüm vektörlerini sıralama vektörüne dönüştür ve şarapnellerin uygunluk değerlerini hesapla,
4. Bombaları patlatarak belirlenen sayıda şarapnel (aday çözüm) üret,
5. Gerçek çözüm vektörlerini sıralama vektörüne dönüştür ve şarapnellerin uygunluk değerlerini hesapla,
6. Bombaları yeni yerlerine taşı,
7. Eğer varsa uygunluk değerinde iyileşme olmayan bombaların şarapnellerini paylaştır,
8. Parametreleri güncelle,
9. Maksimum iterasyon sayısına ulaşıncaya kadar 2-8. işlemleri tekrarla.

4.5. Ayrık El Bombası Patlatma Metodu

Bu raporun daha önceki bölümlerinde belirtildiği gibi Ahrari ve diğerleri (2009) tarafından sunulan EBPM gerçek sayı vektörleri ile çalıştığından çizelgeleme problemlerinde kullanılabilmesi için gerçek sayı vektörleri ile sıralama vektörleri arasında bir dizi dönüşüm işlemi yapılması gerekmektedir. Bu durum literatürde kabul görmüş bazı metasezgiseller için de geçerlidir. Örneğin literatürde kullanımı oldukça yaygın olan Diferansiyel Gelişim Algoritmasının orijinal hali de gerçek sayı vektörlerini kullanmakta ve çizelgele problemlerinde kullanılırken gerçek sayılardan sıralama vektörlerine dönüşüm için ek işlemlere ihtiyaç duymaktadır.

Taşgetiren ve diğerleri (2007) ile Pan ve diğerleri., (2008a) Diferansiyel Gelişim Algoritmasını iş çizelgeleme problemlerine uyarlarken gerçek sayı vektörleri yerine doğrudan sıralama vektörlerini kullanan Ayrık Diferansiyel Gelişim Algoritmasını (DDE) önermişlerdir. DDE'yi oluşturan temel özellik çizelgeleme problemlerinin [2,3,1,4,5] gibi gerçek iş sıralarından oluşan genler şeklinde kodlanmasıdır. DDE ile benzer şekilde ayrık Yapay Arı Kolonisi Algoritması (Pan ve diğ., 2011) ve ayrık parçacık sürü algoritması (Pan ve diğ., 2008b) geliştirilmiştir.

Bu çalışmada ise benzer şekilde Ayrık EBPM (AEBPM) geliştirilmeye çalışılmıştır. Bilindiği üzere EBPM'de çözüm alternatifleri el bombaları ve bunlardan üretilen şarapnellerle temsil edilmektedir. Bir el bombasının yeri;

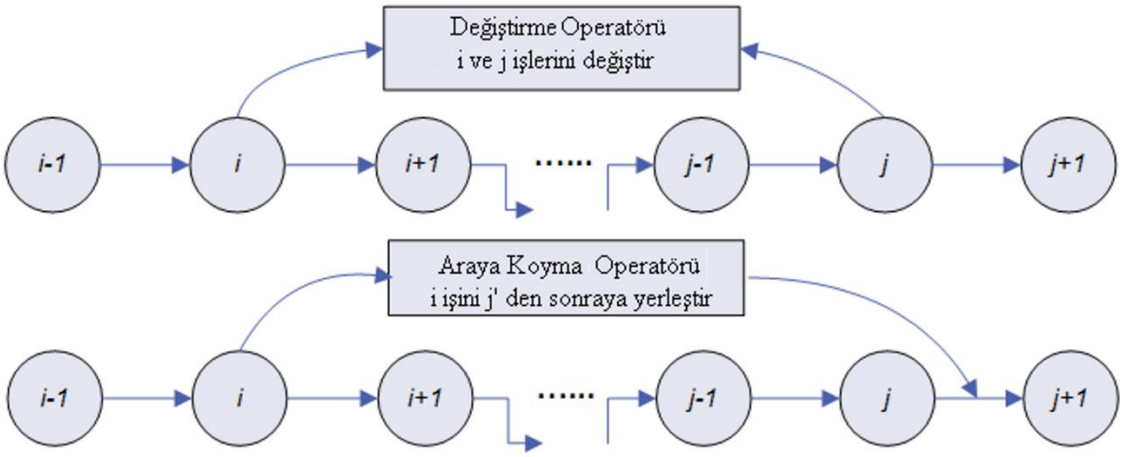
$$X = \{X_m\}, m = 1,2, \dots, n;\ X_m = [-1,1]$$

şeklinde çok boyutlu uzayda gerçek koordinatlarla tanımlanır. Burada n problemin boyutunu başka bir ifadeyle değişken sayısını temsil eder. AEBPM'de ise gerçek koordinat sayıları kullanılmaz $X_i=[X_1,X_2,\dots,X_n]$ şeklinde iş sırları kullanılır. Örneğin 5 işten oluşan bir problemde X=[5,4,3,2,1], X=[1,4,3,2,5], X=[1,2,3,4,5] gibi n! iş sırası oluşturulabilir. Bunlardan her biri çözüm alternatifleridir ve AEBPM'de bir el bombasını veya şarapneli tanımlarlar.

EBPM'den AEBPM'ye dönüştürülürken yeniden tanımlanması gereken diğer bir nokta ise komşuluk ilişkisidir. Orijinal EBPM'de komşuluk ilişkisi şarapnel üretimine dayanmakta olup herhangi bir X noktası (el bombası) etrafında şarapnel üretimi Eşitlik

(4.1) veya Eşitlik (4.6) ile yapılmaktadır. Fakat bu eşitlikler sadece gerçek sayı vektörleri ile kullanılabildiğinden AEBPM'de şarapnel üretimi için yeni bir yönteme ihtiyaç duyulmaktadır.

Şarapnel üretimi için farklı yöntemler kullanılabilir. Bu operatörler Şekil 4.9'da açıklamıştır. Bu çalışmada araya koyma operatörü kullanılmıştır.



Şekil 4.9. Komşuluk operatörleri (Li ve Yin, 2012)

Bu bilgiler ışığında AEBPM'nin işlem adımları aşağıda verilmiştir.

1- Başlangıç parametreleri ve durdurma parametreleri belirlenir. Başlangıçta durdurma kriteri dışında Nq, Ng, R_t ve R_{rd} olmak üzere 4 parametrenin belirlenmesi zorunludur. ($R_{e\text{-}initial}= 2\sqrt{n}$ alınmıştır.)
2- Rassal olarak veya değişik metotlar kullanılarak Nq adet başlangıç çözüm (el bombası) belirlenir. Her bir bomba gerçek iş sıralarından oluşmaktadır. Bu aşamada bombaların birbirinin ajan bölgesine girmesine yani birbirine $R_{t\text{-}initial}$ mesafesinden daha çok yaklaşmasına izin verilmemelidir.
3- Her bir bombanın uygunluk değeri hesaplanarak bombalar artan sıraya göre (iyiden kötüye doğru) sıralanır.
4- Her bir bomba için Nq adet uygun şarapnel üretilir. Şarapnellerin uygun kabul edilebilmesi için kendisinden önce patlatılan bombaların ajan bölgesine girmemesi gereklidir.

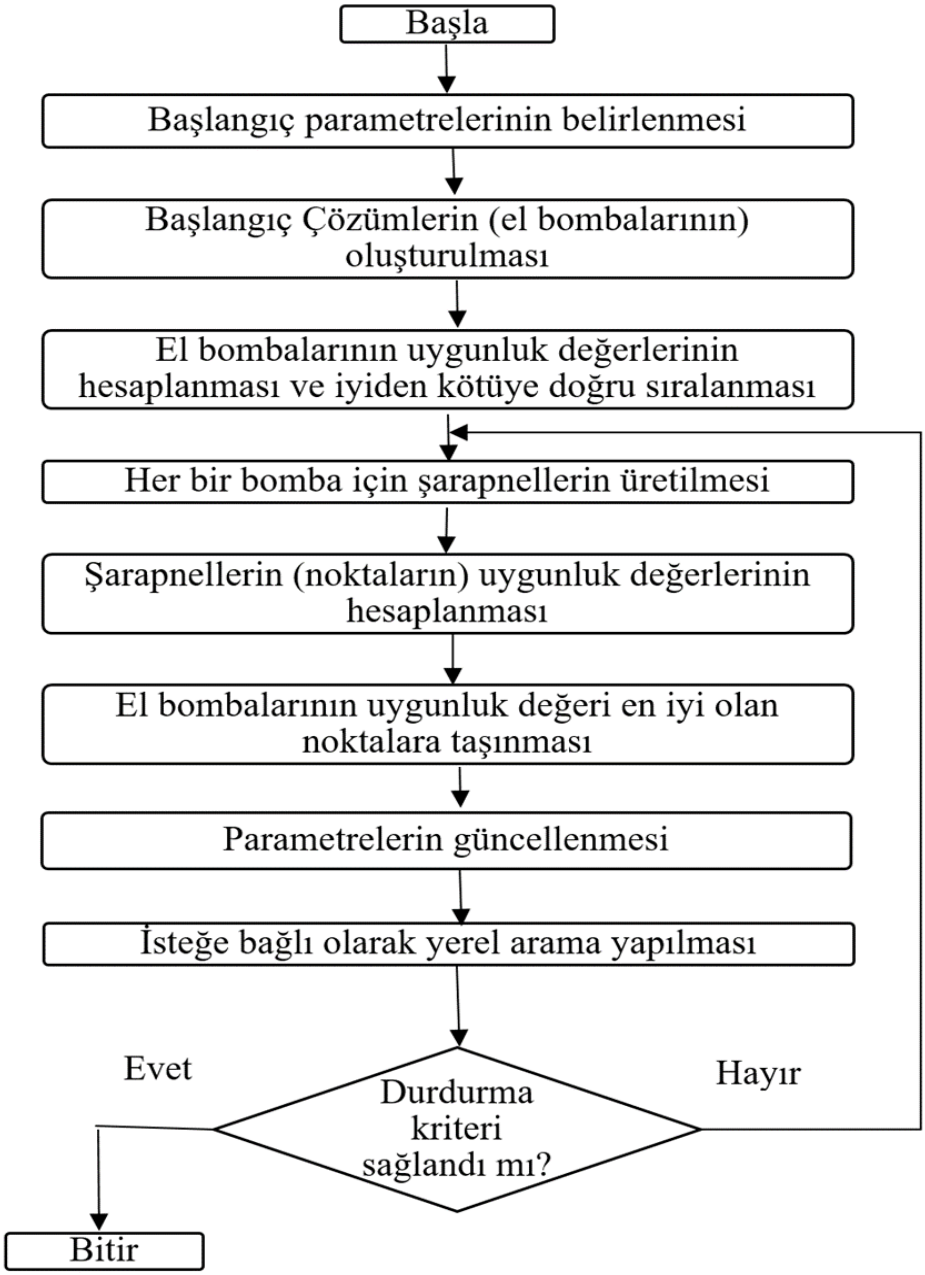
5- Üretilen şarapnellerin uygunluk değerleri hesaplanır.
6- Eğer şarapnellerden birinin uygunluk değeri ait olduğu el bombasından daha iyi ise el bombası şarapnelin olduğu noktaya taşınır.
7- İsteğe bağlı olarak yerel arama yapılır.
8- Durdurma kriterine ulaşıldığında işlem sonlandırılır.

Burada Ng, el bombası sayısı; Nq, şarapnel sayısı; R_t, ajan yarıçapını; Rrd ilk iterasyondaki R_t değerinin son iterasyondaki R_t değerine oranını ifade eder. AEBPM'de durdurma kriteri olarak maksimum iterasyon sayısının veya maksimum CPU zamanının kullanılması mümkündür. ABPM akış şemesı Şekil 4.10'da verilmiştir.

Tablo 4.2'de EBPM, OEBPM ve AEBPM için başlangıçta araştırmacı tarafından belirlenmesi gereken parametreler verilmiştir. AEBPM'de belirlenmesi gereken parametre sayısı diğer yöntemlere göre daha azdır.

Tablo 4.2. EBPM, OEBPM ve AEBPM için başlangıç parametreleri

No	Parametre	EBPM	OEBPM	AEBPM
1	El bombası sayısı (Ng)	+	+	+
2	Optimal arama yönünü belirlemekte kullanılacak el bombası sayısı (Nq-oay)	-	+	-
3	Bir bombanın fırlatacağı şarapnel sayısı (Nq)	+	+	+
4	Patlama yarıçapı ($R_{e\text{-}initial}$)	+	+ ($2\sqrt{n}$)	($2\sqrt{n}$)
5	Ajan bölgesi yarıçapı ($R_{t\text{-}initial}$)	+	+	+
6	İlk iterasyondaki R_t değerinin son iterasyondaki R_t değerine oranını (R_{rd})	+	+	+
7	Patlama yarıçapını güncellemede kullanılan sabit (m_{min})	+	+ (0,2)	-
8	Optimal arama yönün ağırlığını güncellemede kullanılan sabit(P_{sin})	-	+	-
9	Ajan Bölgesi Araştırma Yoğunluğu (P_{TS})	+ (0,2)	+ (0,2)	-
10	Durdurma kriteri (İterasyon sayısı veya maksimum çalışma süresi)	+	+	+



Şekil 4.10. AEBPM akış şeması

Yerel arama yöntemleri daha iyi sonuçlar elde edebilmek amacıyla kullanılmaktadır ve sırlama problemleri için geliştirilmiş birçok yerel arama yöntemi bulunmaktadır. Bunlar, vektörün belirli bir kısmının kesilerek başka bir yerine eklenmesi, kısmi sıralama değişiklikleri ya da vektörün bir bölümünün tersine çevrilerek tekrar vektör içine eklenmesi ve benzeri birçok çeşidi içermektedir (Terzi, 2009). Bu çalışmada Pan ve diğ.,

(2008a) tarafından önerilen Refaranslanmış Yerel Arama (Referenced Local Search, RLS) yöntemi kullanılmıştır.

4.6. El Bombası Patlatma Metodu ile İlgili Literatür Çalışmaları

EBPM ile ilgili ilk çalışmada sunulan metodun performansı sürekli ve çok modlu matematiksel fonksiyonlar üzerinde test edilmiş ve literatürde kabul görmüş Genetik Algoritma ve Yapay Arı Kolonisi gibi metotlarla karşılaştırılmıştır (Ahrari ve Atai, 2010). Ahrari ve arkadaşları tarafından yapılan diğer bir çalışma ile EBPM'nin komşuluk ilişkisi optimal arama yönü özelliği eklenerek geliştirilmiştir (Ahrari ve diğ., 2009). Yine Ahrari ve arkadaşları evrimsel algoritmaların sağlamlığını değerlendirdikleri çalışmalarında EBPM'nin küresel minimumun değerine hızlı yakınsadığı belirtilmiştir (Ahrari ve diğ., 2010).

Pallantla and Singh EBPM'yi maksimum ağırlık klik probleminin çözümünde kullanmışlardır. NP-Zor bir problem olan maksimum ağırlık klik problemi, ağırlıklı ağda verilen bir noktaya bağlı maksimum ağırlıklı alt ağı araştırır. Bu çalışma öncelikle sürekli problemler için geliştirilen EBPM'nin kesikli problemlerin çözümünde kullanıldığı bilinen ilk çalışmadır (Pallantla ve Singh, 2012).

Rao ve diğerleri; doğrusal olmayan sürekli büyük ölçekli problemlerin optimizasyonu için Öğretme-Öğrenmeye Dayalı Optimizasyon yöntemini sundukları çalışmada metodun performansını diğer metotlarla karşılaştırmıştır. Karşılaştırılan metotlar arasında EBPM de bulunmaktadır. Çalışmada sunulan Öğretme-Öğrenmeye Dayalı Optimizasyon metodunun performansının EBPM'den daha iyi olduğu belirtilmektedir (Rao ve diğ., 2012).

Sürekli optimizasyonda global minimum için özerk olmayan olasılıksal arama isimli çalışmada optimizasyon metotlarının kararlılığı ve yakınsaması konusunda genel bir çerçeve sunulmaktadır. Bu kapsamda EBPM'nin matematiksel yakınsama analizi de sunulmaktadır (Tarłowski, 2014).

Etkili kümelenme için el bombası patlama yaklaşımının genişletilmesi isimli çalışmada, modifiye EBPM ve K- Ortalama Kümeleme metodu birleştirilerek etkin karma bir kümeleme algoritması sunulmaktadır (Ghanavati ve diğ., 2016)

Öğretme Öğrenme Temelli Optimizasyon (TLBO) algoritması kullanılarak kullanarak ısı borusunun optimum tasarımı çalışmasında sunulan TLBO algoritması ile daha önce aynı problemin çözümünde kullanılan EBPM'nin de içinde olduğu bazı metotların performansları karşılaştırılmaktadır. Bu çalışmada TLBO algoritmasının EBPM'den daha iyi sonuçlar ürettiği belirtilmektedir (Rao ve More, 2015).

Mishra ve diğerleri (2016) güç kalitesinin iyileştirilmesi için fotovoltaik beslemeli DSTATCOM optimizasyonu için Öğretme Öğrenme Temelli Optimizasyon (TLBO) algoritmasını önermişlerdir. Çalışmada önerilen yöntem EBPM ile karşılaştırılmış ve TLBO algoritmasının global optimum çözüme EBPM'den daha hızlı yaklaştığı belirtilmiştir. Mishra ve Ray (2016) aynı optimizasyon problemi için JAYA isimli yeni bir metot önermişlerdir. Çalışmada TLBO algoritması, EBPM ve JAYA metodunun performansı karşılaştırılmış ve EBPM'nin bunlar arasında en yavaşı olduğu belirtilmiştir.

Bouchekara ve diğerleri (2016) çok amaçlı optimal güç akışı (MOPF) problemini çözmek için bulanık bir EBPM önermişlerdir.

Mouwafi ve diğerleri (2016) akıllı güç sistemlerinde faz ölçüm birimlerinin optimum şekilde yerleştirilmesinde Yapay Arı Kolonisi Algoritması temelli çok aşamalı bir metot önermişler ve önerdikleri metodun performansını daha önce aynı konuda yapılmış olan EBPM çalışmasının sonuçları (Khavari ve diğ., 2011) ile karşılaştırmıştır.

Rakhade ve diğerleri (2017) plaka kanatlı tip eşanjör optimizasyonu için EBPM'yi kullanmıştır.

Zhang ve diğerleri (2015) EBPM'nin güvenilirlik ve hızlı yakınsama özelliklerinden faydalanarak YAKA'nın performansını geliştirmek için 2 farklı melez YAKA geliştirmiş ve performanslarını değerlendirmiştir. Melez algoritmanın birinde işçi arı aşamasında diğerinde ise gözcü arı aşamasında EBPM kullanılmıştır. Çalışmada melez kullanımın birçok durumda performansı geliştirdiği fakat karmaşıklığı artırdığı ve zaman tüketimini artırdığı rapor edilmiştir. Benzer başka bir çalışmada ise EBPM ve Cauchy Operatörü kullanılarak melez bir YAKA geliştirilmiştir (Zheng ve diğ., 2015).

Salhi ve diğerleri (2016) EBPM kullanılarak geliştirilmiş melez YAKA'yı optimal güç akışının hesaplanmasında kullanmıştır. EBPM kullanılarak geliştirilmiş melez YAKA;

dinamik ekonomik emisyon dağıtımı probleminin çözümünde (Marouani ve diğ., 2018) ve kablosuz sensör ağlarında (Famila ve diğ., 2020) da kullanılmıştır.

Literatürde ulaşılan 20'den fazla çalışma incelenmiştir. Bu çalışmalarda EBPM'nin kesikli gerçek hayat problemlerine de uygulandığı görülmüştür fakat iş çizelgeleme problemlerinde EBPM'nin kullanıldığı bir çalışmaya rastlanmamıştır.

EBPM'nin diğer metasezgisel yöntemlerle karşılaştırıldığı çalışmalarda EBPM'nin çoğunlukla daha yavaş olduğu ve performansının diğer yöntemlerin gerisinde kalığı görülmüştür. Çalışmalar arasında EBPM'nin hızlı yakınsama özelliğinden faydalanmak için oluşturulan melez yöntemler dikkat çekmektedir. Bu şekilde geliştirilen yöntemlilerin optimum çözüme yakınsama performanslarının artığı fakat melez kullanımın bağlı olarak zaman tüketiminin de artığı rapor edilmiştir.

5. BULGULAR VE TARTIŞMA

Yukarıda yapılan literatür çalışmaları başlığında da görüldüğü gibi EBPM teorik ve gerçek yaşam problemlerine uygulanmış ve performansı test edilmiştir. EBPM'nin uygulandığı problemler arasında kesikli problemlerde bulunmaktadır.

Yapılan çalışmalar arasında EBPM'nin bu özelliğinin metodun performansına olan etkisini inceleyen ve EBPM'nin iş çizelgeleme problemlerinin çözümünde kullanıldığı çalışmalara rastlanmamıştır. Çalışmanın bu bölümünde EBPM'nin iş çizelgeleme problemlerinin çözümündeki performansının ölçülmesi ve bazı özeliklerinin araştırmasına yönelik deneysel çalışmalar yapılmıştır. Yapılan deneysel çalışmalar hedefleri bakımından şu şekilde sınıflandırılabilir:

- Normal dağılıma uygunluk testleri,
- Ajan bölgesinin ve optimal arama yönünün etkisinin değerlendirilmesi,
- EBPM'nin doğrulanması ve performansının değerlendirilmesi,
- EBPM'nin performansının Diferansiyel Gelişim Algoritması ile karşılaştırılması,
- EBPM tarafından üretilen başlangıç çözümlerin DGA'nın performansına katkısının araştırılması,
- NEH algoritması ile üretilen başlangıç çözümlerin kullanılmasının sağladığı katkının incelenmesi,
- EBPM'nin iterasyonlar süresince yakınsama durumunun incelenmesi,
- Ayrık EBPM'nin performansının ölçülmesi.

Permütasyon akış tipi çizelgeleme problemi için literatürde yaygın 2 grup test probleminin kullanıldığı görülmüştür. 1. grupta Taillard (Taillard, 1993) tarafından sunulan 120 adet akış tipi test problemi yer almaktadır. Taillard; iş sayısı 20 ile 500, makine sayısı 5 ile 20 arasında değişen her grupta iş ve makine sayıları eşit 10'ar problemin olduğu 12 grup test problemi oluşturmuştur. Bu test problemlerine http://mistic.heig-vd.ch/taillard/problemes.dir/ordonnancement.dir/ordonnancement.html adresinden ulaşılabilir. 2. grupta ise Carlier (1978), Heller (1960) ve Reeves (1995) tarafından sunulan ve http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/flowshop1.txt sayfasından (Mattfeld ve Vaessens, 2023) ulaşılabilen problemler yer almaktadır. Bu grupta boyutları (11x5) ile (100x10) arasında değişen farklı boyutlarda toplam 31 adet

problem bulunmaktadır. Bu çalışmada her iki grupta yer alan test problemleri ile de deneyler gerçekleştirilmiştir.

Akış tipi çizelgeleme problemlerinde yaygın olarak kullanılan performans ölçütlerinden biri de maksimum tamamlanma zamanıdır (makespan). Bu nedenle deneylerde; Eşitlik (5.1) ile hesaplanan en iyi maksimum tamamlanma zamanının, üst sınır değerlerinden ortalama bağıl sapma yüzdesi (relative percentage deviation), $\Delta_{ort,}$ yanıt değişkeni olarak kullanılmıştır.

$$\Delta_{ort} = \sum_{i=1}^{R} (\frac{(H_i - U_i).100}{U_i})/R \tag{5.1}$$

Burada H_i verilen bir örnek için test edilmiş herhangi bir sezgisel metot tarafından üretilen çözüm, U_i ise test problemleri için bilinen üst sınır değerleridir. Bu çalışmada Taillard'ın test problemlerinin üst sınır değerleri (Taillard, 2023)dan, 2. grupta yer alan problemlerininkiler (Ancau, 2012)dan alınmıştır. R ise birbirinden bağımsız olarak yapılan tekrar sayısıdır. Bu çalışmada yapılan deneylerin hepsi aksi belirtilmedikçe 20 tekrar ile gerçekleştirilmiştir. Bazı deneylerde metotların performanslarını karşılaştırmak için Δ_{ort}, değerine ek olarak Eşitlik (5.2) ve Eşitlik (5.3) ile hesaplanan Δ_{min} ve Δ_{max} değerleri de hesaplanmıştır.

$$\Delta_{min} = \min_{i=1:R} \left(\frac{(H_i - U_i).100}{U_i}\right) \tag{5.2}$$

$$\Delta_{min} = \max_{i=1:R} \left(\frac{(H_i - U_i).100}{U_i}\right) \tag{5.3}$$

Çalışma kapsamında gerçekleştirilen deneyler Intel (R) Core i7-8550 U@1.8 Ghz işlemci, 8 GB RAM ve Windows 8.1 Pro işletim sistemine sahip bilgisayar Matlab 7.8.0 (2009a) yazılımı kullanılarak gerçekleştirilmiştir. Bu çalışma kapsamında kodlanmadan sonuçları ilgili çalışmadan alınan metotların kodlandığı programlama dilleri ve kullanılan bilgisayarların özellikleri mümkün olduğunca ilgili başlık altında verilmiştir.

Başlangıç parametrelerinin metasezgisel algoritmaların performansı üzerinde önemli etkisi bulunmaktadır. Başlangıç parametreleri Ahrari ve diğ. (2009) ile Ahrari ve Atai

(2010) tarafından verilen tavsiyeler ve gerçekleştirilen basit ön testlerin sonucunda Tablo 5.1'de belirtildiği şekilde belirlenmiş olup aksi belirtilmediği sürece çalışmada bu parametreler kullanılmıştır. Tablo 5.1'de verilen değerler çalışmada çoğunlukla kullanılan değerdir, deneyler gereği faklı değerle kullanıldığında ilgili başlık altında belirtilmiştir.

Tablo 5.1. Çalışmada yaygın kullanılan EBPM başlangıç parametre değerleri

No	Parametre	Değer
1	El bombası sayısı (Ng)	nxm/20
2	Optimal arama yönünü belirlemekte kullanılacak el bombası sayısı (Nq-oay)	2
3	Bir bombanın fırlatacağı şarapnel sayısı (Nq)	40
4	Patlama yarıçapı ($R_{e\text{-}initial}$)	$2\sqrt{n}$
5	Ajan bölgesi yarıçapı ($R_{t\text{-}initial}$)	$\sqrt{n}$
6	İlk iterasyondaki R_t değerinin son iterasyondaki R_t değerine oranını (R_{rd})	100
7	Patlama yarıçapını güncellemede kullanılan sabit (m_{min})	0,2
8	Optimal arama yönün ağırlığını güncellemede kullanılan sabit(P_{sin})	0,1
9	Ajan Bölgesi Araştırma Yoğunluğu (P_{TS})	0,2
10	Durdurma kriter (İterasyon sayısı veya maksimum çalışma süresi)	-

Çalışmada durdurma kriteri olarak farklı maksimum çalışma süreleri kullanılmıştır. Maksimum çalışma süreleri, işlemci tarafından harcanan süre (CPU zamanı) şeklinde saniye biriminde belirlenmiştir.

5.1. Ajan Bölgesinin ve Optimal Arama Yönünün Etkisinin Değerlendirilmesi

EBPM'yi diğer metasezgisellerden ayıran özellik, el bombalarının ve ürettikleri şarapnellerin diğer bombalara ajan bölgesi yarıçapından daha fazla yaklaşmasına izin verilmemesidir.

Ajan bölgesi sayesinde arama uzayı homojen olarak taranmakta ve algoritmanın yerel optimum noktalarına takılması önlenmektedir. Fakat noktalar arasındaki mesafelerin hesaplanması her bir iterasyondaki işlem süresini uzatmaktadır. Bu nedenle ajan bölgesinin katkısının araştırılmasına ihtiyaç duyulmaktadır.

Ahrari ve diğerleri (2009, 2010) tarafından sunulan iki EBPM arasındaki temel fark optimal arama yönüdür. Bu özelliğin amaç fonksiyonun düzgün şekilde artığı veya azaldığı durumlarda metodun performansını artırdığı kabul edilmektedir. Fakat PATÇP'de katkısı bilinmemektedir.

Ajan bölgesi yarıçapı (R_t) ve optimal arama yönünü belirlemekte kullanılacak el bombası sayısı (Nq-oay) için Tablo 5.2'de verilen ikişer farklı seviye seçilerek metodun performansı üzerine etkileri tespit edilmeye çalışılmıştır. Ajan bölgesi yarıçapı için yüksek seviye$\sqrt{n}$ seçilirken; R_t =0 değeri ile algoritma çalışmadığı için düşük seviye olarak 10^{-10} gibi çok küçük bir değer seçilmiştir.

Tablo 5.2. Kullanılan parametre seviyeleri ve metot isimleri

Parametre	Metot/Seviye			
	EBPM1	EBPM2	EBPM3	EBPM4
Nq-oay	5	0	0	5
R_t	$\sqrt{n}$	$\sqrt{n}$	10^{-10}	10^{-10}

Durdurma kriteri olarak maksimum çalışma süresi = (nxmx0.1) CPU zamanı ve maksimum iterasyon sayısı=300 birlikte kullanılarak Tailard'ın ilk 6 problem grubu üzerinde (toplam 60 problem) 4 farklı deney yapılmıştır. Her bir problem için elde edilen makespan ve bağıl sapma yüzdelerine (Δ) ait tanımlayıcı istatistikler Ek-A'da verilmiştir. Her bir problem grubu için elde edilen ortalama bağıl sapma yüzdeleri (Δ_{ort}) Tablo 5.3'te özetlenmiştir.

Metotların karşılaştırılmasında kullanılabilecek istatistiksel yöntemler parametrik ve parametrik olmayan şeklinde iki sınıfa ayrılmaktadır. Parametrik yöntemlerin kullanabilmesi için verilerin normal dağılım göstermesi gerekmektedir. Bu nedenle sonuçların analizinde kullanılacak istatistiksel yönteme karar vermek için öncelikle verilerin normal dağılma uygunluğu test edilmiştir.

Tablo 5.3. Farklı R_t ve Nq-ory sevilerine ait ortalama bağıl sapma yüzdeleri (Δ_{ort})

Problem Grubu (nxm)	EBPM1	EBPM2	EBPM3	EBPM4
20×5	1,1995	1,5110	3,8212	3,4076
20×10	2,4414	3,3961	6,8882	6,7772
20×20	2,0248	3,0460	6,1134	5,8891
50×5	0,6648	1,2657	3,0004	2,7774
50×10	4,4445	7,9294	11,4259	10,8618
50×20	5,5707	10,5638	14,2459	13,8382
Ortalama	2,7243	4,6187	7,5825	7,2585

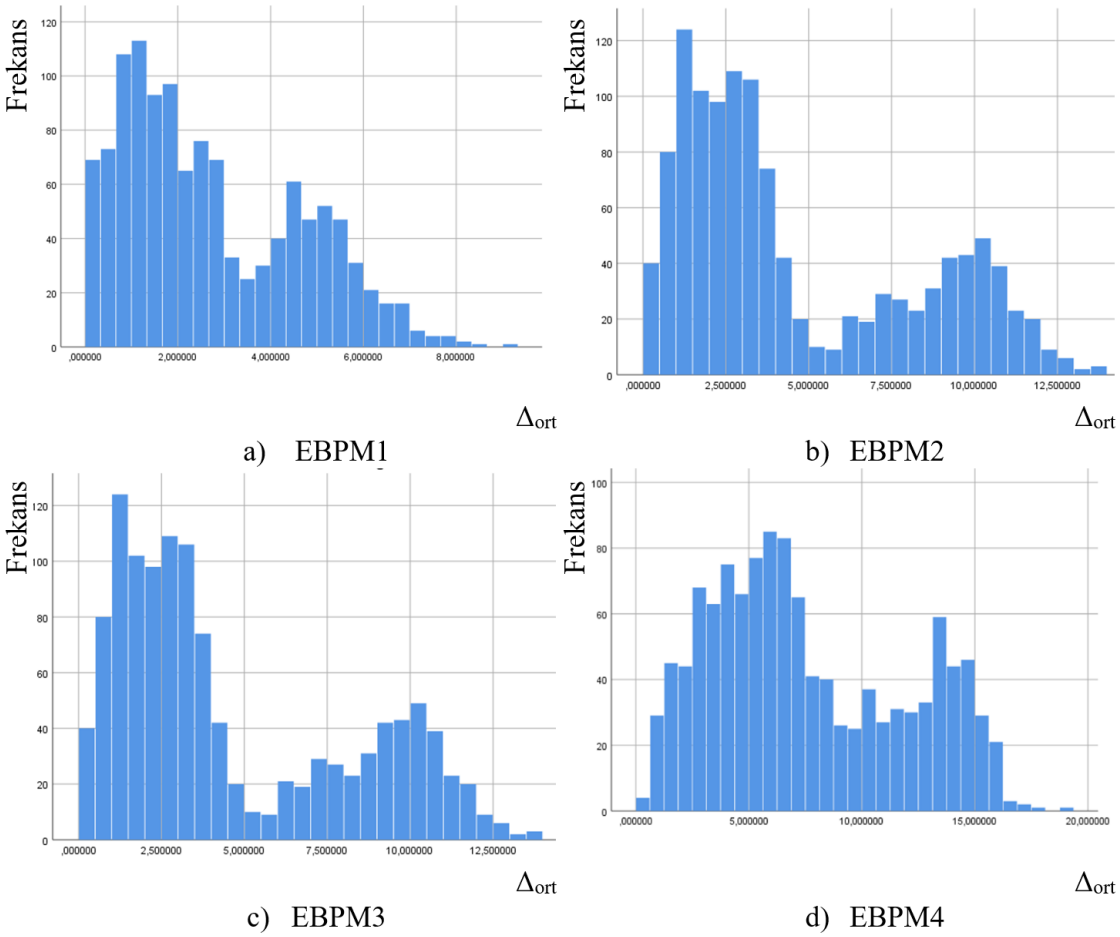
5.1.1. Normallik Testi

Verileri analiz etmede ilk olarak önemlilik/hipotez testlerinin varsayımlarının sağlanıp sağlanmadığı kontrol edilmelidir. Eğer veri seti dağılımsal varsayımlara sahip değilse, bu varsayımlar altında yapılacak olan tüm istatistiksel analizler geçersiz olacaktır ve çoğu zaman yanlış sonuçlar elde edildiği için yanlış yorumlar elde etmemize neden olacaktır. Bu nedenle istatistiksel analizler yöntemine karar vermek için veri setinin normal dağılıp dağılmadığı araştırılmalıdır. Normal dağılıma uygunluk tanımlayıcı, grafiksel ve istatistiksel yöntemlerle incelenebilir (Arslan ve diğ., 2020).

Normal dağılıma uygunluğun değerlendirilmesi konusunda bazı araştırmacıların Kolmogorov-Smirnov, Shapiro Wilk, Anderson-Darling vb. yöntemleri yeterli bulurken bazı araştırmacıların 3 aşamalı bir değerlendirme yaptığı görülmüştür. 3 aşamalı normallik değerlendirmesinde; ilk aşamada veriler şekilsel olarak incelenmekte, ikinci aşamada verilerin çarpıklık (skewness) ve basıklık (kurtosis) değerlerine bakılmakta, üçüncüde ise verilere Kolmogorov-Smirnov, Shapiro Wilk vb. testler uygulanmaktadır (Yücenur ve diğ., 2011).

Bu çalışmada verilerin normallik testi IBM SPSS 25 paket programı ile gerçekleştirilmiştir. Paket program tarafından üretilen normallik test raporu Ek-B'de verilmiştir.

İlk aşamada Şekil 5.1'de verilen ortalama bağıl sapma yüzdelerine (Δ_{ort}) ait histogram grafikleri incelenmiştir. Normal dağılam uygun histagramların çana bezer bir şekle sahip olmaları gerekmektedir. Şekil 5.1'de yer alan histogramların çana benzemediği, ikişer tepelerinin bulunduğu ve 4 histogramın da 4 normal dağılım sergilemediği görülmüştür.



Şekil 5.1. Metotların ortalama bağıl sapma yüzdelerine (Δ_{ort}) ait histogram grafikleri

İkinci aşamada veriler Tablo 5.4'te verilen çarpıklık ve basıklık değerleri açısından incelenmiştir. İdeal normal dağılımda çarpıklık ve basıklık değerleri sıfırdır. Fakat genellikle bu değerlerin sıfır olmadığı görüldüğünden uygulamada araştırmacılar tarafından sınır değerler belirlenmiştir. Çarpıklık ve basıklık değerleri için kabul görmüş en dar aralık [-1,1] olarak belirlenmiştir. Çarpıklık ve basıklık bakımından metotların Δ_{ort} değerlerinin normal dağılıma uyduğu görülmüştür.

Tablo 5.4. Metotların Δ_{ort} değerlerine ait çarpıklık ve basıklık istatistikleri

	EBPM1	EBPM2	EBPM3	EBPM4
Çarpıklık	0,612	0,737	0,421	0,418
Basıklık	-0,630	-0,855	-0,980	-0,962

Normallik testinin son aşamasında ise Kolmogorov-Smirnov ve Shapiro-Wilk normallik testleri uygulanmış olup test sonuçları Tablo 5.5'te metotların ortalama bağıl sapma yüzdelerine (Δ_{ort}) ve makespan ölçütleri için birlikte verilmiştir. Tabloda verilen değerler incelendiğinde p (sig.)<0.05 olduğundan tüm metotlardan elde edilen makespan ve Δ_{ort} sonuçlarının normal dağılıma uymadığı anlaşılmıştır. Bu sonuç Kolmogorov-Smirnov ve Shapiro-Wilk normallik testlerinin her ikisi için de geçerlidir.

Tablo 5.5. Kolmogorov-Smirnov ve Shapiro-Wilk testi sonuçları

Metot	Ölçüt	Kolmogorov-Smirnov			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
EBPM1	Makespan	,135	1200	,000	,927	1200	,000
	Δ_{ort}	,114	1200	,000	,936	1200	,000
EBPM2	Makespan	,132	1200	,000	,925	1200	,000
	Δ_{ort}	,181	1200	,000	,876	1200	,000
EBPM3	Makespan	,127	1200	,000	,927	1200	,000
	Δ_{ort}	,101	1200	,000	,943	1200	,000
EBPM4	Makespan	,124	1200	,000	,927	1200	,000
	Δ_{ort}	,095	1200	,000	,946	1200	,000

Verilerin normallik testinde 3 kriterden birinde testi geçtiği, ikisinde normal dağılıma uygun çıkmadığı görülmüştür. Verilerin normal dağılıma uymadığı sonucuna varılarak parametrik olmayan analiz yöntemlerinin kullanılmasına karar verilmiştir.

Bu nedenle metotlar arasında %5 anlamlılık seviyesinde istatistiksel olarak anlamlı bir fark olup olmadığı öncelikle parametrik olmayan bir yöntem olan Kruskal-Wallis Test ile incelemiştir. Daha sonra hangi gruplar arasında anlamlı fark olduğu ikili karşılaştırmalar ile Mann-Whitney U testi uygulanarak incelenmiştir. Bu amaçla yapılan 6 adet ikili karşılaştırmaya ait Mann-Whitney U test sonuçları Tablo 5.6'da özetlenmiştir.

Tablo 5.6. Mann-Whitney U test istatistikleri

	EBPM1-EBPM2	EBPM1-EBPM3	EBPM1-EBPM4	EBPM2-EBPM3	EBPM2-EBPM4	EBPM3-EBPM4
Mann-Whitney U	513273,000	214778,000	240215,500	415191,000	439306,500	687692,000
Wilcoxon W	1233873,000	935378,000	960815,500	1135791,000	1159906,500	1408292,000
Z	-12,179	-29,764	-28,266	-17,957	-16,537	-1,903
Asymp. Sig. (2-tailed)	,000	,000	,000	,000	,000	,057

Tablo 5.6 incelendiğinde sadece EBPM3 ile EBPM4 arasında %5 önem düzeyinde anlamlı bir fark olmadığı görülmektedir. %10 önem düzeyinde EBPM3 ile EBPM4 arasında dahil tüm ikili karşılaştırmalarda anlamlı fark olduğu açıktır. Ajan bölgesi yarıçapı seviyesinin değiştiği tüm karşılaştırmalarda (EBPM1- EBPM3, EBPM1- EBPM4, EBPM2- EBPM3 ve EBPM2- EBPM4) istatistiksel olarak anlamlı fark bulunmaktadır ve R_t'nin yüksek seviyedeki değerlerinde ($R_t=\sqrt{n}$) elde edilen ortalama bağıl sapma yüzdeleri daha küçüktür. Bu nedenle ajan bölgesi yarıçapının (R_t) EBPM'nin performansı üzerinde anlamlı bir etkisinin olduğu anlaşılmaktadır.

Elde edilen optimal arama yönünün etkisi için ise kesin bir sonuca varılmamaktadır. R_t sabit tutulurken Nq-oay seviyesinin değiştirildiği EBPM1- EBPM2 karşılaştırmasında %5 önem düzeyinde anlamlı fark varken EBPM3- EBPM4 karşılaştırmasında anlamlı fark görülmemiştir.

5.2. EBPM'nin Doğrulanması

EBPM'nin makul sürede kabul edilebilir sonuçlar üretebildiğini doğrulamak için Taillard'ın kıyaslama problemleri maksimum tamamlanma zamanı ölçütü kullanılarak çözülmüş ve sonuçlar literatürdeki çalışmalarla karşılaştırılmıştır. Bu yolla EBPM'nin PATÇP'lerinde kullanılabilirliği değerlendirilmiştir.

EBPM'de kullanılan başlangıç parametreleri Tablo5.1'de belirtilmiş olup durdurma kriteri olarak maksimum çalışma süresi = (nxmx0.1) CPU zamanı ve maksimum iterasyon sayısı=300 birlikte kullanılmıştır. Durdurma kriteri literatürdeki birçok çalışmaya (Tasgetiren diğ., 2007; Dasgupta ve Das, 2015; Li ve Yin, 2012; Liu ve Liu, 2013) benzer veya daha kısa belirlenmiştir. Böylelikle çözüm için harcanan sürenin literatürde kabul görmüş süreleri aşmaması sağlanmıştır.

Literatürde PATÇP için önerilmiş hızlı ve başarılı sonuç veren birçok yöntemin olduğu bilinmektedir. Fakat bunların birçoğu özellikle güncel olanları melez algoritmalar veya NEH vb. yöntemlerle elde edilmiş başlangıç çözümleri kullanan yöntemlerdir. Orijinal EBPM, rassal olarak üretilen başlangıç çözümleri kullandığından adil bir karşılaştırma için benzer çalışmalar araştırılmıştır ve karşılaştırma amacıyla bu kritere uyan; Taşgetiren diğ. (2007) tarafından önerilen Genetik Algoritma ile Parçacık Sürü Optimizasyonu ve Yağmahan ve Yenisey (2006) tarafından önerilen Karınca Kolonisi Optimizasyonu seçilmiştir:

Karşılaştırma için seçilen yöntemler tekrar kodlanmamış; yukarıda belirtilen kaynaklardan alınmıştır. Önerilen EBPM ve seçilen yöntemlere ait ortalama bağıl sapma yüzdeleri (Δ_{ort}), minimum bağıl sapma yüzdeleri (Δ_{min}), maksimum bağıl sapma yüzdeleri (Δ_{max}) ve standart sapmaları (Δ_{std}) Tablo 5.7'de verilmiştir.

Tablo 5.7. Problem gruplarına ve yöntemlere göre Δ_{ort}, Δ_{std}, Δ_{min} ve Δ_{max} değerleri

Problem Grubu (nxm)	GA				PSO				EBPM				KKO			
	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}
20 x 5	3,14	1,81	0,52	7,49	1,75	1,23	0,00	6,44	**1,34**	0,58	0,00	3,52	2,63	5.91	1,81	3,33
20 x 10	5,38	1,71	1,93	10,02	3,25	1,18	0,61	7,05	**2,72**	0,89	0,87	5,64	4,74	7,92	3.88	5.47
20 x 20	4,18	1,3	1,07	8,05	2,82	1,15	0,44	5,46	**2,19**	0,74	0,41	4,34	4,46	11,07	3,58	5,23
50 x 5	1,67	0,79	0,25	3,36	1,14	0,7	0,03	3,53	**0,68**	0,49	0,00	1,94				
50 x 10	6,06	1,52	3,3	10,25	5,29	1,28	2,3	8,49	**4,73**	1,23	1,58	7,64				
50 x 20	7,47	1,09	4,75	10,08	7,21	1,31	4,55	10,19	**5,98**	1,17	3,70	9,72				
100 x 5	0,85	0,4	0,04	2,12	0,63	0,35	0,04	1,49	**0,52**	0,33	0,00	1,78				
100 x 10	3,13	0,94	1,43	6,11	3,27	1,04	0,74	5,98	**2,80**	0,94	0,62	5,45				
100 x 20	**7,38**	0,97	4,98	11,06	8,25	1,07	5,15	11,45	8,16	0,94	5,94	11,68				
200 x10	**2,05**	0,45	1,11	3,03	2,47	0,71	0,71	4,73	5,51	1,26	2,09	8,28				
200 x 20	**6,08**	0,8	4,48	8,28	8,05	0,85	6,02	10,01	14,20	0,68	12,46	15,63				

Tablo 5.7'de yer alan 11 problem grubunun ilk 8'inde EBPM diğer yöntemlerden daha iyi Δ_{ort} değerine ulaşmıştır. Büyük problem grupları olan son 3 problem grubunda ise GA daha iyi Δ_{ort} değerlerine sahiptir. EBPM son iki problem grubunda GA ve PSO yöntemlerinin gerisinde 9. problem grubunda ise GA ile PSO arasında bir değere sahiptir. KKO için sadece ilk 3 problem grubuna ait değerler bulunmakta olup bu problem gruplarında EBPM, KKO'ya göre daha iyi Δ_{ort} değerlerine sahiptir.

Tablo 5.7'deki veriler incelendiğinde EBPM'nin performansının küçük ve orta büyüklükteki problem gruplarında, büyük problem gruplardan göreceli olarak daha iyi olduğu anlaşılmaktadır. Bu durum başlangıç parametrelerinin büyük problem gruplarına uygun belirlenememesinden veya problem boyutu arttıkça ajan bölgelerine olan uzaklıkların hesaplanması için ihtiyaç duyulan zamanın daha fazla artmasından kaynaklanabilir.

5.3. EBPM'nin Performansının DGA ile Karşılaştırılması

Kullanılan bilgisayarın özellikleri, programlama dili ve kodların yapısı metasezgisel yöntemlerin hızını önemli derecede etkilemektedir. Bu nedenle adil bir değerlendirme için yöntemlerin çalıştırıldığı ortamların mümkün olduğunca aynı olması gerekmektedir.

Literatürde bu sorunu aşmak için yöntemlerin çözüme ulaşmak için harcadığı süreler yerine ziyaret edilen nokta sayıları veya iterasyon sayılarının ölçüldüğü çalışmalar da bulunmaktadır. Fakat ziyaret edilecek noktanın belirlenmesi için veya bir iterasyonda gerçekleştirilen işlemlerin sayısı çok farklı olabilir. Ayrıca klasik optimizasyon teknikleri yerine sezgisel yöntemlerin tercih edilmesinin en önemli nedeni çözüme ulaşmak için ihtiyaç duyulan sürenin kısalığıdır. Bu nedenle yöntemlerin performansları karşılaştırılırken durdurma kriteri olarak CPU zamanı cinsinden maksimum çalışma suresinin kullanılmasının daha adil olduğu düşünülmektedir.

Metasezgisel yöntemlerin çalıştırıldığı şartların farklılığından kaynaklanan bu sorunu aşmak için sıklıkla başvurulan yaklaşımlarda biri karşılaştırma için kullanılan metotların da yeniden kodlanarak aynı bilgisayarda elde edilen sonuçların karşılaştırılmasıdır. Kodlama için ek zaman gerektirmesi, kodlanacak yöntemler hakkında da detaylı bilgiye ihtiyaç duyulması, rassallık ve bazı parametrelerdeki belirsizlikler nedeniyle orijinal çalışmayla aynı sonuçların elde edilememesi gibi dezavantajlarına rağmen bu yaklaşım en çok kabul gören yaklaşımdır.

Bu çalışmada da aynı yaklaşım benimsenerek sağlıklı bir karşılaştırma için Diferansiyel Gelişim Algoritması Matlab ile kodlanmıştır. DGA, Price ve Storn tarafından 1995 yılında sürekli optimizasyon problemleri için geliştirilmiş ve literatürde bir çok çalışmaya konu olmuştur. Zamanla kesikli problemlere uyarlanmış versiyonları dahil farklı DGA

uygulamaları bulunmaktadır. Bu çalışmada Keskintürk (2006) ve Heris (2015)den faydalanılarak DGA kodlanmıştır. Parametreler optimizasyon çalışması yapılmadan aşağıdaki gibi seçilmiştir:

- NP: Popülasyon büyüklüğü = 50;
- D: Değişken sayısı (gen sayısı) = İş sayısı (n),
- CR: Çaprazlama oranı [0,1, 1,0] =0,2,
- F: Ölçekleme faktörü = [0,2-0,8] aralığında rassal sayı,

İki grupta yer alan test problemlerinin tamamı optimal arama yönlü EBPM ve DGA ile çözülmüştür. Birinci grubu oluşturan Taillard'ın test problemlerine ait bağıl yüzde sapma değerleri Tablo 5.8'de, 2. grup problemlere ait sonuçlar 5.9'da özetlenmiştir. Her iki grupta da başlangıç popülasyonları rassal olarak oluşturulmuş ve durdurma kriteri olarak nxmx0,1 CPU zamanı kullanılmıştır.

Tablo 5.8. Taillard'ın test (1. grup) problemlerine ait sonuçlar

Problem Grubu (nxm)	EBPM				DGA			
	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}
20 x 5	1,34	0,58	0,00	3,52	**0,83**	0,52	0,00	2,03
20 x 10	2,72	0,89	0,87	5,64	**1,41**	0,59	0,00	3,05
20 x 20	2,19	0,74	0,41	4,34	**0,90**	0,51	0,18	2,51
50 x 5	**0,68**	0,49	0,00	1,94	1,36	0,63	0,00	2,91
50 x 10	**4,73**	1,23	1,58	7,64	6,89	1,23	4,61	9,69
50 x 20	**5,98**	1,17	3,70	9,72	9,04	0,50	7,43	10,22
100 x 5	**0,52**	0,33	0,00	1,78	1,12	0,38	0,58	1,88
100 x 10	**2,80**	0,94	0,62	5,45	5,46	1,12	3,12	7,90
100 x 20	**8,16**	0,94	5,94	11,68	11,14	1,08	8,72	13,27
200 x10	5,51	1,26	2,09	8,28	**5,08**	1,26	1,97	7,07
200 x 20	14,20	0,68	12,46	15,63	**12,00**	0,58	10,42	13,25
500x20	11,67	0,46	10,29	12,87	**9,43**	0,41	8,38	10,25
Genel Toplam	5,04	4,27	0,00	15.63	5,39	4,20	0,00	13,77

Tablo 5.9. Diğer test problemlerine (2. grup) ait sonuçlar

Problem (nxm)	EBPM				DGA			
	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}
car1 (11x5)	**0,00**	0,00	0,00	0,00	**0,00**	0,00	0,00	0,00
car2 (13x4)	0,15	0,66	0,00	2,93	**0,00**	0,00	0,00	0,00
car3 (12x5)	0,76	0,54	0,00	1,19	**0,00**	0,00	0,00	0,00
car4 (14x4)	**0,00**	0,00	0,00	0,00	**0,00**	0,00	0,00	0,00
car5 (10x6)	0,54	0,52	0,00	1,31	**0,14**	0,33	0,00	1,31
car6 (8x9)	**0,08**	0,24	0,00	0,76	0,34	0,39	0,00	0,76
car7 (7x7)	**0,00**	0,00	0,00	0,00	**0,00**	0,00	0,00	0,00
car8 (8x8)	**0,00**	0,00	0,00	0,00	**0,00**	0,00	0,00	0,00
hel1 (100x10)	**2,06**	0,86	0,58	3,49	3,01	0,20	2,52	3,29
hel2 (20x10)	2,39	0,79	0,74	3,68	**0,77**	0,16	0,74	1,47
reC01 (20x5)	1,19	1,48	0,16	6,34	**0,16**	0,00	0,16	0,16
reC03 (20x5)	0,59	0,65	0,18	2,98	**0,15**	0,38	0,00	1,71
reC05 (20x5)	0,66	0,72	0,24	2,50	**0,24**	0,00	0,24	0,24
reC07 (20x10)	1,09	0,26	0,00	1,15	**0,56**	0,46	0,00	1,15
reC09 (20x10)	2,24	0,74	0,65	3,32	**0,30**	0,43	0,00	1,63
reC11 (20x10)	1,97	1,00	0,84	4,26	**0,42**	0,51	0,00	1,61
reC13 (20x15)	2,20	0,94	0,47	4,56	**0,47**	0,26	0,26	1,04
reC15 (20x15)	2,35	0,55	1,18	3,54	**1,26**	0,85	0,56	3,74
reC17 (20x15)	3,52	0,94	1,52	5,15	**1,05**	0,42	0,37	1,68
reC19 (30x10)	**3,55**	1,06	1,62	5,06	4,31	0,29	3,63	4,78
reC21 (30x10)	2,91	0,90	1,64	5,01	**2,77**	0,33	1,98	3,42
reC23 (30x10)	3,58	0,92	1,94	5,57	**3,08**	0,15	2,78	3,38
reC25 (30x15)	**4,26**	0,91	2,31	5,73	4,35	0,34	3,66	4,85
reC27 (30x15)	**3,11**	0,82	1,98	5,14	3,42	0,36	2,82	3,92
reC29 (30x15)	**4,29**	1,07	2,71	6,12	5,37	0,52	3,76	6,12
reC31 (50x10)	**5,22**	1,05	3,51	7,13	8,11	0,47	6,83	8,87
reC33 (50x10)	**1,90**	0,47	0,96	2,79	3,95	0,24	3,37	4,37
reC35 (50x10)	**0,96**	0,87	0,00	2,90	0,84	0,22	0,34	1,10
reC37 (75x20)	**7,85**	0,67	6,65	9,49	11,54	0,38	10,75	12,18
reC39 (75x20)	**6,49**	0,82	4,70	7,86	9,17	0,43	8,14	9,87
reC41 (75x20)	**8,07**	0,84	6,88	10,16	11,77	0,31	11,19	12,32

Tablo 5.8'de verilen ortalama bağıl sapma yüzdeleri (Δ_{ort}) bakımından EBPM ve DGA karşılaştırıldığında, 12 problem grubunun altısında EBPM'nin, altısında ise DGA'nın daha iyi sonuçlara ulaştığı görülmüştür. EBPM; orta büyüklükteki problemlerde DGA'dan daha Δ_{ort} değerlerine sahipken küçük ve büyük boyutlu problemlerde DGA'nın gerisinde kalmıştır.

Tablo 5.9'da verilen 2. Grupta yer alan toplam 31 test problemine ait ortalama bağıl sapma yüzdeleri (Δ_{ort}) incelendiğinde EBPM'nin; 4 problemde DGA ile aynı (0,00), 12 problemde DGA'dan daha iyi, 13 problemde ise DGA'dan daha kötü değerlere sahip olduğu görülmektedir.

Her iki gruba ait sonuçlan incelediğinde ilk bakışta metotlar arasında baskınlık görülmemiştir. Bu nedenle metotlar arasında %5 anlamlılık seviyesinde istatistiksel olarak anlamlı bir fark olup olmadığı, iki gruptaki toplam 151 probleme ait Δ_{ort} değerlerine Mann-Whitney U testi uygulanarak incelenmiştir. Mann-Whitney U test sonuçları Tablo 5.10'da özetlenmiştir.

Tablo 5.10. EBPM ile DGA karşılaştırması Mann-Whitney U test istatistikleri

Metot	N	Sıra Ortalaması	Sıra Toplamı	Mann-Whitney U	Z	P
EBPM	3020	2999,68	9059018,50	4497308,500	-,928	,353
DGA	3020	3041,33	9184801,50			

Tablo 5.10 incelendiğinde EBPM ile DGA arasında %5 önem düzeyinde anlamlı bir fark olmadığı görülmektedir. Bu sonuç EBPM'nin; tanımlanan şartlar altında literatürde kabul görmüş DGA ile benzer performansa sahip olduğu, PATÇP'nin makespan ölçütüne göre çözümünde DGA kadar başarılı olduğu şeklinde yorumlanabilir.

5.4. NEH ile Üretilen Başlangıç Çözümlerin Katkısı

Başlangıç çözümlerin nasıl belirlendiğinin metasezgisel yöntemlerin performansı üzerinde önemli etkisi bulunmaktadır. Literatürde yer alana güncel çalışmalarda genellikle başlangıç çözümlerin bir kısmı bir veya daha çok sezgisel yöntem kullanılarak belirlenmektedir. Ayrıca literatürde başlangıç çözümüm belirlenmesinde farklı

sezgisellerin kullanımının çözüm kalitesine etkisini araştıran çok sayıda çalışma örnekleri (Kaya ve diğ., 2020; Madenoğlu, 2020) de bulunmaktadır. Bu çalışmalarda başlangıç çözümün rassal olarak oluşturulduğu veya farklı sezgisellerle oluşturulduğu durumlarda ilgili metasezgisel yöntemle elde edilen çözümler karşılaştırılmaktadır. Başlangıç çözümlerin oluşturulmasında en çok tercih edilen sezgisellerden biri de NEH ve geliştirilmiş versiyonlarıdır (Madenoğlu, 2020; Kızılay ve Çil, 2021).

NEH sezgiseli ile üretilen sonuçlar iş sırası şeklinde olduğundan sürekli konum vektörleri ile çalışan EBPM ve DGA'da doğrudan kullanılamamaktadır. İş sıralarının konum vektörlerine dönüştürülmesinde kullanılacak yöntemin de çözüm kalitesini etkileyeceği bilinmektedir. Bu çalışmada bu amaçla Liu ve diğerlerinden (2007, 2014) uyarlanan Eşitlik (5.4) kullanılmıştır.

$$X_{NEH} = X_{min,j} + \left(X_{max,j} - X_{min,j}\right).(S_{NEH,J} - 1)/n \ , \quad j = 1,2,\dots,n. \tag{5.2}$$

Çalışmanın bu bölümüne kadar gerçekleştirilen deneylerin hepsinde başlangıç çözümler rassal olarak belirlenmiştir. Bu bölümünde Taillard (1990) tarafından sunulan hızlandırılmış NEH sezgiseli kullanarak birer başlangıç çözüm EBPM ve DGA yöntemlerinin başlangıç çözümleri arasına dahiledilerek metotların performanslarına etkileri araştırılmıştır. Başlangıç çözümler dışında bir değişiklik yapılmadan önceki başlıkta tanımlanan EBPM ve DGA kullanılarak Taillard'ın test problemleri çözülmüştür. Elde edilen sonuçlar başlangıç çözümler arasında NEH ile üretilen çözümlerin de olduğunu ifade etmek için "NEH" alt indisi kullanılarak Tablo 5.11'de özet şekilde verilmiştir. Tablo 5.11'de bulunan "Fark" sütunu, ortalama bağıl yüzde sapma değerlerinin Tablo 5.8'deki değerlere göre ne kadar değiştiğini ifade etmektedir. $EBPM_{NEH}$'de Tablo 5.1'deki, DGA_{NEH} aşağıdaki başlangıç parametreleri ile çalıştırılmıştır:

- NP: Popülasyon büyüklüğü= nx10;
- D: Değişken sayısı (gen sayısı) = İş sayısı (n),
- CR: Çaprazlama oranı [0.1,1.0] =0.9,
- F: Ölçekleme faktörü = [0.2-0.8] aralığında rassal sayı,
- Durdurma kriteri olarak maksimum süresi =nxmx0.1 CPU zamanı.

Tablo 5.11. $EBPM_{NEH}$ ve DGA_{NEH} ile Taillard'ın problemlerinde elde edilen sonuçlar

Problem Grubu (nxm)	$EBPM_{NEH}$					DGA_{NEH}				
	Δ_{ort}	Fark	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Fark	Δ_{std}	Δ_{min}	Δ_{max}
20 x 5	1,19	0,15	0,63	0,00	3,43	**0,76**	0,07	0,48	0,00	2,68
20 x 10	2,57	0,15	0,72	0,67	3,97	**1,45**	-0,04	0,69	0,20	4,72
20 x 20	2,16	0,03	0,68	0,37	4,18	**0,88**	0,02	0,49	0,09	2,82
50 x 5	**0,37**	0,31	0,35	0,00	1,13	0,69	0,67	0,44	0,00	1,42
50 x 10	**3,83**	0,90	1,17	1,19	5,98	5,03	1,86	1,07	2,83	6,26
50 x 20	**5,29**	0,69	0,69	3,53	7,28	6,66	2,38	0,88	5,58	8,60
100 x 5	**0,31**	0,21	0,24	0,00	0,75	0,47	0,65	0,27	0,08	0,91
100 x 10	**1,50**	1,30	0,60	0,15	3,20	2,21	3,25	0,78	1,25	3,88
100 x 20	**4,92**	3,24	0,78	2,75	6,36	5,34	5,80	0,75	3,78	6,41
200 x10	**1,26**	4,25	0,40	0,74	2,25	1,26	3,82	0,40	0,74	2,25
200 x 20	**4,41**	9,79	0,54	3,56	5,15	**4,41**	7,59	0,54	3,56	5,15
500x20	**2,07**	9,60	0,36	1,49	2,68	**2,07**	7,36	0,36	1,49	2,68
Genel Toplam	**2,49**	2,55	1,78	0,00	7,28	2,60	2,79	2,16	0,00	6,41

Tablo incelendiğinde NEH ile üretilen bir başlangıç çözümün kullanılması ile EBPM'nin çözüm kalitesinin, problem gruplarına göre %0.03 ile %9.79 arasında değişmekle birlikte ortalama %2,55 arttığı görülmektedir. İstisnaları bulunmakla birlikte, genelde problem boyutu büyüdükçe artış oranının da büyüdüğü gözlenmektedir. Benzer durum DGA için de geçerlidir. Fakat genel toplamda iki yöntemde de hemen hemen aynı oranda ilerleme sağlanmış olmasına rağmen daha iyi sonuca ulaşılan problem grubu sayısında önemli değişiklik gözlemlenmiştir. Ortalama bağıl sapma yüzdeleri (Δ_{ort}) bakımından EBPM ve DGA karşılaştırıldığında; Tablo 5.8'de 6 grupta EBPM'nin, 6 grupta ise DGA'nın daha iyi sonuçlara ulaştığı, Tablo 5.11'de ise 7 grupta $EBPM_{NEH}$'nin, 3 grupta DGA_{NEH}'nin daha iyi sonuçlara ulaştığı, 2 grupta ise benzer sonuçlara ulaşıldığı görülmüştür.

Ayrıca Tablo 5.12'de NEH, $EBPM_{NEH}$ ve DGA_{NEH}'nin Ortalama bağıl sapma yüzdeleri (Δ_{ort}) ve $EBPM_{NEH}$ ve DGA_{NEH}'nin NEH ile üretilen sonucu ne kadar geliştirdikleri verilmiştir. Tablo incelendiğinde $EBPM_{NEH}$ ile 12 problem grubunun 11'inde NEH ile üretilen çözümlerden daha iyi sonuçlara ulaşılmıştır. En büyük problem grubunda ise NEH ile üretilen çözüm geliştirilememiştir. DGA ise 8 problem grubunda gelişim sağlamış büyük boyutlu problemlerden oluşan son 4 grupta NEH ile üretilen çözümler

geliştirilememiştir. Bu durum ilk bakışta Optimal arama yönlü EBPM'nin mevcut çözümleri geliştirmekte DGA'dan daha iyi olduğu şeklinde yorumlanabilir. Fakat iki yöntemde de birer başlangıç çözüm NEH ile üretilmiş olup bunların başlangıç çözümler içindeki oranları farklıdır. Bu nedenle bu konuda daha doğru bir sonuca varmak için ilerleyen çalışmalarda farklı ve eşit oranda belirlenmiş başlangıç çözümlerle daha detaylı bir karşılaştırma yapılabilir.

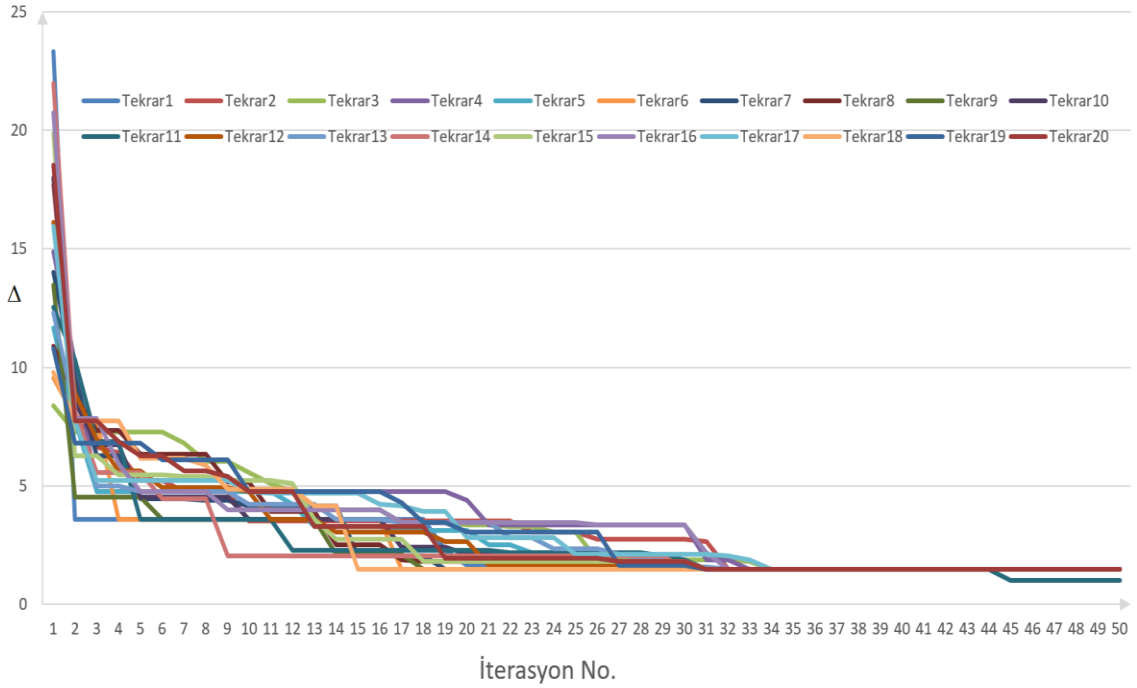
Tablo 5.12. NEH, $EBPM_{NEH}$ ve DGA_{NEH}'nin ortalama bağıl sapma yüzdeleri (Δ_{ort})

Problem Grubu (nxm)	NEH	$EBPM_{NEH}$	NEH- $EBPM_{NEH}$	DGA_{NEH}	NEH- DGA_{NEH}
	Δ_{ort}	Δ_{ort}	Δ_{ort}	Δ_{ort}	Δ_{ort}
20 x 5	3,2585	1,1863	2,0722	**0,7565**	2,5020
20 x 10	4,6011	2,5682	2,0330	**1,4475**	3,1536
20 x 20	3,7309	2,1577	1,5732	**0,8800**	2,8509
50 x 5	0,7272	**0,3685**	0,3587	0,6903	0,0369
50 x 10	5,0729	**3,8336**	1,2393	5,0261	0,0468
50 x 20	6,6596	**5,2905**	1,3691	6,6593	0,0003
100 x 5	0,5272	**0,3149**	0,2123	0,4661	0,0611
100 x 10	2,2150	**1,4974**	0,7175	2,2146	0,0003
100 x 20	5,3446	**4,9218**	0,4228	5,3446	0,0000
200 x10	1,2579	**1,2572**	0,0007	1,2579	0,0000
200 x 20	4,4076	**4,4057**	0,0019	4,4076	0,0000
500x20	2,0661	2,0661	0,0000	2,0661	0,0000
Genel Toplam	3,3224	**2,4890**	0,8334	2,6014	0.7210

5.5. EBPM'nin İterasyonlar Süresince Yakınsamasının Grafiksel Analizi

"Optimal Arama Yönlü EBPM'nin Grafiksel Analizi" başlığı altında örnek bir sürekli fonksiyon üzerinde EBPM'nin optimum noktaya yakınsama durumu analiz edilmişti. Bu başlık altında ise EBPM'nin PATÇP'de iterasyonlar boyunca optimum noktaya ne kadar yaklaştığı incelenmiştir. Bu amaçla Taillard'ın test problemleri arasından seçilen bazı problemler optimal orama yönlü EBPM ile çözülerek her bir iterasyonda ulaşılan bağıl sapma yüzdeleri (Δ) kaydedilmiştir. Bu aşamada Tablo 5.1'de verilen başlangıç parametreleri kullanılmış ve tüm başlangıç çözümler rassal olarak oluşturulmuştur. Örnek

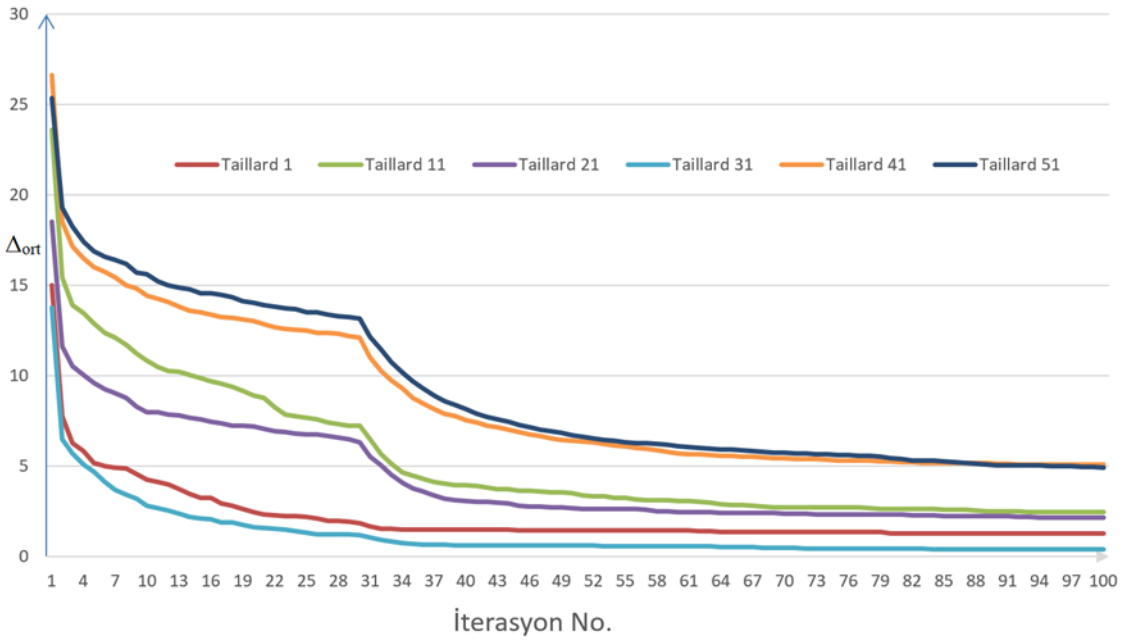
olarak Şekil 5.2'de Taillard'ın 1. probleminim çözümünde ilk 50 iterasyon boyunca elde edilen bağıl sapma yüzdeleri (Δ), Şekil 5.3'te Taillard'ın 1., 11., 21., 41. ve 51 problemlerinin çözümünde ilk 100 iterasyon boyunca elde edilen ortalama bağıl sapma yüzdeleri (Δ_{ort}) gösterilmiştir.



Şekil 5.2. Taillard'ın 1. Probleminim çözümünde ilk 50 iterasyon boyunca elde edilen bağıl sapma yüzdeleri (Δ)

Şekil 5.2 incelendiğinde tüm tekrarlarda ilk beş iterasyonda bağıl sapma yüzdelerinin hızlı bir şekilde düştüğü, daha sonra düşüşün hızının giderek azaldığı görülmektedir. 14. tekrarda 9. iterasyondan sonra çok az bir düşüş gerçekleşmiş bağıl sapma yüzdesi neredeyse yatay bir çizgi halinde devam etmiştir. 35. iterasyondan sonra tüm tekrarlara ait bağıl sapma yüzdelerinin neredeyse tek bir çizgi halinde devam ettiği görülmektedir. Bağıl sapma yüzdelerindeki düşüş 35. iterasyondan sonra çok az da olsa devam etmiştir. Örneğin 11. tekrarda, 45. iterasyondaki iyileşme gözle görülebilmektedir. Grafikte yer almamakla birlikte 11. tekrarda, 71., 18. tekrarda 80., 16. tekrarda 131 ve 10. tekrarda 132. iterasyonda optimum sonuca ulaşılmıştır. Diğer tekrarlarda durdurma kriterine takılarak algoritma optimum sonuca ulaşılmadan durdurulmuştur.

Şekil 5.3'te Şekil 5.2'den farklı olarak, bir probleme ait tekrar çalışmalarda elde edilen bağıl sapma yüzdeleri (Δ) yerine; değişik boyutlardaki 5 problemde 20'şer tekrar çalıştırma sonucunda elde edilen ortalama bağıl sapma yüzdeleri (Δ_{ort}) verilmiştir. Şekil 5.3'te farklı boyuttaki problemlerde de ilk iterasyonlarda; özellikle ilk 10 iterasyonda optimum sonuca hızlı bir şekilde yaklaşıldığı, yakınsama hızının ilerleyen iterasyonlarda düştüğü ve bir süre sonra neredeyse hiç ilerleme sağlanamadığı görülmektedir.



Şekil 5.3. Taillard'ın 1.,11.,21.41.ve 51. problemlerinin çözümünde ilk 100 iterasyon boyunca elde edilen ortalama bağıl sapma yüzdeleri (Δ_{ort})

5.6. Melez DGA$_{EBPM}$ Yöntemi

EBPM ile ilgili literatür çalışmalarından da metodun optimal sonuca ulaşma, özellikle yerel optimum noktalardan kaçınma konusunda başarılı olduğu fakat popüler diğer metotlara göre daha yavaş olduğu; bu nedenle güncel çalışmalarda metodun genellikle diğer metotlarla melez olarak kullanıldığı bilinmektedir.

EBPM'nin DGA ile melez kullanımının hız konusundaki dezavantajı azaltacağı düşünülmektedir. Bu nedenle geliştirilen melez algoritmada, çözüm uzayını homojen bir şekilde tarama başarısı nedeniyle başlangıç çözümlerin EBPM ile üretilmesi ve sonrasında çözümlerin DGA ile geliştirilmesi öngörülmüştür. Bu amaçla başlangıç

çözümlerin tamamı rassal olarak üretilen EBPM 10 iterasyon çalıştırılmış, üretilen el bombaları DGA'da başlangıç popülasyonuna eklenmiştir. El bombaları (nxm/20 adet) dışındaki başlangıç çözümler rassal olarak üretilmiştir. Bu şekilde oluşturulan melez DGA_{EBPM} algoritmasında kullanılan başlangıç parametreleri şunlardır:

- NP: Popülasyon büyüklüğü= nx10;
- D : Değişken sayısı (gen sayısı) = İş sayısı (n),
- CR : Çaprazlama oranı [0.1,1.0] =0.9,
- F : Ölçekleme faktörü = [0.2-0.8] aralığında rassal sayı,
- Durdurma kriteri olarak maksimum süresi =nxmx0.1 CPU zamanı.

Oluşturulan melez DGA_{EBPM} ile her iki gruptaki problemlerin tamamı çözülmüştür. Tablo 5.13'te DGA_{EBPM} ile Taillard'ın test problemlerinde elde edilen sonuçlar, Tablo 5.14'te ise 2. grup test problemlerinde elde edilen Δ_{ort} değerleri DGA, DGA_{NEH} metotları ile birlikte verilmiştir.

Tablo 5.13. DGA, DGA_{NEH} ve DGA_{EBPM} ile Taillard'ın problemlerinde elde edilen sonuçlar

Problem Grubu (nxm)	DGA				DGA_{NEH}					DGA_{EBPM}				
	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Fark	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Fark	Δ_{std}	Δ_{min}	Δ_{max}
20 x 5	0,83	0,52	0,00	2,03	**0,76**	0,07	0,48	0,00	2,68	0,84	-0,01	0,54	0,00	2,36
20 x 10	1,41	0,59	0,00	3,05	1,45	-0,04	0,69	0,20	4,72	**1,33**	0,08	0,64	0,13	3,66
20 x 20	0,90	0,51	0,18	2,51	**0,88**	0,02	0,49	0,09	2,82	0,91	-0,01	0,50	0,00	2,42
50 x 5	1,36	0,63	0,00	2,91	**0,69**	0,67	0,44	0,00	1,42	1,15	0,21	0,57	0,00	2,51
50 x 10	6,89	1,23	4,61	9,69	**5,03**	1,86	1,07	2,83	6,26	6,00	0,89	1,27	3,40	12,47
50 x 20	9,04	0,50	7,43	10,22	**6,66**	2,38	0,88	5,58	8,60	7,90	1,14	0,60	6,47	9,17
100 x 5	1,12	0,38	0,58	1,88	**0,47**	0,65	0,27	0,08	0,91	0,94	0,18	0,33	0,32	1,79
100 x 10	5,46	1,12	3,12	7,90	**2,21**	3,25	0,78	1,25	3,88	4,12	1,34	1,01	2,45	6,68
100 x 20	11,14	1,08	8,72	13,27	**5,34**	5,80	0,75	3,78	6,41	9,39	1,75	1,02	7,11	11,46
200 x10	5,08	1,26	1,97	7,07	**1,26**	3,82	0,40	0,74	2,25	3,63	1,45	1,11	1,54	7,71
200 x 20	12,00	0,58	10,42	13,25	**4,41**	7,59	0,54	3,56	5,15	11,08	0,92	0,90	9,40	14,04
500 x 20	9,43	0,41	8,38	10,25	**2,07**	7,36	0,36	1,49	2,68	11,81	-2,38	0,45	10,64	13,16

Tablo 5.14. DGA, DGA_{NEH} ve DGA_{EBPM} ile 2. grup test problemlerinde elde Δ_{ort} değerleri

Problem (nxm)	DGA	DGA_{NEH}	DGA -DGA_{NEH}	DGA_{EBPM}	DGA-DGA_{EBPM}
car1 (11x5)	0,00	0,00	0,00	0,00	0,00
car2 (13x4)	0,00	0,00	0,00	0,00	0,00
car3 (12x5)	0,00	0,00	0,00	0,00	0,00
car4 (14x4)	0,00	0,00	0,00	0,00	0,00
car5 (10x6)	0,14	0,05	0,09	0,08	0,06
car6 (8x9)	0,34	0,19	0,15	0,19	0,15
car7 (7x7)	0,00	0,00	0,00	0,00	0,00
car8 (8x8)	0,00	0,00	0,00	0,00	0,00
hel1 (100x10)	3,01	0,58	2,43	1,58	1,43
hel2 (20x10)	0,77	0,77	0,00	0,74	0,03
reC01 (20x5)	0,16	0,16	0,00	0,16	0,00
reC03 (20x5)	0,15	0,09	0,06	0,08	0,07
reC05 (20x5)	0,24	0,24	0,00	0,33	-0,09
reC07 (20x10)	0,56	0,65	-0,09	0,64	-0,08
reC09 (20x10)	0,30	0,24	0,06	0,15	0,15
reC11 (20x10)	0,42	0,50	-0,08	0,27	0,15
reC13 (20x15)	0,47	0,46	0,01	0,54	-0,07
reC15 (20x15)	1,26	1,15	0,11	0,96	0,30
reC17 (20x15)	1,05	1,13	-0,08	1,18	-0,13
reC19 (30x10)	4,31	4,13	0,18	3,81	0,50
reC21 (30x10)	2,77	2,62	0,15	2,58	0,19
reC23 (30x10)	3,08	3,04	0,04	3,05	0,03
reC25 (30x15)	4,35	4,23	0,12	3,96	0,39
reC27 (30x15)	3,42	3,18	0,24	2,78	0,64
reC29 (30x15)	5,37	4,54	0,83	4,69	0,68
reC31 (50x10)	8,11	4,20	3,91	7,07	1,04
reC33 (50x10)	3,95	3,62	0,33	3,37	0,58
reC35 (50x10)	0,84	0,81	0,03	0,73	0,11
reC37 (75x20)	11,54	5,57	5,97	9,91	1,63
reC39 (75x20)	9,17	4,34	4,83	7,98	1,19
reC41 (75x20)	11,77	6,69	5,08	10,10	1,67

Tablo 5.13'te "Fark" sütunlarında DGA metoduna ait Δ_{ort} ile ilgili metoda ait Δ_{ort} değerleri arasındaki fark verilmiştir. Tablo incelendiğinde Taillard'ın test problemlerinde DGA_{NEH} metodunun 11 grupta, DGA_{EBPM} metodunun 8 grupta DGA'dan daha iyi olduğu görülmektedir. Ayrıca DGA_{NEH}, 12 problem grubunun 11'inde melez DGA_{EBPM} metodundan daha iyi sonuçlar üretmiştir. Özellikle büyük problem gruplarında DGA_{NEH} metudunun diğer metotlara baskın çözümlere ulaştığı görülmektedir.

Tablo 5.14'te yer alan 31 problemin 19'ünde DGA_{EBPM} en az diğer iki yöntem kadar iyi çözümler üretmiştir. İki grupta yer alan problem gruplarına ait ortalama bağıl sapma yüzdeleri (Δ_{ort}) birlikte değerlendirildiğinde DGA ve DGA_{EBPM} arasında fark olduğu görülmektedir. Bu farkın %5 anlamlılık seviyesinde istatistiksel olarak anlamlı bir fark olup olmadığını görmek için 151 probleme ait Δ_{ort} değerlerine Mann-Whitney U testi uygulanmıştır. Bu test sonucuna ait test istatistikleri Tablo 5.15'te özetlenmiştir.

Tablo 5.15. DGA ile DGA_{EBPM} karşılaştırması Mann-Whitney U test istatistikleri

Metot	N	Sıra Ortalaması	Sıra Toplamı	Mann-Whitney U	Z	P
DGA	3020	3106,09	9380385,00	4301725,000	-3,815	,000
DGA_{EBPM}	3020	2934,91	8863435,00			

Tablo 5.15 incelendiğinde %5 önem düzeyinde DGA ve DGA_{EBPM} arasında anlamlı bir fark olduğu görülmektedir. Bu sonuç; makespan ölçütüne göre PATÇP'nin çözümünde, EBPM ile üretilen başlangıç çözümlerin DGA'nın çözüm kalitesini artırdığı ve DGA_{EBPM} ile DGA'dan daha başarılı sonuçlar elde edildiği şeklinde yorumlanabilir.

5.7. Ayrık EBM ile Yapılan Deneysel Çalışmalar

Orijinal EBPM'de sürekli konum vektörlerinin iş sıralarına dönüştürülmesi için gerçekleştirilen işlemler yöntemin hızını düşürmektedir. Bu durum sürekli konum vektörleri ile çalışan DGA, PSO vb. yöntemler için de geçerlidir. Bu yöntemlerin ayrık versiyonlarının önerildiği çalışmalardan esinlenerek geliştirilen Ayrık EBPM (AEBPM) önceki bölümlerde tanıtılmıştır. Bu başlık altında 2 grupta yer alan test problemleri AEBPM ve Refaranslanmış Yerel Arama (Pan ve diğ., 2008a) eklenmiş versiyonu

$AEBPM_{RYA}$ ile çözülerek sonuçları değerlendirilmiştir. Metotlar Tablo 5.16'da verilen başlangıç parametreleri ile çalıştırılmıştır.

Tablo 5.16. AEBPM ve $AEBPM_{RYA}$'de kullanılan başlangıç parametre değerleri

No	Parametre	Değer
1	El bombası sayısı (Ng)	nxm/20
2	Bir bombanın fırlatacağı şarapnel sayısı (Nq)	40
3	Patlama yarıçapı ($R_{e\text{-}initial}$)	$2\sqrt{n}$
4	Ajan bölgesi yarıçapı ($R_{t\text{-}initial}$)	$\sqrt{n}$
5	İlk iterasyondaki R_t değerinin son iterasyondaki R_t değerine oranını (R_{rd})	100
6	Durdurma kriter (İterasyon sayısı veya maksimum çalışma süresi)	nxmx0.06

Komşuluk operatörü olarak değiştirme (swap) kullanışmış, rassal olarak seçilen iki iş yerleri değiştirilerek şarapneller üretilmiştir. $AEBPM_{RYA}$'da her iterasyonda yerel arama kullanılmıştır. Tanımlanan AEBPM ve $AEBPM_{RYA}$ ile Taillard'ın test problemlerinin çözümünde elde edilen sonuçlar tablo 5.17'de EBPM ile birlikte verilmiştir.

Tablo 5.17. AEBPM ve $AEBPM_{RYA}$ ile Taillard'ın problemlerinde elde edilen sonuçlar

Problem Grubu	EBPM				AEBPM				$AEBPM_{RYA}$			
	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}
20 x 5	1,34	0,58	0,00	3,52	0,57	0,51	0,00	1,87	0,00	0,00	0,00	0,00
20 x 10	2,72	0,89	0,87	5,64	1,33	0,51	0,00	2,65	0,09	0,15	0,00	0,50
20 x 20	2,19	0,74	0,41	4,34	0,98	0,30	0,18	1,72	0,06	0,08	0,00	0,31
50 x 5	0,68	0,49	0,00	1,94	0,21	0,24	0,00	0,91	0,02	0,06	0,00	0,35
50 x 10	4,73	1,23	1,58	7,64	3,17	0,84	1,08	4,58	0,99	0,58	0,00	2,15
50 x 20	5,98	1,17	3,70	9,72	4,68	0,63	3,12	6,10	1,65	0,43	0,46	2,77
100 x 5	0,52	0,33	0,00	1,78	0,30	0,23	0,00	0,75	0,01	0,03	0,00	0,11
100 x 10	2,80	0,94	0,62	5,45	1,48	0,46	0,41	2,82	0,34	0,26	0,00	0,97
100 x 20	8,16	0,94	5,94	11,68	4,69	0,77	2,63	6,20	1,90	0,36	0,91	2,84
200 x10	5,51	1,26	2,09	8,28	0,98	0,29	0,48	2,16	0,27	0,17	0,00	0,87
200 x 20	14,20	0,68	12,46	15,63	3,63	0,56	2,25	4,94	1,86	0,33	1,10	2,74
500 x 20	11,67	0,46	10,29	12,87	1,76	0,33	1,17	2,56	0,88	0,19	0,41	1,33

Tablo 5.17 incelendiğinde tüm problem gruplarında AEBPM'nin EBPM'den daha başarılı olduğu, 3 metot arsında $AEBPM_{RYA}$ diğerlerine baskın olduğu anlaşılmaktadır. Özellikle büyük boyutlu problemlerde çözüm kalitesi her iki yöntemde de EBPM'ye göre önemli derecede artmıştır. $AEBPM_{RYA}$ ile ortalama bağıl sapma yüzdeleri (Δ_{ort}) tüm problem gruplarında %2'nin altına, 9 problem grubunda ise %1'in altına düşmüştür.

Tablo 5.18. AEBPM ve $AEBPM_{RYA}$ ile 2. grup test problemlerinde elde Δ_{ort} değerleri

Problem (nxm)	EBPM				AEBPM				$AEBPM_{RYA}$			
	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}	Δ_{ort}	Δ_{std}	Δ_{min}	Δ_{max}
car1 (11x5)	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
car2 (13x4)	0,15	0,66	0,00	2,93	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
car3 (12x5)	0,76	0,54	0,00	1,19	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
car4 (14x4)	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
car5 (10x6)	0,54	0,52	0,00	1,31	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
car6 (8x9)	0,08	0,24	0,00	0,76	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
car7 (7x7)	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
car8 (8x8)	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
hel1 (100x10)	2,06	0,86	0,58	3,49	0,35	0,08	0,19	0,39	-0,01	0,04	-0,19	0,00
hel2 (20x10)	2,39	0,79	0,74	3,68	0,70	0,16	0,00	0,74	-0,07	0,23	-0,74	0,00
reC01 (20x5)	1,19	1,48	0,16	6,34	0,14	0,05	0,00	0,16	0,08	0,08	0,00	0,16
reC03 (20x5)	0,59	0,65	0,18	2,98	0,12	0,08	0,00	0,18	0,00	0,00	0,00	0,00
reC05 (20x5)	0,66	0,72	0,24	2,50	0,24	0,00	0,24	0,24	0,21	0,09	0,00	0,24
reC07 (20x10)	1,09	0,26	0,00	1,15	0,92	0,39	0,13	1,15	0,00	0,00	0,00	0,00
reC09 (20x10)	2,24	0,74	0,65	3,32	0,47	0,31	0,13	1,30	0,00	0,00	0,00	0,00
reC11 (20x10)	1,97	1,00	0,84	4,26	0,72	0,31	0,14	1,40	0,00	0,00	0,00	0,00
reC13 (20x15)	2,20	0,94	0,47	4,56	1,11	0,29	0,31	1,66	0,15	0,07	0,00	0,26
reC15 (20x15)	2,35	0,55	1,18	3,54	1,30	0,20	1,08	1,79	0,06	0,15	0,00	0,56
reC17 (20x15)	3,52	0,94	1,52	5,15	1,69	0,53	0,89	2,52	0,04	0,11	0,00	0,37
reC19 (30x10)	3,55	1,06	1,62	5,06	1,86	0,26	1,39	2,34	0,44	0,19	0,29	0,86
reC21 (30x10)	2,91	0,90	1,64	5,01	1,63	0,08	1,44	1,83	1,41	0,13	0,84	1,44
reC23 (30x10)	3,58	0,92	1,94	5,57	2,05	0,37	1,24	2,54	0,46	0,08	0,25	0,65
reC25 (30x15)	4,26	0,91	2,31	5,73	2,98	0,39	2,35	3,66	0,67	0,16	0,28	1,11
reC27 (30x15)	3,11	0,82	1,98	5,14	2,43	0,35	1,90	3,03	0,67	0,29	0,25	1,14
reC29 (30x15)	4,29	1,07	2,71	6,12	3,50	0,55	2,06	3,98	0,65	0,31	0,00	1,27
reC31 (50x10)	5,22	1,05	3,51	7,13	3,48	0,32	3,02	3,84	1,18	0,56	0,30	1,84
reC33 (50x10)	1,90	0,47	0,96	2,79	1,31	0,30	0,83	1,73	0,13	0,14	0,00	0,55
reC35 (50x10)	0,96	0,87	0,00	2,90	0,15	0,10	0,00	0,43	0,00	0,00	0,00	0,00
reC37 (75x20)	7,85	0,67	6,65	9,49	4,62	0,31	3,98	5,33	2,38	0,31	1,74	2,87
reC39 (75x20)	6,49	0,82	4,70	7,86	3,51	0,45	2,18	4,11	1,33	0,25	0,94	1,67
reC41 (75x20)	8,07	0,84	6,88	10,16	5,02	0,29	4,46	5,48	2,47	0,37	1,77	3,15

AEBPM ve $AEBPM_{RYA}$ ile 2. grup test problemlerinin çözümünde elde sonuçlar Tablo 5.18'de verilmiştir. Bu gruptaki test problemlerinde de 3 metot arsında $AEBPM_{RYA}$'nin diğerlerine baskın olduğu görülmektedir.

Bu grupta yer alan Heller (1960) tarafından üretilen iki problem grubunda üst sınır değerin alındığı kaynaktan (Ancau, 2012) daha iyi bir makespan değerine ulaşılmış olup çözümleri Ek-C'de verilmiştir. Ek-C'de verilen bu makespan değerlerine daha önce (Abdel-Basset ve diğ., 2018) tarafından da ulaşılmıştır.

5.8. AEBPM'nin Performansın Kabul Görmüş Metasezgisellerle Karşılaştırılması

Bu başlık altında AEBPM ve $AEBPM_{RYA}$ yöntemlerinin performansları literatürde kabul görmüş çalışmalarla karşılaştırılmıştır. Öncelikle literatürde durdurma kriteri olarak maksimum çalışma zamanını kullanan çalışmalar araştırılmış ve seçilen çalışmalar gruplara ayrılarak aşağıdaki karşılaştırmalar yapılmıştır.

İlk grupta Ayrık Diferansiyel Gelişim (DDE) (Pan ve diğ., 2008a), Referanslanmış Yerel Arama (RLS) ile geliştirilmiş Ayrık Diferansiyel Gelişim (DDE_{RLS}) (Pan ve diğ., 2008a), Değişken Komşuluk Arama (VNS) ile geliştirilmiş Parçacık Sürü Optimizasyonu (PSO_{VNS}) (Tasgetiren et al., 2007) ve Genetik Algoritma ile Değişken Komşuluk Arama metotlarının birlikte kullanıldığı melez bir sezgisel olan $NEGA_{VNS}$ (Zobolas ve diğ., 2009) seçilmiştir. Bu metotlarda kullanılan maksimum çalışma süreleri ve çalışmaların yapıldığı bilgisayarların özellikleri Tablo 5.19'da verilmiştir.

Tablo 5.19. Karşılaştırılan metotlara ait bilgiler

Yöntem	İşlemci	Programlama Dili	Maksimum Çalışma Süresi
PSO_{VNS}	Intel P4 2.6 GHz	C	n<100 ise 300 n>=100 ise 600
DDE	Intel P4 3.0 GHz	Visual C++	$nxmx15x10^{-3}$
DDE_{RLS}	Intel P4 3.0 GHz	Visual C++	$nxmx15x10^{-3}$
$NEGA_{VNS}$	Intel P4 2.4 GHz	C++	nxmx0,1

Çalışmanın önceki bölümlerinde Diferansiyel Gelişim Algoritması ile karşılaştırmalar yapıldığı ve bu çalışmada önerilen yöntemle aynı yerel arama kullanıldığı için DDE ve DDE_{RLS} özellikle bu gruba dahil edilmiştir. İstenildiğinde daha rahat ulaşılabilmesi için

ilgili kaynaktaki kısaltmalar kullanılmıştır. Karşılaştırma için seçilen yöntemler tekrar kodlanmadan sonuçları ilgili çalışmadan alınarak Tablo 5.20'de verilmiştir.

Tablo 5.20. Taillar'ın Problemlerine Ait Ortalama Bağıl Yüzde Sapma (Δ_{ort}) Değerleri

Problem Grubu (nxm)	PSO_{VNS}	DDE	DDE_{RLS}	$NEGA_{VNS}$	AEBPM	$AEBPM_{RYA}$
20×5	0.03	0,33	0,04	**0.00**	0,57	**0,00**
20×10	0.02	0,43	0,01	**0.01**	1,33	0,09
20×20	0.05	0,46	**0,02**	**0.02**	0,98	0,06
50×5	**0.00**	0,09	**0,00**	**0.00**	0,21	0,02
50×10	**0.57**	1,42	0,45	0.82	3,17	0,99
50×20	1.36	2,66	**0,66**	1.08	4,68	1,65
100×5	**0.00**	0,03	**0,00**	**0.00**	0,30	0,01
100×10	0.18	0,67	0,15	**0.14**	1,48	0,34
100×20	1.45	2,68	**0,98**	1.40	4,69	1,90
200×10	0.18	0,43	**0,07**	0.16	0,98	0,27
200×20	1.35	2,21	**0,99**	1.25	3,63	1,86
500×20	-	1,18	**0,49**	0.71	1,76	0,88
Maks. Çalışma Süresi	n<100 ise 300 n>=100 ise 600	nxmx0,015		nxmx0,1	nxmx0,06	

Tablo 5.20 incelendiğinde DDE_{RLS} metodunun diğer metotlara göre çok daha kısa süre çalıştırılmış olmasına rağmen 12 problem grubunun 5'inde optimal değere diğer metotlardan daha çok yaklaşmıştır. 3 problem grubunda ise en az diğer metotlar kadar iyi sonuç üretmeyi başarmıştır. İlk problem grubunda $AEBPM_{RYA}$, $NEGA_{VNS}$ ile birlikte en iyi değere sahiptir. Diğer problem gruplarında ise AEBPM ve $AEBPM_{RYA}$ diğer yöntemlerin gerisinde kalmış olmakla birlikte farkın en çok olduğu problem grubunda (100×20) bile en iyi yöntemle aradaki fark %1'in altındadır.

Kullanılan programlama dilinin metotların hızı, dolayasıyla maksimum çalışma süresi durdurma kriteri altındaki performansı üzerinde önemli bir etkisi bulunmaktadır. Bu nedenle adil bir karşılaştırma için literatürde Matlab ile kodlama yapılan çalışmalar araştırılmıştır. Ayrıca literatür taramasında EBPM ve Yapay Arı Kolonisi Algoritmasının birlikte kullanımı ile oluşturulan melez yöntemler dikkat çekmektedir. Bu nedenle AEBPM'nin YAKA tabanlı melez algoritmalarla da karşılaştırılmasında fayda bulunmaktadır. Li ve Yin (2012) geliştirdikleri Komposit Ayrık Yapay Arı Kolonisi

(Composite Discrete Artifical Bee Colony, CDABC) metodunu Tablo 5.21'de verilen metotlarla karşılaştırmışlardır.

Tablo 5.21. CDABC'nin Karşılaştırıldığı Çalışmalar (Li ve Yin, 2012)

Metod	Yıl	Çalışma
Taillard'ın hızlandırıcılarıyla geliştirdiği NEH (NEHT)	1990	Taillard (1990)
Ruiz ve diğerlerinin genetik algoritması (GA-RMA)	2006	Ruiz ve diğ. (2006)
Osman ve Potts'un Tavlama Benzetimi (SA_OP)	1989	Osman ve Potts (1989)
Widmer ve Herz'in önerdiği Tabu Arama Algoritması (SPIRIT)	1989	Widmer ve Hertz (1989)
Andreas'ın geliştirdiği Genetik Algoritması (GA_AN)	2004	Nearchou A. (2004)
Aldowaisan ve Allahvedi'nin önerdiği GA (GA_AA)	2003	Aldowaisan ve Allahverdi (2003)
Murata ve diğ. tarafından önerilen Melez GA (GA_M1T)	1996	Murata ve diğ. (1996)
Yeni Parçacık Sürü Optimizasyonu önerisi (NPSO)	2008	Lian ve diğ. (2008)
karınca koloni optimizasyon algoritması (ACO)	2008	Yağmahan ve Yenisey (2008)
Geliştirilmiş Parçacık Sürü Optimizasyonu (IPSO)	2008	Zhang ve diğ. (2008)
Akış Tipi Çizelgeleme için GA (GAReev)	1995	Reevest (1995)
İki Yeni Karınca Kolonisi Optimizasyonu önerisi (M-MMAS and PACO)	2004	Rajendran ve Ziegler (2004)

CDABC, Matlab 7,0 programı kullanılarak kodlanmış ve Pentium 3,0 GHz işlemci 512 MB RAM'e sahip bir bilgisayar kullanılmıştır. Tablo 5.21'de belirtilen metotlardan en iyi sonuca sahip 4 metot ve CDBC ait ortalama bağıl sapma yüzdeleri (Li ve Yin, 2012)den alınmış olup AEBPM ve AEBPM$_{RYA}$'ya ait sonuçlarla birlikte Tablo 5.22'de verilmiştir. Tablodaki Δ_{ort} değerleri Taillard'ın test problemlerine aittir.

Tablo 5.22. CDBC ve 4 metot ile AEBPM ve AEBPM$_{RYA}$'nin Δ_{ort} değerleri

Problem Grubu (nxm)	M-MMAS	NPSO	IPSO	PACO	CDABC	AEBPM	AEBPM$_{RYA}$
20×5	0,11	0,21	0,04	0,20	0,03	0,57	**0,00**
20×10	0,15	0,37	0,23	0,32	0,21	1,33	**0,09**
20×20	0,09	0,24	0,19	0,21	0,17	0,98	**0,06**
50×5	0,02	**0,01**	0,05	0,08	0,03	0,21	0,02
50×10	1,30	1,85	1,05	0,90	**0,90**	3,17	0,99
50×20	2,10	1,59	1,83	**1,46**	1,77	4,68	1,65
100×5	0,03	0,03	0,03	0,04	0,03	0,30	**0,01**
100×10	0,46	0,37	0,39	0,35	0,49	1,48	**0,34**
100×20	2,59	2,19	2,06	2,17	**1,32**	4,69	1,90
200×10	0,37	0,37	0,32	**0,26**	0,37	0,98	0,27
200×20	2,34	1,92	1,80	2,00	**1,36**	3,63	1,86
500×20	1,06	1,19	1,16	0,98	**0,70**	1,76	0,88
Maks. Çalışma Süresi	nxmx0,06						

Tablo 5.22 incelendiğinde AEBPM$_{RYA}$'nın 5 grupta diğer metotlardan daha iyi sonuçlar verdiği 2 grupta ise en az diğer metotlar kadar iyi değerler sahip olduğu görülmektedir. CDABC 4 grupta, NPSO 2. Grupta, PACO ise 1 grupta en iyi sonuçlara ulaşmıştır. AEBPM$_{RYA}$'nın en iyi sonuca ulaşmadığı problem gruplarında en iyi yönteme ait Δ_{ort} değeri ile arasındaki en büyük fark %0,58'dir. Bu sonuçlara bakarak AEBPM$_{RYA}$'nın maksimum tamamlanma zamanı ölçütüne göre en az Tablo 5.22'de yer alan diğer yöntemler kadar veya onlardan daha başarılı olduğu söylenebilir.

Li ve Yin, CDABC ile 2. Grupta yer alan test problemlerini de çözmüşlerdir. 2. Grupta yer alan test problemleri Xie ve diğ. (2014) tarafından da maksimum çalışma süresi durdurma ile çözülmüştür. Xie ve diğ. (2014) önerdikleri melez Öğretme- Öğrenme Tabanlı Optimizasyon Algoritmasını (HTLBO) literatürde kabul görmüş diğer yöntemlerle karşılaştırmışlardır. HTLBO, Öğretme- Öğrenme Tabanlı Optimizasyon Algoritması (TLBO) ve Değişken Komşuluk Arama (VNS) C++ ile koldanmış ve IntelCore2 Duo 2.0 GHz işlemciye sahip bir bilgisayar kullanılmıştır. Heller'in sunduğu problemeler dışında 2. grupta yer alan test problemlerinin CDBC, TLBO, HTLBO ve VNS yöntemleri ile çözümünden elde edilen ortalama bağıl sapma yüzdeleri (Δ_{ort}) ilgili kaynaklardan alınarak Tablo 5.23'te verilmiştir.

Tablo 5.23'te verilen sonuçlar incelendiğinde Carlier test problemlerinde AEBPM ve AEBPM$_{RYA}$; HTLBO, VNS ve CDABC ile benzer, TLBO'dan daha iyi Δ_{ort} değerlerine ulaşmıştır. Reevest'in test problemlerinde, 39 ve 4 numaralı problemler hariç HTLBO, diğer yöntemlerle aynı veya daha iyi Δ_{ort} değerlerine sahiptir. 39 ve 4 numaralı problemlerde ise CDABC daha iyi sonuçlara ulaşmıştır. Bu problem grubunda AEBPM ve AEBPM$_{RYA}$, TLBO'dan daha iyi Δ_{ort} değerlerine sahipken HTLBO ve VNS'nin geresinde kalmıştır.

Tabloda yer alan problemler arasında AEBPM$_{RYA}$, en yüksek Δ_{ort} değerlerine sahip olduğu problem 2,47 ile Rec41'dir. 75 iş ve 20 makinenin olduğu bu problemde AEBPM$_{RYA}$ ile bulunan maksimum tamamlanma zamanı(makespan) bu probleme ait üst sınırdan yalnızca ortalama %2.47 daha yüksektir. Bu problemde en iyi sonuca ulaşan CDABC ile arasındaki fark %1' den daha azdır. Bu durum ve yukarıdaki sonuçlar birlikte değerlendirildiğinde Tablo 5.23'te yer alan metotlar arasında genelde orta seviyede Δ_{ort} değerlerine sahip olan AEBPM$_{RYA}$ ile bu grupta yer alan problemler çözüldüğünde makul

sürede kabul edilebilir maksimum tamamlanma zamanına sahip çözümlere ulaşılmaktadır. Bununla birlikte kodlama dillerinin, kod yapılarının, kullanılan bilgisayarların ve çalışma sürelerinin farklı olması nedeniyle sadece tabloda yer alan sonuçlara bakarak metotlar arasında bir üstünlük sıralaması yapmak doğru bir yaklaşım olmayacaktır.

Tablo 5.23. CDBC, TLBO, HTLBO ve VNS yöntemlerine ait Δ_{ort} değerleri

Problem (nxm)	CDABC	HTLBO	TLBO	VNS	AEBPM	$AEBPM_{RYA}$
car1 (11x5)	0,00	0,00	0,00	0,00	0,00	0,00
car2 (13x4)	0,00	0,00	0,00	0,00	0,00	0,00
car3 (12x5)	0,00	0,00	0,324	0,00	0,00	0,00
car4 (14x4)	0,00	0,00	0,00	0,00	0,00	0,00
car5 (10x6)	0,00	0,00	0,593	0,00	0,00	0,00
car6 (8x9)	0,00	0,00	0,00	0,00	0,00	0,00
car7 (7x7)	0,00	0,00	0,00	0,00	0,00	0,00
car8 (8x8)	0,00	0,00	0,00	0,00	0,00	0,00
reC01 (20x5)	0,0321	0,00	0,160	0,00	0,14	0,08
reC03 (20x5)	0,00	0,00	0,00	0,00	0,12	0,00
reC05 (20x5)	0,242	0,00	0,242	0,242	0,24	0,21
reC07 (20x10)	0,00	0,00	0,911	0,00	0,92	0,00
reC09 (20x10)	0,00	0,00	1,607	0,00	0,47	0,00
reC11 (20x10)	0,00	0,00	1,887	0,00	0,72	0,00
reC13 (20x15)	0.135	0,00	1,347	0,00	1,11	0,15
reC15 (20x15)	0,133	0,00	2,205	0,00	1,30	0,06
reC17 (20x15)	0,073	0,00	3,785	0,00	1,69	0,04
reC19 (30x10)	0,392	0,215	3,440	0,287	1,86	0,44
reC21 (30x10)	1,056	0,199	2,677	0,694	1,63	1,41
reC23 (30x10)	0,428	0,174	2,984	0,448	2,05	0,46
reC25 (30x15)	0,664	0,279	4,497	0,637	2,98	0,67
reC27 (30x15)	0,615	0,252	4,130	0,380	2,43	0,67
reC29 (30x15)	0,791	0,00	6,209	0,437	3,50	0,65
reC31 (50x10)	0,348	0,263	5,818	0,936	3,48	1,18
reC33 (50x10)	0,025	0,00	3,308	0,273	1,31	0,13
reC35 (50x10)	0,00	0,00	0,458	0,00	0,15	0,00
reC37 (75x20)	1,902	1.495	6,461	3,979	4,62	2,38
reC39 (75x20)	1,089	1.238	4,993	2,025	3,51	1,33
reC41 (75x20)	1,682	1,855	5,907	0,448	5,02	2,47
Maks. Çalışma Süresi	nxnxmx0,005	mxnx0,3			nxmx0,06	

Tablo 5.23'te değerlendirilmesi gereken önemli bir sonuç da; $AEBPM_{RYA}$, TBLO'dan genellikle daha düşük veya eşit Δ_{ort} değerlerine sahipken AEBPM'nin HTLBO'dan daha yüksek Δ_{ort} değerlerine sahip olmasıdır. $AEBPM_{RYA}$, AEBPM'ye, HTLBO ise TBLO yerel arama eklenerek oluşturulmuştur. HTLBO'da yerel aramanın çözüm kalitesini $AEBPM_{RYA}$'ya göre göreceli olarak daha fazla artırdığı görülmektedir. İlerleyen çalışmalarda aynı yerel arama metodunun AEBPM'de de kullanılması halinde çözüm kalitesinin daha fazla gelişme ihtimali bulunmaktadır.

Bu başlık altında 3 grup halinde yapılan karşılaştırmalarda Matlab kullanarak kodlanan AEBPM ve $AEBPM_{RYA}$; Tablo 5.22'de Matlab kullanılarak kodlanan metotlarla, Tablo 5.20 ve Tablo 5.23'te çoğunlukla başka programlama dilleri ile kodlanmış metotlarla karşılaştırılmıştır. Matlab'la kodlanan metotlar arasında $AEBPM_{RYA}$'nın performansı daha öne çıkarken, diğer programlama dilleri ile kodlanan metotlar karşısında $AEBPM_{RYA}$'nın performansı daha gerilerde kalmıştır. Bu durumun metasezgisel metotların kodlanlamasında kullanılan programlama dillerinin hız farklarından kaynaklandığı düşünülmektedir. Bu çalışmada kullanılan kodlar profesyonel bir destek alınmadan geliştirilmiş olup Ek-D'de verilmiştir.

AEBPM ve $AEBPM_{RYA}$ yöntemlerinin maksimum tamamlanma zamanı ölçütüne göre performanslarının literatürde kabul görmüş çalışmalarla karşılaştırılması sonucunda; bu yöntemlerim, özellikle melez $AEBPM_{RYA}$ yönteminin, permütasyon akış tipi çizelgeleme problemlerinin (PATÇP) çözümümde makul sürelerde kabul edilebilir çözümler ürettiği görülmüştür.

6. SONUÇLAR VE ÖNERİLER

Bu çalışmada öncelikle orijinal EBPM'nin permütasyon akış tipi sıralama problemlerinde kullanılabilirliği ve ajan bölgesi yarıçapı (R_t) ile optimal arama yönü özelliklerinin EBPM'nin performansına olan katkısını araştırmak üzere testler gerçekleştirilmiştir. Daha sonra başlangıç çözümlerin bir kısmının NEH algoritması ile oluşturulmasının çözüm kalitesi üzerine etkisi incelenmiştir. Son olarak ayrık EBPM ve melez $AEBPM_{RYA}$ metotları oluşturulmuştur. Geliştirilen metotların performansları sık kullanılan test problemleri üzerinde test edilmiş ve literatürde kabul görmüş yöntemlerle karşılaştırılmıştır.

EBPM'nin de dahil olduğu metasezgisel yöntemlerin sonuçları az veya çok rassallık içerir; optimum çözüme yakınsaması ispat edilemez ve performansları başlangıç parametrelerinin seçimi gibi bir çok etkene bağlıdır. Sürekli sayılardan oluşan konumların iş sıralarına dönüşümünde kullanılacak kodlama yönetimi de EBPM'nin çizelgeleme problemleri üzerindeki performansında etkilidir. EBPM [-1,1] aralığında ölçeklendirilmiş çözüm uzayını yönteme özgü olan ajan bölgesi yarıçapı özelliği sayesinde homojen olarak taramaktadır. Fakat en iyi iş sırasının bulunmasının hedeflendiği kesikli çizelgeleme problemlerinde sürekli konum bilgileri doğrudan kullanılamadığından dönüşüm için faklı kodlama yöntemlerine ihtiyaç duyulmaktadır. Uygulanan dönüşüm sonrasında çözüm uzayının yapısı büyük oranda değiştiğinden ajan bölgesi yarıçapının etkisi hakkında soru işareti oluşmaktadır.

Bu çalışmada konumların iş sıralarına dönüşümünde En Küçük Pozisyon Değeri (EKPD) yöntemi kullanılmış ve ajan bölgesinin katkısının araştırılmasına yönelik testler gerçekleştirilmiştir. Test sonuçları Kruskal-Wallis Testi ve Whitney U testi uygulanarak incelenmiş ve ajan bölgesinin EBPM'nin performansına %5 anlamlılık seviyesinde istatistiksel olarak anlamlı bir katkısının olduğu görülmüştür. Fakat bu sonucun durdurma kriteri, başlangıç parametrelerinin seçimi vb. test şartlarına bağlı olarak değişebileceği unutulmamalıdır.

Optimal arama yönünün, düzgün şekilde artan veya azalan sürekli fonksiyonların optimizasyonunda metodun performansını artırdığı önceki çalışmalardan bilinmektedir. PATÇP kesikli yapıya sahip (kombinatoryal) olduğundan bu özelliğin PATÇP'de

çözüme katkısının araştırılmasına ihtiyaç duyulmuştur. Bu amaçla optimal arama yönü bulunan (Nq-oay=5) ve bulunmayan (Nq-oay=0) EBPM'ler ile elde edilen sonuçlar Whitney U testi uygulanarak diğer değişkenlerin sabit olduğu ikişerli gruplar (EBPM1-EBPM2 ve EBPM3- EBPM4) halinde karşılaştırılmıştır. Whitney U testleri sonucunda %5 önem düzeyinde EBPM1-EBPM2 karşılaştırmasında anlamlı fark varken EBPM3-EBPM4 karşılaştırmasında anlamlı fark görülmemiştir. Bu durum değişik nedenlerden kaynaklanmış olabilir. Bunlardan ilki, iki özelliğin etkileşiminin anlamlı etkiye sahip olmasıdır. EBPM1 ve EBPM2'de ajan bölgesi yarıçapı (R_t)=√n) olarak, EBPM3 ve EBPM4'te $R_t = 10^{-10}$ olarak seçilmiştir. Bu nedenle optimal arama yönünün tek başına anlamlı bir katkısının olmadığı fakat ajan bölgesi yarıçapı ile etkileşiminin anlamlı bir katkısının olduğu düşünülebilir. Bu çalışmada elde edilen verilerle bu konuda kesin bir sonuca varılamamıştır. İlerleyen çalışmalarda deney tasarımı teknikleri ile daha kesin sonuçlara ulaşabileceği düşünülmektedir.

EBPM'nin PATÇP'nin maksimum tamamlanma zamanı ölçütüne göre çözümündeki performansını değerlendirmek için iki aşamalı bir değerlendirme yapılmıştır. İlk aşamada EBPM'nin makul sürede kabul edilebilir sonuçlar üretebildiğini doğrulamak için durdurma kriteri, literatürdeki birçok çalışmaya benzer veya daha kısa seçilerek Taillard'ın test problemleri çözülmüştür. Bu yolla çözüm için harcanan sürenin literatürde kabul görmüş süreleri aşmaması sağlanmıştır. Elde edilen sonuçlar literatürde kabul görmüş metotların sonuçları ile karşılaştırılmıştır. Karşılaştırmaya dahil edilen 11 problem grubunun ilk 8'inde EBPM diğer yöntemlerden daha iyi ortalama bağıl sapma yüzdelerine (Δ_{ort}) ulaştığı görülmüştür.

İkinci aşamada sağlıklı bir karşılaştırma için Diferansiyel Gelişim Algoritması (DGA) Matlab ile kodlanmış ve çok sayıda test problemi EBPM ve DGA ile aynı bilgisayarda çözülmüştür. Metotlar arasında %5 anlamlılık seviyesinde istatistiksel olarak anlamlı bir fark olup olmadığı, Mann-Whitney U testi uygulanarak incelenmiş ve EBPM ile DGA arasında %5 önem düzeyinde anlamlı bir fark olmadığı, EBPM'nin; tanımlanan şartlar altında DGA ile benzer performansa sahip olduğu görülmüştür. Bu sonuç; EBPM'nin; tanımlanan şartlar altında literatürde kabul görmüş yöntemler ile benzer performansa sahip olduğu, PATÇP'nin makespan ölçütüne göre çözümünde kullanılabileceği şeklinde yorumlanabilir.

EBPM'nin PATÇP'nin makespan ölçütüne göre çözümünde kabul edilebilir sonuçlar ürettiği doğrulandıktan sonra performansını geliştirmek için bazı iyileştirmeler yapılmıştır. Bu kapsamda başlangıç çözümlerden biri NEH ile üretilerek $EBPM_{NEH}$ oluşturulmuştur. Bu yolla, Taillard'ın test problemlerinde elde edilen sonuçlarda problem gruplarına göre %0.03 ile %9.79 arasında değişmekle birlikte ortalama %2,55 ilerleme (Tablo 5.11) sağlanmıştır. Genellikle problem boyutu büyüdükçe ilerleme oranının da arttığı gözlemlenmiştir.

Orijinal EBPM'de sürekli konum vektörlerinin iş sıralarına dönüştürülmesi için gerçekleştirilen işlemler yöntemin hızını düşürdüğü için bu çalışma kapsamında Ayrık EBPM (AEBPM) ve yerel arama eklenmiş versiyonu melez $AEBPM_{RYA}$ geliştirilmiştir. Çözüm kalitesinin her iki yöntemde de EBPM'ye göre önemli derecede arttığı görülmüştür. $AEBPM_{RYA}$ ile ortalama bağıl sapma yüzdeleri (Δ_{ort}) Taillard'ın tüm test problem gruplarında %2'nin altına, 9 problem grubunda ise %1'in altına düşmüştür.

Tablo 6. 1. Taillar'ın Problemlerinde NEH, AEBPM ve $AEBPM_{RYA}$'nin Δ_{ort} değerleri

Problem Grubu (nxm)	NEH	$EBPM_{NEH}$	AEBPM	$AEBPM_{RYA}$	NEH-$EBPM_{RYA}$
20 x 5	3,2585	1,1863	0,57	**0,00**	3,2585
20 x 10	4,6011	2,5682	1,33	**0,09**	4,5111
20 x 20	3,7309	2,1577	0,98	**0,06**	3,6709
50 x 5	0,7272	0,3685	0,21	**0,02**	0,7072
50 x 10	5,0729	3,8336	3,17	**0,99**	4,0829
50 x 20	6,6596	5,2905	4,68	**1,65**	5,0096
100 x 5	0,5272	0,3149	0,30	**0,01**	0,5172
100 x 10	2,2150	1,4974	1,48	**0,34**	1,875
100 x 20	5,3446	4,9218	4,69	**1,90**	3,4446
200 x10	1,2579	1,2572	0,98	**0,27**	0,9879
200 x 20	4,4076	4,4057	3,63	**1,86**	2,5476
500x20	2,0661	2,0661	1,76	**0,88**	1,1861
Genel Ort.	3,3224	2,4890	1,9817	**0,6725**	2,6499

Tablo 6.1'de Taillar'ın problemlerinde NEH, $EBPM_{NEH}$, AEBPM ve $AEBPM_{RYA}$'nin ortalama bağıl sapma yüzdeleri (Δ_{ort}) ve $AEBPM_{RYA}$ ile NEH arasındaki fark verilmiştir. Tablo6.1'de de görüldüğü üzer $EBPM_{NEH}$, AEBPM ve $AEBPM_{RYA}$ ile elde edilen sonuçlar NEH algoritmasına göre daha iyidir. Özellikle $AEBPM_{RYA}$ tüm porblem gruplarında diğer yöntemlere göre daha iyi Δ_{ort} değerlerine sahiptir. $AEBPM_{RYA}$ ile NEH algortitmsına göre ortalama 2,6499 daha iyi Δ_{ort} değerlerine ulaşılmıştır.

Geliştirilen $AEBPM_{RYA}$ ve AEBPM'nin performansı önceki bölümde literatürde kabul görmüş yöntemlerle karşılaştırılmıştır. Bu karşılaştırmalarda $AEBPM_{RYA}$ ile bazı problemlerde diğer yöntemlerden daha küçük veya onlara eşit Δ_{ort} değerlerine ulaşırken, genellikle diğer yöntemler arasında orta seviyede değerler elde edildiği gözlemlenmiştir. Fakat $AEBPM_{RYA}$'nın en yüksek Δ_{ort} değerlerine sahip olduğu problem grubunda dahi en iyi yöntemle arasındaki fark %1'den daha azdır.

Geliştirilen $AEBPM_{RYA}$ ve AEBPM'nin performansı literatürde kabul görmüş yöntemlerle karşılaştırılmıştır. Bu karşılaştırmalarda $AEBPM_{RYA}$ ile bazı problemlerde diğer yöntemlerden daha küçük veya onlara eşit Δ_{ort} değerlerine ulaşırken, genellikle diğer yöntemler arasında orta seviyede değerler elde edildiği gözlemlenmiştir. Fakat $AEBPM_{RYA}$'nın en yüksek Δ_{ort} değerlerine sahip olduğu problem grubunda dahi en iyi yöntemle arasındaki fark %1' den daha azdır.

Kullanılan bilgisayar, yazılım dili ve kodların yapısı metasezgisel yöntemlerin hızını dolayısıyla performansını etkilemektedir. Aynı bilgisayar ve programlama dili kullanılsa dahi kodların yapısı algoritmaların hızını etkilemektedir. Bu çalışmada geliştirilen yöntemlerin kodları profesyonel destek alınmadan geliştirilmiştir. Karşılaştırma için seçilen yöntemler ise yeniden kodlanmadan sonuçları ilgili çalışmalardan alınarak doğrudan kullanılmıştır. Matlab kullanarak kodlanan AEBPM ve $AEBPM_{RYA}$; benzer şekilde Matlab kullanılarak kodlanan metotlarla karşılaştırıldığında $AEBPM_{RYA}$'nın performansı daha ön plana çıkarken, diğer programlama dilleri ile kodlama yapılan çalışmalarla yapılan karşılaştırmalarda $AEBPM_{RYA}$'nın performansı daha orta seviyelerde kalmıştır. Bu durumun programlama dillerinin hız farklarından kaynakladığı düşülmektedir.

Kodlama ve kullanılan bilgisayar ile ilgili faktörlere ek olarak, başlangıç parametrelerinin seçimi de metasezgisellerin performansı üzerinde önemli bir etkiye sahiptir. Bu nedenle tek bir çalışmanın sonucuna bakarak bir metasezgisel yöntemin değerlerinden daha iyi veya kötü olduğunu iddia etmek doğru bir yaklaşım olmayacaktır. Fakat metasezgiselin kabul edilebilir sonuçlara ulaşılabilirliği hakkında daha doğru kararlar verilebilir.

Bu çalışma sonucunda; orijinal EBPM'nin ve bu çalışma kapsamında geliştirilen AEBPM ve $AEBPM_{RYA}$ yöntemlerinin PATÇP'lerin çözümünde makul sürelerde kabul edilebilir maksimum tamamlanma zamanlarına sahip sonuçlara ulaşabildiği ve bu problemlerin çözümünde kullanılabileceği görülmüştür.

İzleyen çalışmalarda, önerilen yöntemlerin performansları başlangıç parametreleri deney tasarımı ile belirlenerek geliştirilebilir; farklı yerel arama yöntemlerinin kullanılması veya başlangıç çözümlerin farklı yöntemlerle oluşturulmasının çözüm kalitesine etkisi araştırılabilir. Ayrıca farklı problem türleri ve performans ölçütleri üzerinde çalışılması alternatif araştırma konuları olabilir. Diğer metasezgiseller üzerinde çalışan araştırmacılar için yöntemlere ajan bölgesi vb. bir özelliğin eklenerek etkisinin araştırılması da ileri bir çalışma konusu olarak düşünülebilir.

KAYNAKLAR

Abdel-Basset, M., Manogaran, G., El-Shahat, D. (2018). Hybrid whale optimization algorithm based on local search strategy for thepermutation flow shop scheduling problem. *Future Generation Computer Systems, 85*, 129–145.

Abraham, A., Xhafa, F. (2008). *Metaheuristics for Scheduling in Industrial and Manufacturing Applications.* Springer-Verlag.

Agarwal, A., Çolak, S., Eryarsoy, E. (2006). Improvement heuristic for the flow-shop scheduling problem: An adaptive-learning approach. *European Journal of Operational Research, 169*(3), 801-815.

Aguiar, H., Junior, O. (2015). Evolutionary Global Optimization, Manifolds and Applications. *Studies in Systems, Decision and Control, 43*, 17-18.

Ahmadizar, F. (2012). A new ant colony algorithm for makespan minimization in permutation flow shops. *Computers and Industrial Engineering, 63*(2), 355-361.

Ahrari, A., Shariat-Pinahi, M., Atai, A. (2009). GEM : A novel evolutionary optimization method with improved neighborhood search. *Applied Mathematics and Computation, 210*(2), 376-386.

Ahrari, A., Atai, A. (2010). Grenade Explosion Method - A novel tool for optimization of multimodal functions. *Applied Soft Computing Journal, 10*(4), 1132-1140.

Ahrari, A., Saadatmand, M., Shariat-Panahi, M., Atai, A. (2010). On the limitations of classical benchmark functions for evaluating robustness of evolutionary algorithms. *Applied Mathematics and Computation, 215*(9), 3222-3229.

Aker, S. B. (1956). A graphical approach to production scheduling problems. *Operations Research, 4*(2), 244-245.

Aldowaisan, T., Allahverdi, A. (2003). New heuristics for no-wait flowshops to minimize makespan. *Computers & Operations Research, 30*(8), 1219-1231.

Ali, A., Gajpal, Y., Elmekkawy, T. (2021). Distributed permutation flowshop scheduling problem with total completion time objective. *OPSEARCH, 58*(2), 425-447.

Amallynda, I. (2019). TheDiscrete Particle Swarm Optimization Algorithms for Multi-Objective Permutation Flowshop Scheduling Problem. *Jurnal Teknik Industri, 20*(2), 105-116.

Ancau, M. (2012). On Solving Flowshop Scheduling Problems. *Proceedings of the Romanian Academy - Series A, 13*(1), 71-79.

Arslan, A. K., Tunç, Z., Cemil, Ç. (2020). Normal Dağılıma Uygunluğu Değerlendirmek için Açık Kaynak Web Tabanlı Yazılım: Normal Dağılımı İnceleme Yazılımı. *Fırat Tıp Dergisi, 25*(2), 62-68.

Aslan, E. (2022). *İmalat Sistemleirnde Çizelgeleme* (2 b.). Ankara: Seçkin Yayıncılık.

Bacha, S. Z., Benatchba, K., Tayeb, F. B.-S. (2022). Adaptive search space to generate a per-instance genetical gorithm for the permutation flow shop problem. *Applied Soft Computing*(124), 1-13.

Baker, K., Trietsch, D. (2009). *Principles of Sequencing and Scheduling.* John Wiley & Sons, Inc.

Baysal, E. M., Durmaz, T., Sarucan, A., Engin, O. (2012). Açık Atölye Tipi Çizelgeleme Problemlerinin Paralel Kanguru Algoritması ile Çözümü. *Gazi Üniv. Müh. Mim. Fak. Der., 27*(4), 855-864,.

Bayzan, Ş.,(2009). GPRS Verileri Yardımıyla Araç Rotalarının Belirlenmesi Problemine Farklı Bir Yaklaşım. *XI. Akademik Bilişim Konferansı* Şanlıurfa, 11-13 Şubat 2009.

Bean, J. (1994). Genetic Algorithms and Random Keys for Sequencing and Optimization. *HomeORSA Journal on Computing, 6*(2), 154-160.

Bendeş, E., Özkan, Ç. (2008). Direk Lineer Trasformasyon Yönteminde Yapay Zeka Tekniklerinin Uygulanması. *2. Uzaktan Algılama ve Coğrafi Bilgi Sistemleri Sempozyumu UZAL-CBS.* Kayseri.

Blum, C., Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys, 35*(3), 268-308.

Bouchekara, H., Chaib, ·., Abido, ·. (2016). Multiobjective optimal power flow using a fuzzy based grenade explosion method. *Energy Systems, 7*, 699-721.

Campbell, H., Dudek, R., Smith, M. (1970, 6). A Heuristic Algorithm for the n Job, m Machine Sequencing Problem. *Management Science, 16*(10), 630-637.

Carlier, J. (1978). Ordonnancements a contraintes disjonctives. *R.A.I.R.O. Recherche operationelle/Operations Research, 12*, 333-351.

Cerit, B., Onural, A. Ş., Yilmaz, B. (2005). Montaj ve işleme alt sistemlerini içeren bir esnek üretim sisteminin iki aşamalı çizelgelenmesi. *Teknoloji, 8*, 148.

Conway , R. W., Maxwell, W. L., Miller, L. W. (1967). *Theory of Scheduling. USA,* Addison-Wesley Publishing Co.

Conway, R. W., Maxwell, W. L., Miller, L. W. (2003). *Theory of Scheduling.* New York: Dover Publications.

Cura, T. (2006). *Modern Meta Sezgisel Teknikler ve Uygulamaları.* İstanbul: Papatya Yayınları.

Çelik, Y., Yıldız, İ., Karadeniz, A. T. (2019). Son Üç Yılda Geliştirilen Metasezgisel Algoritmalar Hakkında Kısa Bir İnceleme. *Avrupa Bilim ve Teknoloji Dergisi* (Özel sayı), 463-477.

Çiçekli, U. G., Bozkurt, s. (2016). Permütasyon Akış Tipi Çizelgeleme Probleminin Dağınık Arama İle Optimizasyonu. *Ege Akademik Bakış, 16* (Özel Sayı), 31-40.

Dağ, S. (2012). Akış Tipi Çizelgeleme Problemlerinin Sezgisel Yöntemlerle Optimizasyonu. Doktora Tezi, İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü, İstanbul, 319908.

Dannenbring, D. (1977). An Evaluation of Flow Shop Sequencing Heuristics. *Management Science, 23*(11), 1149-1259.

Dasgupta, P., Das, S. (2015). A discrete inter-species cuckoo search for flowshop scheduling problems. *Computers and Operations Research, 60*, 111-120.

Daya, M., Al-Fawzan, M. (1998). A tabu search approach for the flow shop scheduling problem. *European Journal of Operational Research, 109*(1), 88-95.

Doğru, A. S., Eren, T. (2020). Kablosuz Sensör Ağlarında Konum Belirlemede Parçacık Sürü Optimizasyonu, Yarasa Algoritması, Diferansiyel Gelişim Algoritması ve Ateşböceği Algoritması Yöntemlerinin Karşılaştırılması. *Uluslararası Mühendislik Araştırma ve Geliştirme Dergisi, 12*(3), 52-64.

Dokur, E., Yüzgeç, U., Kurban, M. (2019). Diferansiyel Gelişim Algoritma Tabanlı Uyarlamalı Sinirsel Bulanık Çıkarım Sistemi ile Kısa Dönem Rüzgar Hızı Kestirimi. *Bitlis Eren Üniversitesi Fen Bilimleri Dergisi,, 8*(3), 1057-1068.

Dong, X. Y., Huang, H. K., Chen, P. (2008). An Improved NEH Based Heuristic for The Permutation Flowshop Problem. *Computers and Operations Research, 35*, 3962-3968.

Eksioglu, B., Ekşioğlu, S. D., Jain, P. (2008). A tabu search algorithm for the flowshop scheduling problem with changing neighborhoods. *Computers & Industrial Engineering, 54*(1), 1-11.

Engin, O., Fıglalı, A. (2002). Genetik Algoritmalarla akış tipi çizelgelemede üreme yöntemi optimizasyonu. *İtü dergisi, 1*(1).

Famila, S., Jawahar, A., Sariga, A., Shankar, K. (2020). Improved artificial bee colony optimization based clustering algorithm for SMART sensor environments. *Peer-to-Peer Networking and Applications, 13*(4), 1071-1079.

Ghanavati, M., Wong, R., Fong, S., Gholamian, M. (2016). Extending the grenade explosion approach for effective clustering. *The 10th International Conference on Digital Information Management, ICDIM 2015* (s. 28-35). Jeju, Korea (South), 21-23 Oct. 2015.

González-Neir, E. M., García-Cáceres, R. G., Caballero-Villalobos, J. P., Molina-Sánchez, L. P., Montoya-Torres, J. R. (2016). Stochastic flexible flow shop scheduling problem under quantitative andqualitative decision criteria. *Computers & Industrial Engineering*, 128–144.

Göçmen, Ö., Tunçbilek, N., Esnaf, Ş. (2010). Permütasyon Akış Tipi Çizelgele Problemleri için CDS ve NEH Algoritmalarının Kıyaslanması ve Yeni bir Algoritma Önerisi. *10. Ulusal Üretim Araştırmaları Sempozyumu.* Girne, Kıbrıs (KKTC), 1-04 Ekim 2010 .

Grabowski, J., Wodecki, M. (2004). A very fast tabu search algorithm for the permutation flow shop problem with makespan criterion. *Computers and Operations Research, 31*(11), 1891-1909.

Gupta, J. (1971). A Functional Heuristic Algorithm for the Flowshop Scheduling Problem. *Journal of the Operational Research Society, 22*, 39-47.

Hefetz, N., Adiri, I. (1982). An efficient optimal algorithm for the two-machine unit-time jobshop schedule-length problem. *Mathematics of Operations Research, 7*(3), 54-360.

Heller, J. (1960). ome numerical experiments for an MxJ flow shop and its decision-. *Operations Research, 8*, 178-184.

Heris, M. K. (2015). *Differential Evolution (DE) in MATLAB*. 05 05, 2023 tarihinde Yarpiz: https://yarpiz.com/231/ypea107-differential-evolution adresinden alındı

Ho, J. C., Chang, Y. (1991). A New Heuristic For The n-Job, m-Machine Flowshop Problem. *European Journal of Operations Research,, 52*, 194-202.

Ho, J., Chang, Y. (1991). A new heuristic for the n-job, M-machine flow-shop problem. *European Journal of Operational Research, 52*(2), 194-202.

Ishibuchi, H., Misaki, S., Tanaka, H. (1995). Modified simulated annealing algorithms for the flow shop sequencing problem. *European Journal of Operational Research, 81*(2), 388-398.

İşler, M. C., Toklu, B., Çelik, V., Ersöz, S. (2009). Öğrenme Etkili Tam Zamanında Çizelgeleme Problemi ve KOBİ'de Uygulama. *International Journal of Engineering Research and Development, 1*(2), 29-33.

Iyer, S. K., Saxena , B. (2004). Improved genetic algorithm for the permutation flowshop scheduling problem. *Computers & Operations Research, 31*(4), 593-606.

Johnson, S. M. (1954). Optimal two- and three-stage production schedules with setup. *Naval Research Logistics Quarterly, 1*(1), 61-68.

Kaçar, T., Olgun, M. O. (2020). Bir Metal Fabrikası İçin İş Çizelgeleme Yazılımı. *BŞEÜ Fen Bilimleri Dergisi* , 421-443.

Kadı, D. (2020). Çok Amaçlı Hibrit Akış Tipi Çizelgeleme Probleminin Metasezgisel Yöntemle Çözülmesi ve Bir Tekstil İşletmesinde Uygulama. Doktora Tezi, Çukurova Üniversitesi, Sosyal Bilimler Enstitüsü. Adana, 643778.

Karaboğa, D. (2002). *Yapay Zeka Optimizasyon Algoritmaları.* Ankara: Atlas Yayın Dağıtım.

Karaköse, E. (2022). Sürü İnsansız Hava Araçlarının Görev Paylaşımı için Genetik Algoritma Tabanlı Bir. *Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 34*(1), 351 - 360.

Kaya, S., Karaçizmeli, İ. H., Aydilek, İ. B., Tenekeci, M. E., Gümüşçü, A. (2020, 2). Akış tipi çizelgeme problemlerinin hibrit ateşböceği ve parçacık sürü optimizasyonu algoritmasıyla çözümünde başlangıç popülasyonlarının etkileri. *Pamukkale Universitesi Mühendislik Bilimleri Dergisi, 26*(1), 140-149.

Keskintürk, T. (2006). Differansiyel Gelişim Algoritması . *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi, 1*(9), 85-99.

Khavari, F., Naseri, V., Naghshbandy, A. (2011). Optimal PMUs placement for power system observability using grenade explosion algorithm. *International Review of Electrical Engineering, 6*(3), 1332-1338.

Kızılay, D., Çil, Z. A. (2021). Adaptive Large Neighborhood Search Heuristic for Mixed Blocking Flowshop Scheduling Problem. *International Journal of Pure and Applied Sciences, 7*(2), 152-162.

Kundakçı, N. (2013). Üretim sistemlerinde dinamik iş çizelgeleme problemlerinin sezgisel yöntemlerle çözülmesi. Pamukkale Üniversitesi, Sosyal Bilimler Enstitüsü, Denizli, 332597.

Kurnaz, M. S., Kart, Ö. (2010). İş Akış Çizelgeleme Problemi Üzerinde NEH, FRB3 veFRB4 Sezgisellerinin Karşılaştırılması. *Akademik Bilişim'10 - XII. Akademik Bilişim Konferansı Bildirileri* (s. 625-630), Muğla, 10-12 Şubat.

Küçük, B. (2010). Özdeş Paralel Makineli Bir Üretim Sisteminin Karınca Koloni Algoritması ile Çizelgelenmesi. İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü, İstanbul, 261672.

Küpeli, İ., SARUCAN, A., Küpeli, A. S. (2020). Dağıtık Permütasyon Akış Tipi Çizelgeleme Problemlerinin Yapay Arı Koloni Algoritması İle Çözümü. *El-Cezeri, 7*(2), 549-562.

Li, X., Yin, M. (2012). A discrete artificial bee colony algorithm with composite mutation strategies for permutation flow shop scheduling problem. *Scientia Iranica, 19*(6), 1921-1935.

Li, Y., Pan, Q., Gao, K., Tasgetiren, M., Zhang, B., Li, J. (2021, 9). A green scheduling algorithm for the distributed flowshop problem. *Applied Soft Computing, 109*.

Lian, Z., Gu, X., Jiao, B. (2008). A novel particle swarm optimization algorithm for permutation flow-shop scheduling to minimize makespan. *Chaos, Solitons & Fractals, 35*(5), 851-861.

Liu, B., Wang, L., Jin, Y.-H. (2007). An effective PSO-based memetic algorithm for flow shop scheduling. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society, 37*(1), 18-27.

Liu, Y., Liu, S. (2013). A hybrid discrete artificial bee colony algorithm for permutation flowshop scheduling problem. *Applied Soft Computing Journal, 13*(3), 1459-1463.

Liu, Y., Yin, M., Gu, W. (2014). An effective differential evolution algorithm for permutation flow shop scheduling problem. *Applied Mathematics and Computation, 248*, 143-159.

Madenoğlu, F. (2020). Hibrit Akış Tipi Çizelgeleme Probleminin Parçacık Sürü Optimizasyon Algoritmasıyla Çözümüne Başlangıç Çözümlerin Etkisi. *Journal of Research in Business, 5*(2), 160-173.

Marouani, I., Boudjemline, A., Guesmi, T., Abdallah, H. (2018). A Modified Artificial Bee Colony for the Non-Smooth Dynamic Economic/Environmental Dispatch. *Engineering, Technology & Applied Science Research, 8*(5), 3321-3328.

Mattfeld, D. C., Vaessens, R.J.M., (2023). ORLİB, Brunel University London, http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/flowshop1.txt (Ziyaret tarihi: 24 Mayıs 2023)).

Mishra, S., Ray, P. (2016). Power Quality Improvement Using Photovoltaic Fed DSTATCOM Based on JAYA Optimization. *IEEE Transactions on Sustainable Energy, 7*(4), 1672-1680.

Mishra, S., Ray, P., Dash, S. (2016). A TLBO optimized photovoltaic fed DSTATCOM for power quality improvement. *1st IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems, ICPEICES 2016.* Delhi, India, 1 Haziran 2006

Mouwafi, M., El-Sehiemy, R., Abou El-Ela, A., Kinawy, A. (2016). Optimal placement of phasor measurement units with minimum availability of measuring channels in smart power systems. *Electric Power Systems Research, 141*, 421-431.

Murata, T., Ishibuchi, H., Tanaka, H. (1996). Genetic algorithms forflowshop scheduling problems. *Computers & Operations Research, 30*(4), 1061–1071.

Nahmias, S. (2001). Gantt Charts. S. Nahmias içinde, *Encyclopedia of Operations Research and Management Science* (s. 324-325). New York: Springer.

Nawaz, M., Enscore, E., Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega, 11*(1), 91-95.

Nearchou, A. (2004). A novel metaheuristic approach for the flow shop scheduling problem. *Engineering Applications of Artificial Intelligence*, *17*(3), 289-300.

Nearchou, A. (2004). The effect of various operators on the genetic search for large scheduling problems. *International Journal of Production Economics, 88*(2), 191-203.

Onwubolu, G., Davendra, D. (2006). Scheduling flow shops using differential evolution algorithm. *European Journal of Operational Research, 171*(2), 674-692.

Osman, I. H., Potts, C. N. (1989). Simulated annealing for permutation flow-shop scheduling. *Omega, 17*(6), 51-557.

Özalp, M., Alp, S. (2020). Uygun dağıtım rotası belirleme probleminde hibrit sezgisel bir yöntem önerisi. *Akıllı Ulaşım Sistemleri ve Uygulamaları Dergisi , 3*(1), 59 - 70.

Özcan, B. (2013). Akış Tipi Çizelgeleme Problemlerinin çözümü için Veri Madenciliği Tabanlı Bir Model Önerisi. Kocaeli Üniversitesi, Fen Bilimleri Enstitüsü, Kocaeli, 342568.

Palamutçuoğlu, B. (2022). Üretim Çizelgeleme Problemlerinde Yapay Zekâ Uygulamaları: Bir Derleme Çalışması. *3. Sektör Sosyal Ekonomi Dergisi, 57*(4), 3360-3379.

Pallantla, M., Singh, A. (2012). Grenade explosion method for maximum weight clique problem. *Communications in Computer and Information Science. 306*. 20-27.

Palmer, D. (1965). Sequencing Jobs Through a Multi-Stage Process in the Minimum Total Time—A Quick Method of Obtaining a Near Optimum. *Operational Research Quarterly, 16*(1), 101-107.

Pan, Q., Fatih Tasgetiren, M., Liang, Y. (2008b). A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem. *Computers and Operations Research, 35*(9), 2807-2839.

Pan, Q., Fatih Tasgetiren, M., Suganthan, P., Chua, T. (2011). A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. *Information Sciences, 181*(12), 2455-2468.

Pan, Q., Tasgetiren, M., Liang, Y. (2008a). A discrete differential evolution algorithm for the permutation flowshop scheduling problem. *Computers and Industrial Engineering, 55*(4), 795-816.

Pinedo, M. (2008). *Theory, Algorithms, and Systems* (3 b.). New Jersey: Printice-Hall.

Rajakumar, S., Arunachalam, V., Selladurai, V. (2004). Workflow balancing strategies in parallel machine scheduling. *International Journal of advanced Manufacturing Technology*(24), 366-374.

Rajendran, C. (1995). Heuristics for scheduling in flowshop with multiple objectives. *European Journal of Operational Research, 82*(3), 540-555.

Rajendran, C., Ziegler, H. (2004). Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs. *European Journal of Operational Research, 155*(2), 426-438.

Rakhade, R., Patil, N., Panchbhai, M. (2017). Application of Grenade Explosion Method Optimization for Plate-Fin Type Heat Exchanger (PFHE). *IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE) e-ISSN*, 17-18.

Ramanan, T. R., Iqbal, M., Umarali, K. (2014). A particle swarm optimization approach for permutation flow shop scheduling problem. *International Journal for Simulation and Multidisciplinary Design Optimization, 5*(1), 1-6.

Rao, R., More, K. (2015). Optimal design of the heat pipe using TLBO (teaching-learning-based optimization) algorithm. *Energy, 80*, 535-544.

Rao, R., Savsani, V., Vakharia, D. (2012). Teaching-Learning-Based Optimization: An optimization method for continuous non-linear large scale problems. *Information Sciences, 183*(1), 1-15.

Ravindran, D., Selvakumar, S. J., Sivaraman, R., Haq, A. (2005). Flow Shop Scheduling with Multiple Objective of Minimizing Makespan and Total Flowtime. *The International Journal of Advanced manufacturing Technology, 25*(9-10), 1007-1012.

Reevest, C. (1995). A Genetıc Algorıthm For Flowshop Sequencıng. *Computers Ops Res, 22*(1), 5-13.

Ruiz, R., Maroto, C. (2005). A comprehensive review and evaluation of permutation flowshop heuristics. *European Journal of Operational Research, 165*(2), 479-494.

Ruiz, R., Maroto, C., Alcaraz, J. (2006). Two new robust genetic algorithms for the flowshop scheduling problem. *Omega, 34*(5), 461-476.

Sağir, M., Saraç, T. (2007). Hazırlık zamanı kaynaklı üretim kayıplarını en azlayan çok amaçlı bir çizelgeleme algoritması. *Endüstri Mühendisliği Dergisi, 18*, 14.

Salhi, A., Salhi, A., Naimi, D., Bouktir, T. (2016). Optimal power flow resolution using artificial bee colony algorithm based grenade explosion method. *J. Electrical Systems, 12*(4), 734-756.

Stützle , T. (1998). An ant approach to the flow shop Problem. *Proceedings, 6th European Cong-ress on Intelligent Techniques and Soft Com-puting (EUFIT' 98),*, (s. 1560-1564). Aachen, Germany, 7 -10 September 1998.

Surender Reddy. (2016). Congestion Management Using Multi-Objective Grenade Explosion Method. *Wseas Transactions On Power Systems, 11,* 81-89.

Szwarc, W. (1960). Solution of the Akers-Friedman scheduling problem. *Operations Research, 8*(6), 782-788.

Şevkli, M., Yenisey, M. M. (2006). Atölye tipi çizelgeleme problemleri için parçacık sürü optimizasyonu yöntemi. *İTÜ Dergisi, 5*(2), 58-68.

Taillard, E.,D., (1990). Some Efficient Heuristic Methods for The Flowshop Sequencing Problem . *European Journal of Operational Research 47(1),* 65-74.

Taillard, E.,D., (1993). Some Efficient Heuristic Methods for The Flowshop Sequencing Problem . *European Journal of Operational Research 64(2),* 278-285.

Taillard, E.,D., (2023). *Some Efficient Heuristic Methods for The Flowshop*. http://mistic.heig-vd.ch/taillard/problemes.dir/ordonnancement.dir/flowshop.dir/best_lb_up.txt. (Ziyaret tarihi: 24 Mayıs 2023)

Tarłowski, D. (2014). Nonautonomous stochastic search for global minimum in continuous optimization. *Journal of Mathematical Analysis and Applications, 412*(2), 631-645.

Tasgetiren, M., Liang, Y.-C., Sevkli, M., Gencyilmaz, G. (2007). A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. *European Journal of Operational Research, 177*(3), 1930-1947.

Tasgetiren, M., Pan, Q., Suganthan, P., Liang, Y. (2007). A discrete differential evolution algorithm for the no-wait flowshop scheduling problem with total flowtime criterion. *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Scheduling, CI-Sched 2007* (s. 251-258). Honolulu, Hawaii, USA, 1-5 April 2007

Tasgetiren, M., Pan, Q.-K., Suganthan, P., Chen, A.-L. (2011). A discrete artificial bee colony algorithm for the total flowtime minimization in permutation flow shops. *Information Sciences, 181*(16), 3459-3475.

Terzi, Ü. (2009). Gezgin Satıcı Problemi İçin Diferansiyel Gelişim Algoritması Tabanlı Bir Metasezgisel Önerisi. Doktora Tezi, Kocaeli Üniversitesi, Fen Bilimleri Enstitüsü, Kocaeli, 262170

Tosun, Ö. (2012). Yapay Arı Kolonisi Algoritması ve Permütasyon Akış Tipi Çizelgeleme Problemine Uygulanması. Doktor Tezi, Akdeniz Üniversitesi, Sosyal Bilimler Enstitüsü, Antalya, 325150.

Tseng, L., Lin, Y. (2009). A hybrid genetic local search algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research, 198*(1), 84-92.

Widmer, M., Hertz, A. (1989). A new heuristic method for the flow shop sequencing problem. *European Journal of Operational Research, 41*(2), 186-193.

Xie, Z., Zhang, C., Shao, X., Lin, W., Zhu, H. (2014). An effective hybrid teaching–learning-based optimization algorithmfor permutation flow shop scheduling problem. *Advances in Engineering Software, 77*, 35-47.

Yağmahan, B., Yenisey, M. M. (2006). Akış Tipi Çizelgeleme Problemi İçin KKE Parametre Eniyileme. *İTÜ Dergisi, 5*(2), 133-141.

Yağmahan, B., Yenisey, M. M. (2008). Ant colony optimization for multi-objective flowshop scheduling problem. *Computers & Industrial Engineering, 54*(3), 411–420.

Ying, K.-C. (2008). Solving non-permutation flowshop scheduling problems. *International Journal of Advanced Manufacturing Technology, 38*(3), 348-354.

Ying, K.-C., Liao, C.-J. (2004). An ant colony system for permutation flow-shop sequencing. *Computers & Operations Research, 31*(5), 791-801.

Yu, Y., Zhang, F., Yang, G., Wang, Y., Huang, J., Han, Y. (2022). A discrete artificial bee colony method based on variable neighborhood structures for the distributed permutation flowshop problem with sequence-dependent setup times. *Swarm and Evolutionary Computation, 75*.

Yücenur, G. N., Demirel, N. Ç., Ceylan, C., Demirel, T. (2011). Hizmet Değerinin Müşterilerin Davranışsal Niyetleri Üzerindeki Etkisinin Yapısal Eşitlik Modeli ile Ölçülmesi. *Doğuş Üniversitesi Dergisi, 12*(1), 156-168.

Zhang, C., Sun, J., Zhu, X., Yang, Q. (2008). An improved particle swarm optimization algorithm for flowshop scheduling problem. *Information Processing Letters, 108*(4), 204-209.

Zhang, C., Zheng, J., Zhou, Y. (2015). Two modified Artificial Bee Colony algorithms inspired by Grenade Explosion Method. *Neurocomputing, 151*(P3), 1198-1207.

Zheng, J.-G., Zhang, C.-Q., Zhou, Y.-Q. (2015). Artificial Bee Colony Algorithm Combined with Grenade Explosion Method and Cauchy Operator for Global Optimization. *Hindawi Publishing Corporation Mathematical Problems in Engineering*, 1-15.

Zobolas, G. İ., Tarantilis, C. D., Ioannou, G. (2009). Minimizing makespan in permutation flow shop schedulingproblems using a hybrid metaheuristic algorithm. *Computers & Operations Research, 36,* 1249 – 1267.

EKLER

Ek-A

Tablo A. 1. Ajan Bölgesinin ve Optimal Arama Yönünün Etkisinin Değerlendirilmesine Yönelik Test Sonuçlarına Ait Tanımlayıcı İstatistikler

P. No	İstatistik	EBPM1		EBPM2		EBPM3		EBPM4	
		Makespan	Δ	Makespan	Δ	Makespan	Δ	Makespan	Δ
1	Ortalama	1295,45	1,36541471	1296,40	1,43974961	1305,20	2,12832551	1298,60	1,61189358
	N	20	20	20	20	20	20	20	20
	S. Sapma	4,605	0,360344523	2,683	0,209959435	9,661	0,755911943	4,489	0,351219249
	Minimum	1278	0,000000	1285	0,547731	1297	1,486698	1295	1,330203
	Maksimum	1297	1,486698	1297	1,486698	1328	3,912363	1312	2,660407
2	Ortalama	1365,40	0,47093451	1363,95	0,36423841	1373,80	1,08903606	1370,35	0,83517292
	N	20	20	20	20	20	20	20	20
	S. Sapma	2,113	0,155453917	2,874	0,211487354	6,818	0,501687492	5,393	0,396816328
	Minimum	1359	0,000000	1359	0,000000	1365	0,441501	1365	0,441501
	Maksimum	1368	0,662252	1366	0,515085	1383	1,766004	1383	1,766004
3	Ortalama	1095,80	1,36910268	1102,05	1,94727105	1143,80	5,80943571	1134,70	4,96762257
	N	20	20	20	20	20	20	20	20
	S. Sapma	3,847	0,355881296	4,872	0,450673167	18,312	1,693979400	12,782	1,182421595
	Minimum	1089	0,740056	1098	1,572618	1098	1,572618	1103	2,035153
	Maksimum	1100	1,757632	1115	3,145236	1163	7,585569	1162	7,493062
4	Ortalama	1307,50	1,12142305	1317,25	1,87548337	1367,50	5,76179428	1362,70	5,39056458
	N	20	20	20	20	20	20	20	20
	S. Sapma	4,224	0,326681487	8,441	0,652820689	17,635	1,363897300	14,521	1,123029043
	Minimum	1299	0,464037	1308	1,160093	1338	3,480278	1325	2,474865
	Maksimum	1316	1,778809	1331	2,938902	1402	8,430008	1381	6,805878
5	Ortalama	1247,15	0,98380567	1249,85	1,20242915	1274,25	3,17813765	1273,05	3,08097166
	N	20	20	20	20	20	20	20	20
	S. Sapma	3,249	0,263067729	3,897	0,315548980	9,862	0,798505382	11,637	0,942262517
	Minimum	1243	0,647773	1243	0,647773	1253	1,457490	1252	1,376518
	Maksimum	1250	1,214575	1262	2,186235	1291	4,534413	1297	5,020243
6	Ortalama	1210,50	1,29707113	1213,15	1,51882845	1245,20	4,20083682	1234,05	3,26778243
	N	20	20	20	20	20	20	20	20
	S. Sapma	5,717	0,478410931	5,441	0,455340058	18,931	1,584175672	12,446	1,041469932
	Minimum	1195	0,000000	1210	1,255230	1210	1,255230	1211	1,338912
	Maksimum	1224	2,426778	1224	2,426778	1279	7,029289	1268	6,108787
7	Ortalama	1249,80	0,87167070	1252,05	1,05326877	1262,60	1,90476190	1262,75	1,91686844
	N	20	20	20	20	20	20	20	20
	S. Sapma	3,694	0,298105090	2,585	0,208625911	8,714	0,703323227	9,430	0,761137502
	Minimum	1239	0,000000	1251	0,968523	1251	0,968523	1251	0,968523
	Maksimum	1251	0,968523	1259	1,614205	1277	3,066990	1279	3,228410
8	Ortalama	1217,65	0,96600332	1224,70	1,55058043	1268,00	5,14096186	1255,75	4,12520730
	N	20	20	20	20	20	20	20	20
	S. Sapma	5,724	0,474638552	8,992	0,745589481	18,373	1,523492672	15,061	1,248825997
	Minimum	1206	0,000000	1212	0,497512	1230	1,990050	1233	2,238806
	Maksimum	1228	1,824212	1243	3,067993	1293	7,213930	1283	6,384743
9	Ortalama	1253,35	1,89837398	1257,55	2,23983740	1279,85	4,05284553	1278,55	3,94715447
	N	20	20	20	20	20	20	20	20
	S. Sapma	5,102	0,414785142	6,549	0,532455810	10,609	0,862537116	14,905	1,211778358
	Minimum	1237	0,569106	1250	1,626016	1265	2,845528	1255	2,032520
	Maksimum	1261	2,520325	1271	3,333333	1303	5,934959	1311	6,585366
10	Ortalama	1126,30	1,65162455	1129,25	1,91787004	1162,80	4,94584838	1162,65	4,93231047
	N	20	20	20	20	20	20	20	20
	S. Sapma	6,182	0,557970915	7,369	0,665075084	12,776	1,153050496	15,672	1,414429810
	Minimum	1108	0,000000	1111	0,270758	1136	2,527076	1128	1,805054
	Maksimum	1141	2,978339	1148	3,610108	1191	7,490975	1187	7,129964
11	Ortalama	1618,20	2,28824273	1633,10	3,23008850	1696,15	7,21554994	1683,60	6,42225032
	N	20	20	20	20	20	20	20	20
	S. Sapma	9,082	0,574088612	9,408	0,594708156	13,468	0,851351709	27,856	1,760786531
	Minimum	1594	0,758534	1610	1,769912	1666	5,309735	1644	3,919090
	Maksimum	1633	3,223767	1644	3,919090	1718	8,596713	1738	9,860936

Tablo A.1. (Devam) Ajan Bölgesinin ve Optimal Arama Yönünün Etkisinin Değerlendirilmesine Yönelik Test Sonuçlarına Ait Tanımlayıcı İstatistikler

P. No	İstatistik	EBPM1		EBPM2		EBPM3		EBPM4	
		Makespan	Δ	Makespan	Δ	Makespan	Δ	Makespan	Δ
12	Ortalama	1702,90	2,64617239	1720,45	3,70403858	1775,30	7,01024714	1769,65	6,66968053
	N	20	20	20	20	20	20	20	20
	S. Sapma	12,998	0,783458145	13,093	0,789191428	20,131	1,213466570	22,926	1,381925255
	Minimum	1678	1,145268	1699	2,411091	1743	5,063291	1717	3,496082
	Maksimum	1737	4,701627	1747	5,304400	1807	8,921037	1805	8,800482
13	Ortalama	1538,35	2,83088235	1553,75	3,86029412	1600,30	6,97192513	1603,65	7,19585561
	N	20	20	20	20	20	20	20	20
	S. Sapma	12,533	0,837782020	10,197	0,681643976	20,522	1,371817399	17,218	1,150917341
	Minimum	1509	0,868984	1535	2,606952	1566	4,679144	1570	4,946524
	Maksimum	1561	4,344920	1576	5,347594	1633	9,157754	1639	9,558824
14	Ortalama	1408,45	2,28395062	1425,95	3,55482934	1482,90	7,69063181	1486,20	7,93028322
	N	20	20	20	20	20	20	20	20
	S. Sapma	12,420	0,901972848	11,100	0,806093331	17,607	1,278614229	19,718	1,431954399
	Minimum	1391	1,016703	1402	1,815541	1436	4,284677	1454	5,591866
	Maksimum	1429	3,776325	1451	5,374001	1516	10,094408	1520	10,384895
15	Ortalama	1456,00	2,60747005	1477,35	4,11205074	1526,40	7,56871036	1535,05	8,17829457
	N	20	20	20	20	20	20	20	20
	S. Sapma	14,194	1,000291938	10,028	0,706675456	21,256	1,497979586	21,214	1,495023391
	Minimum	1430	0,775194	1455	2,536998	1494	5,285412	1499	5,637773
	Maksimum	1482	4,439746	1491	5,073996	1567	10,429880	1564	10,218464
16	Ortalama	1425,75	2,05798139	1437,80	2,92054402	1495,70	7,06513958	1487,55	6,48174660
	N	20	20	20	20	20	20	20	20
	S. Sapma	13,070	0,935587318	8,501	0,608546390	15,698	1,123703604	20,730	1,483896321
	Minimum	1404	0,501074	1420	1,646385	1468	5,082319	1452	3,937008
	Maksimum	1457	4,294918	1455	4,151754	1529	9,448819	1525	9,162491
17	Ortalama	1508,90	1,67789757	1523,75	2,67857143	1577,35	6,29043127	1573,35	6,02088949
	N	20	20	20	20	20	20	20	20
	S. Sapma	13,218	0,890728317	13,591	0,915856411	22,281	1,501425738	17,854	1,203100624
	Minimum	1493	0,606469	1496	0,808625	1544	4,043127	1544	4,043127
	Maksimum	1529	3,032345	1543	3,975741	1629	9,770889	1609	8,423181
18	Ortalama	1586,40	3,14694408	1598,00	3,90117035	1653,80	7,52925878	1649,10	7,22366710
	N	20	20	20	20	20	20	20	20
	S. Sapma	14,288	0,928999665	9,453	0,614661069	24,289	1,579261955	22,429	1,458296500
	Minimum	1566	1,820546	1581	2,795839	1591	3,446034	1600	4,031209
	Maksimum	1611	4,746424	1618	5,201560	1694	10,143043	1683	9,427828
19	Ortalama	1630,20	2,33521657	1638,55	2,85938481	1676,00	5,21029504	1678,05	5,33898305
	N	20	20	20	20	20	20	20	20
	S. Sapma	10,253	0,643603211	8,029	0,504041859	14,430	0,905806776	15,869	0,996199056
	Minimum	1615	1,381042	1621	1,757690	1642	3,075957	1636	2,699309
	Maksimum	1658	4,080352	1653	3,766478	1694	6,340239	1708	7,219083
20	Ortalama	1631,40	2,53928347	1640,95	3,13953488	1691,70	6,32935261	1691,40	6,31049654
	N	20	20	20	20	20	20	20	20
	S. Sapma	6,878	0,432299319	11,812	0,742427052	22,714	1,427627153	20,760	1,304857698
	Minimum	1617	1,634192	1619	1,759899	1640	3,079824	1660	4,336895
	Maksimum	1640	3,079824	1661	4,399749	1722	8,233815	1735	9,050911
21	Ortalama	2344,05	2,04832390	2368,55	3,11493252	2423,30	5,49847627	2426,30	5,62908141
	N	20	20	20	20	20	20	20	20
	S. Sapma	14,413	0,627470072	17,136	0,745999333	18,723	0,815090516	23,862	1,038818577
	Minimum	2314	0,740096	2345	2,089682	2397	4,353505	2371	3,221593
	Maksimum	2368	3,090988	2404	4,658250	2460	7,096212	2465	7,313888
22	Ortalama	2140,35	1,96998571	2158,00	2,81086232	2229,05	6,19580753	2220,10	5,76941401
	N	20	20	20	20	20	20	20	20
	S. Sapma	13,160	0,626967470	16,003	0,762424447	29,830	1,421161374	26,425	1,258955035
	Minimum	2117	0,857551	2123	1,143402	2175	3,620772	2177	3,716055
	Maksimum	2160	2,906146	2191	4,383040	2275	8,384945	2268	8,051453
23	Ortalama	2377,70	2,22269991	2399,95	3,17927773	2450,85	5,36758383	2447,80	5,23645744
	N	20	20	20	20	20	20	20	20
	S. Sapma	16,591	0,713300679	14,788	0,635764670	23,006	0,989094980	25,249	1,085532757
	Minimum	2335	0,386930	2367	1,762683	2387	2,622528	2399	3,138435
	Maksimum	2400	3,181427	2420	4,041273	2478	6,534824	2499	7,437661
24	Ortalama	2259,55	1,64417454	2287,20	2,88798920	2361,55	6,23256860	2353,95	5,89068826
	N	20	20	20	20	20	20	20	20

Tablo A.1. (Devam) Ajan Bölgesinin ve Optimal Arama Yönünün Etkisinin Değerlendirilmesine Yönelik Test Sonuçlarına Ait Tanımlayıcı İstatistikler

P. No	İstatistik	EBPM1		EBPM2		EBPM3		EBPM4	
		Makespan	Δ	Makespan	Δ	Makespan	Δ	Makespan	Δ
	S. Sapma	13,185	0,593109304	13,081	0,588444598	17,040	0,766537244	28,962	1,302825775
	Minimum	2242	0,854701	2270	2,114260	2333	4,948268	2300	3,463788
	Maksimum	2294	3,193882	2323	4,498426	2393	7,647323	2408	8,322087
	Ortalama	2334,75	1,90964644	2353,10	2,71060672	2439,40	6,47752073	2425,60	5,87516368
	N	20	20	20	20	20	20	20	20
25	S. Sapma	7,772	0,339251488	10,503	0,458429998	26,470	1,155401200	24,797	1,082359620
	Minimum	2322	1,353121	2337	2,007857	2376	3,710170	2388	4,233959
	Maksimum	2345	2,357049	2371	3,491925	2474	7,987778	2470	7,813182
	Ortalama	2266,90	1,83737646	2286,00	2,69541779	2348,40	5,49865229	2347,90	5,47619048
	N	20	20	20	20	20	20	20	20
26	S. Sapma	13,947	0,626545002	12,209	0,548459627	30,457	1,368233244	16,367	0,735272627
	Minimum	2239	0,584007	2256	1,347709	2299	3,279425	2314	3,953279
	Maksimum	2291	2,920036	2314	3,953279	2405	8,041330	2384	7,097934
	Ortalama	2323,55	2,22393313	2342,55	3,05983282	2399,85	5,58073031	2395,45	5,38715354
	N	20	20	20	20	20	20	20	20
27	S. Sapma	12,267	0,539668468	10,298	0,453060188	17,190	0,756277606	15,521	0,682828828
	Minimum	2302	1,275847	2321	2,111747	2358	3,739551	2363	3,959525
	Maksimum	2350	3,387593	2366	4,091509	2424	6,643203	2420	6,467224
	Ortalama	2242,10	1,91363636	2262,85	2,85681818	2329,75	5,89772727	2326,60	5,75454545
	N	20	20	20	20	20	20	20	20
28	S. Sapma	14,987	0,681243777	12,449	0,565861290	18,674	0,848824764	26,767	1,216674065
	Minimum	2212	0,545455	2238	1,727273	2298	4,454545	2296	4,363636
	Maksimum	2272	3,272727	2288	4,000000	2362	7,363636	2394	8,818182
	Ortalama	2281,50	1,98927135	2320,65	3,73938310	2390,05	6,84175235	2392,90	6,96915512
	N	20	20	20	20	20	20	20	20
29	S. Sapma	16,324	0,729728283	16,728	0,747777110	26,514	1,185252460	29,073	1,299652605
	Minimum	2250	0,581135	2279	1,877515	2353	5,185516	2346	4,872597
	Maksimum	2305	3,039785	2345	4,827895	2441	9,119356	2440	9,074654
	Ortalama	2232,20	2,48852158	2252,15	3,40449954	2342,30	7,54361800	2328,35	6,90312213
	N	20	20	20	20	20	20	20	20
30	S. Sapma	20,955	0,962125336	23,424	1,075459307	38,412	1,763637838	27,132	1,245720540
	Minimum	2202	1,101928	2199	0,964187	2267	4,086318	2277	4,545455
	Maksimum	2268	4,132231	2291	5,188246	2402	10,284665	2386	9,550046
	Ortalama	2734,40	0,38179148	2739,55	0,57085169	2764,20	1,47577093	2761,55	1,37848752
	N	20	20	20	20	20	20	20	20
31	S. Sapma	4,914	0,180396373	7,911	0,290400856	15,653	0,574625994	15,188	0,557569943
	Minimum	2724	0,000000	2729	0,183554	2741	0,624082	2731	0,256975
	Maksimum	2746	0,807636	2752	1,027900	2799	2,753304	2782	2,129222
	Ortalama	2865,70	1,11856034	2878,50	1,57021877	2934,15	3,53387438	2931,80	3,45095272
	N	20	20	20	20	20	20	20	20
32	S. Sapma	11,272	0,397749806	12,250	0,432237022	27,396	0,966699494	20,310	0,716644576
	Minimum	2838	0,141143	2848	0,494001	2890	1,976006	2875	1,446718
	Maksimum	2884	1,764291	2898	2,258292	2983	5,257586	2961	4,481299
	Ortalama	2640,20	0,73254483	2655,50	1,31629149	2700,05	3,01602442	2696,30	2,87294926
	N	20	20	20	20	20	20	20	20
33	S. Sapma	8,823	0,336643030	7,884	0,300802283	12,931	0,493357084	17,541	0,669257140
	Minimum	2621	0,000000	2644	0,877528	2679	2,212896	2667	1,755055
	Maksimum	2652	1,182755	2671	1,907669	2727	4,044258	2732	4,235025
	Ortalama	2774,00	0,83605961	2790,85	1,44856416	2853,85	3,73864049	2846,80	3,48237005
	N	20	20	20	20	20	20	20	20
34	S. Sapma	7,766	0,282309126	12,244	0,445086614	23,415	0,851127378	18,981	0,689962806
	Minimum	2753	0,072701	2781	1,090513	2794	1,563068	2805	1,962923
	Maksimum	2782	1,126863	2818	2,435478	2885	4,870956	2876	4,543802
	Ortalama	2867,30	0,15019211	2883,10	0,70206078	2914,95	1,81453021	2911,15	1,68180231
	N	20	20	20	20	20	20	20	20
35	S. Sapma	7,781	0,271761877	12,320	0,430312954	14,028	0,489978112	15,618	0,545513669
	Minimum	2863	0,000000	2864	0,034928	2890	0,943067	2866	0,104785
	Maksimum	2890	0,943067	2901	1,327279	2942	2,759343	2932	2,410059
	Ortalama	2840,50	0,40650407	2863,80	1,23011665	2910,75	2,88971368	2913,75	2,99575822
	N	20	20	20	20	20	20	20	20
36	S. Sapma	9,774	0,345484499	11,372	0,401985609	17,281	0,610837020	20,442	0,722592860
	Minimum	2832	0,106045	2837	0,282785	2887	2,050194	2874	1,590668

Tablo A.1. (Devam) Ajan Bölgesinin ve Optimal Arama Yönünün Etkisinin Değerlendirilmesine Yönelik Test Sonuçlarına Ait Tanımlayıcı İstatistikler

P. No	İstatistik	EBPM1		EBPM2		EBPM3		EBPM4	
		Makespan	Δ	Makespan	Δ	Makespan	Δ	Makespan	Δ
	Maksimum	2873	1,555320	2881	1,838105	2950	4,277130	2959	4,595263
	Ortalama	2746,60	0,79266055	2777,80	1,93761468	2835,40	4,05137615	2825,45	3,68623853
	N	20	20	20	20	20	20	20	20
37	S. Sapma	10,565	0,387709596	15,730	0,577246165	15,125	0,555062033	16,159	0,592978733
	Minimum	2732	0,256881	2756	1,137615	2809	3,082569	2799	2,715596
	Maksimum	2771	1,688073	2809	3,082569	2863	5,064220	2858	4,880734
	Ortalama	2706,25	0,86656728	2727,00	1,63995527	2781,50	3,67126351	2770,35	3,25568394
	N	20	20	20	20	20	20	20	20
38	S. Sapma	2,425	0,090391229	12,691	0,473002286	20,811	0,775668327	16,037	0,597728137
	Minimum	2700	0,633619	2710	1,006336	2743	2,236303	2750	2,497205
	Maksimum	2712	1,080880	2762	2,944465	2815	4,919866	2797	4,248975
	Ortalama	2584,95	1,29114420	2599,30	1,85344828	2644,10	3,60893417	2640,35	3,46199060
	N	20	20	20	20	20	20	20	20
39	S. Sapma	11,043	0,432713914	8,921	0,349580457	16,160	0,633230984	15,584	0,610670875
	Minimum	2568	0,626959	2583	1,214734	2618	2,586207	2611	2,311912
	Maksimum	2599	1,841693	2613	2,390282	2676	4,858934	2668	4,545455
	Ortalama	2784,00	0,07189073	2792,80	0,38820992	2843,30	2,20345075	2823,95	1,50790798
	N	20	20	20	20	20	20	20	20
40	S. Sapma	1,806	0,064932469	10,309	0,370557615	19,564	0,703232577	18,813	0,676255178
	Minimum	2782	0,000000	2782	0,000000	2805	0,826743	2792	0,359454
	Maksimum	2789	0,251618	2824	1,509705	2872	3,235083	2858	2,731848
	Ortalama	3136,95	4,87963892	3250,90	8,68940154	3357,40	12,25008358	3334,45	11,48278168
	N	20	20	20	20	20	20	20	20
41	S. Sapma	26,104	0,872751335	18,688	0,624817854	37,234	1,244862163	40,889	1,367081941
	Minimum	3077	2,875293	3214	7,455700	3290	9,996657	3238	8,258108
	Maksimum	3182	6,385824	3284	9,796055	3407	13,908392	3412	14,075560
	Ortalama	3011,75	5,04883153	3132,25	9,25183118	3222,05	12,38402511	3218,25	12,25148239
	N	20	20	20	20	20	20	20	20
42	S. Sapma	17,075	0,595580664	12,043	0,420063953	26,951	0,940047931	37,219	1,298184401
	Minimum	2982	4,011161	3100	8,126962	3150	9,870945	3106	8,336240
	Maksimum	3044	6,173701	3153	9,975584	3260	13,707708	3271	14,091385
	Ortalama	2990,10	5,32229658	3126,75	10,13561113	3246,70	14,36069038	3216,45	13,29517436
	N	20	20	20	20	20	20	20	20
43	S. Sapma	29,129	1,026040661	21,285	0,749726099	29,921	1,053931820	32,650	1,150067669
	Minimum	2947	3,804156	3095	9,017260	3165	11,482917	3153	11,060232
	Maksimum	3064	7,925326	3170	11,659035	3288	15,815428	3255	14,653047
	Ortalama	3161,60	3,21906627	3271,45	6,80541952	3369,35	10,00163239	3357,45	9,61312439
	N	20	20	20	20	20	20	20	20
44	S. Sapma	35,088	1,145558591	24,401	0,796643719	30,193	0,985727884	30,122	0,983403179
	Minimum	3110	1,534443	3222	5,190989	3307	7,966046	3306	7,933399
	Maksimum	3229	5,419523	3323	8,488410	3413	11,426706	3412	11,394058
	Ortalama	3118,30	4,78158602	3238,25	8,81216398	3338,40	12,17741935	3325,10	11,73051075
	N	20	20	20	20	20	20	20	20
45	S. Sapma	31,553	1,060247412	19,023	0,639219936	35,798	1,202898695	34,400	1,155912950
	Minimum	3071	3,192204	3209	7,829301	3264	9,677419	3262	9,610215
	Maksimum	3180	6,854839	3273	9,979839	3379	13,541667	3379	13,541667
	Ortalama	3138,05	4,39288090	3228,90	7,41516966	3339,60	11,09780439	3308,80	10,07318696
	N	20	20	20	20	20	20	20	20
46	S. Sapma	19,077	0,634641906	23,943	0,796496831	25,853	0,860033890	25,808	0,858556202
	Minimum	3096	2,994012	3173	5,555556	3269	8,749168	3263	8,549568
	Maksimum	3180	5,788423	3270	8,782435	3397	13,007319	3351	11,477046
	Ortalama	3232,70	4,51665050	3301,85	6,75234400	3387,80	9,53119948	3389,10	9,57322987
	N	20	20	20	20	20	20	20	20
47	S. Sapma	30,065	0,972034183	24,816	0,802317782	41,817	1,352001214	32,919	1,064312943
	Minimum	3177	2,715810	3236	4,623343	3283	6,142903	3330	7,662464
	Maksimum	3289	6,336890	3345	8,147430	3473	12,285807	3451	11,574523
	Ortalama	3135,60	3,24662496	3231,90	6,41751729	3331,75	9,70530128	3315,05	9,15541653
	N	20	20	20	20	20	20	20	20
48	S. Sapma	28,940	0,952909617	25,955	0,854633671	28,859	0,950239576	31,602	1,040563835
	Minimum	3065	0,921962	3180	4,708594	3270	7,672045	3257	7,243991
	Maksimum	3200	5,367139	3273	7,770826	3371	10,997695	3375	11,129404
49	Ortalama	3027,25	4,49603038	3122,10	7,77010701	3245,35	12,02450811	3227,95	11,42388678

Tablo A.1. (Devam) Ajan Bölgesinin ve Optimal Arama Yönünün Etkisinin Değerlendirilmesine Yönelik Test Sonuçlarına Ait Tanımlayıcı İstatistikler

P. No	İstatistik	EBPM1		EBPM2		EBPM3		EBPM4	
		Makespan	Δ	Makespan	Δ	Makespan	Δ	Makespan	Δ
	N	20	20	20	20	20	20	20	20
	S. Sapma	12,126	0,418570363	23,344	0,805795041	31,859	1,099713253	37,424	1,291828596
	Minimum	3000	3,555402	3064	5,764584	3170	9,423542	3135	8,215395
	Maksimum	3061	5,661029	3157	8,974802	3292	13,634795	3305	14,083535
	Ortalama	3204,20	4,54159869	3287,05	7,24469821	3393,75	10,72593801	3372,10	10,01957586
	N	20	20	20	20	20	20	20	20
50	S. Sapma	17,025	0,555466849	21,130	0,689391897	39,715	1,295745738	38,765	1,264763917
	Minimum	3161	3,132137	3250	6,035889	3269	6,655791	3299	7,634584
	Maksimum	3241	5,742251	3330	8,646003	3461	12,920065	3457	12,789560
	Ortalama	4037,70	4,87532468	4218,60	9,57402597	4377,10	13,69090909	4354,00	13,09090909
	N	20	20	20	20	20	20	20	20
51	S. Sapma	25,483	0,661893891	33,480	0,869600231	51,328	1,333182712	40,778	1,059167819
	Minimum	3986	3,532468	4125	7,142857	4255	10,519481	4267	10,831169
	Maksimum	4096	6,389610	4264	10,753247	4446	15,480519	4419	14,779221
	Ortalama	3913,35	5,65199784	4079,05	10,12553996	4202,05	13,44627430	4189,55	13,10880130
	N	20	20	20	20	20	20	20	20
52	S. Sapma	40,344	1,089190678	34,264	0,925065146	33,138	0,894665923	32,093	0,866434992
	Minimum	3836	3,563715	3989	7,694384	4133	11,582073	4124	11,339093
	Maksimum	3982	7,505400	4130	11,501080	4258	14,956803	4244	14,578834
	Ortalama	3863,00	6,12637363	4036,25	10,88598901	4161,70	14,33241758	4164,55	14,41071429
	N	20	20	20	20	20	20	20	20
53	S. Sapma	27,290	0,749721621	34,267	0,941389105	45,545	1,251230018	39,630	1,088749926
	Minimum	3807	4,587912	3972	9,120879	4076	11,978022	4097	12,554945
	Maksimum	3923	7,774725	4094	12,472527	4224	16,043956	4239	16,456044
	Ortalama	3924,30	5,49193548	4100,45	10,22715054	4224,05	13,54973118	4225,50	13,58870968
	N	20	20	20	20	20	20	20	20
54	S. Sapma	48,872	1,313752219	25,176	0,676778730	30,036	0,807416655	41,485	1,115186479
	Minimum	3863	3,844086	4054	8,978495	4146	11,451613	4152	11,612903
	Maksimum	4061	9,166667	4153	11,639785	4264	14,623656	4301	15,618280
	Ortalama	3832,15	6,15373961	4053,05	12,27285319	4181,00	15,81717452	4157,35	15,16204986
	N	20	20	20	20	20	20	20	20
55	S. Sapma	38,130	1,056242416	37,436	1,036995111	45,814	1,269093362	40,419	1,119645919
	Minimum	3791	5,013850	3990	10,526316	4106	13,739612	4078	12,963989
	Maksimum	3899	8,005540	4114	13,961219	4292	18,891967	4210	16,620499
	Ortalama	3882,40	5,47133931	4075,85	10,72670470	4201,05	14,12795436	4169,70	13,27628362
	N	20	20	20	20	20	20	20	20
56	S. Sapma	38,922	1,057362496	27,963	0,759654940	37,282	1,012821587	39,064	1,061238378
	Minimum	3831	4,074980	4032	9,535452	4135	12,333605	4099	11,355610
	Maksimum	3990	8,394458	4119	11,898941	4269	15,973920	4245	15,321923
	Ortalama	3916,55	5,73839093	4096,55	10,59800216	4249,50	14,72732181	4230,80	14,22246220
	N	20	20	20	20	20	20	20	20
57	S. Sapma	24,265	0,655100710	39,260	1,059923363	27,433	0,740636404	45,353	1,224436216
	Minimum	3881	4,778618	4034	8,909287	4178	12,796976	4147	11,960043
	Maksimum	3979	7,424406	4183	12,931965	4286	15,712743	4317	16,549676
	Ortalama	3909,15	5,91032241	4095,35	10,95502574	4241,40	14,91194798	4217,30	14,25900840
	N	20	20	20	20	20	20	20	20
58	S. Sapma	33,693	0,912829421	33,718	0,913506326	45,762	1,239822675	41,883	1,134744572
	Minimum	3863	4,659984	4012	8,696830	4136	12,056353	4114	11,460309
	Maksimum	3991	8,127879	4153	12,516933	4306	16,662151	4280	15,957735
	Ortalama	3948,95	5,50227091	4131,90	10,39006145	4258,60	13,77504675	4244,00	13,38498531
	N	20	20	20	20	20	20	20	20
59	S. Sapma	30,258	0,808378892	27,421	0,732581008	40,311	1,076974571	34,053	0,909767323
	Minimum	3882	3,713599	4085	9,137056	4162	11,194229	4191	11,969009
	Maksimum	3989	6,572268	4204	12,316324	4344	16,056639	4309	15,121560
	Ortalama	3935,75	4,78567625	4127,20	9,88285410	4284,85	14,08013845	4277,25	13,87779553
	N	20	20	20	20	20	20	20	20
60	S. Sapma	23,324	0,620967133	36,198	0,963730105	33,108	0,881467456	56,763	1,511274630
	Minimum	3899	3,807242	4050	7,827476	4221	12,380192	4166	10,915868
	Maksimum	3981	5,990415	4194	11,661342	4345	15,681576	4355	15,947817
	Ortalama	2475,70	2,72429210	2533,35	4,61865809	2606,15	7,58248432	2597,86	7,25854828
Toplam	N	1200	1200	1200	1200	1200	1200	1200	1200
	S. Sapma	915,878	1,919874299	973,595	3,571193013	1007,162	4,293694728	1002,481	4,207294561

Tablo A.1. (Devam) Ajan Bölgesinin ve Optimal Arama Yönünün Etkisinin Değerlendirilmesine Yönelik Test Sonuçlarına Ait Tanımlayıcı İstatistikler

P. No	İstatistik	EBPM1		EBPM2		EBPM3		EBPM4	
		Makespan	Δ	Makespan	Δ	Makespan	Δ	Makespan	Δ
	Minimum	1089	0,000000	1098	0,000000	1098	0,441501	1103	0,104785
	Maksimum	4096	9,166667	4264	13,961219	4446	18,891967	4419	16,620499

Ek-B

Normallik testine ait IBM SPSS 25 paket programı raporu aşağıda verilmiştir:

EXAMINE VARIABLES=Makespan1 Yüzdesapma1Rt2_Ngrd_2 Makespan2 Yüzdesapma2Rt2_Ngrd_1 Makespan3

Yüzdesapma3Rt1_Ngrd_1 Makespan4 Yüzdesapma4Rt1_Ngrd_2

/PLOT BOXPLOT HISTOGRAM NPPLOT /COMPARE GROUPS /STATISTICS DESCRIPTIVES EXTREME /CINTERVAL 95

/MISSING LISTWISE /NOTOTAL.

Explore

Notes

Output Created		31-JAN-2023 13:12:14
Comments		
Input	Data	C:\Users\TURGAY OZTURK\Documents\IBM\SPSS\Ajan bölgesi anlamlılık testleri1\Normallik test verileri_AjanBölgesi&OptimalAramaYönü.sav
	Active Dataset	DataSet2
	Filter	<none>
	Weight	<none>
	Split File	<none>
	N of Rows in Working Data File	1200
Missing Value Handling	Definition of Missing	User-defined missing values for dependent variables are treated as missing.
	Cases Used	Statistics are based on cases with no missing values for any dependent variable or factor used.
Syntax		EXAMINE VARIABLES=Makespan1 Yüzdesapma1Rt2_Ngrd_2 Makespan2 Yüzdesapma2Rt2_Ngrd_1 Makespan3 Yüzdesapma3Rt1_Ngrd_1 Makespan4 Yüzdesapma4Rt1_Ngrd_2 /PLOT BOXPLOT HISTOGRAM NPPLOT /COMPARE GROUPS /STATISTICS DESCRIPTIVES EXTREME /CINTERVAL 95 /MISSING LISTWISE /NOTOTAL.
Resources	Processor Time	00:00:07,59
	Elapsed Time	00:00:03,69

Case Processing Summary

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
Makespan1	1200	100,0%	0	0,0%	1200	100,0%
Yüzde sapma1 (Rt2_Ngrd_2)	1200	100,0%	0	0,0%	1200	100,0%
Makespan2	1200	100,0%	0	0,0%	1200	100,0%
Yüzde sapma2 (Rt2_Ngrd_1)	1200	100,0%	0	0,0%	1200	100,0%
Makespan3	1200	100,0%	0	0,0%	1200	100,0%
Yüzde sapma3 (Rt1_Ngrd_1)	1200	100,0%	0	0,0%	1200	100,0%
Makespan4	1200	100,0%	0	0,0%	1200	100,0%
Yüzde sapma4 (Rt1_Ngrd_2)	1200	100,0%	0	0,0%	1200	100,0%

Descriptives

			Statistic	Std. Error
Makespan1	Mean		2475,70	26,439
	95% Confidence Interval for Mean	Lower Bound	2423,82	
		Upper Bound	2527,57	
	5% Trimmed Mean		2465,52	
	Median		2484,00	
	Variance		838831,611	
	Std. Deviation		915,878	
	Minimum		1089	
	Maximum		4096	
	Range		3007	
	Interquartile Range		1569	
	Skewness		,122	,071
	Kurtosis		-1,163	,141
Yüzde sapma1 (Rt2_Ngrd_2)	Mean		2,72429210	,055421997
	95% Confidence Interval for Mean	Lower Bound	2,61555722	
		Upper Bound	2,83302698	
	5% Trimmed Mean		2,64149772	
	Median		2,27416389	
	Variance		3,686	
	Std. Deviation		1,919874299	
	Minimum		,000000	
	Maximum		9,166667	
	Range		9,166667	
	Interquartile Range		3,238576	
	Skewness		,612	,071
	Kurtosis		-,630	,141
Makespan2	Mean		2533,35	28,105
	95% Confidence Interval for Mean	Lower Bound	2478,20	
		Upper Bound	2588,49	
	5% Trimmed Mean		2519,46	
	Median		2501,50	
	Variance		947887,387	
	Std. Deviation		973,595	
	Minimum		1098	

	Maximum		4264	
	Range		3166	
	Interquartile Range		1659	
	Skewness		,197	,071
	Kurtosis		-1,119	,141
Yüzde sapma2 (Rt2_Ngrd_1)	Mean		4,61865809	,103091462
	95% Confidence Interval for Mean	Lower Bound	4,41639836	
		Upper Bound	4,82091782	
	5% Trimmed Mean		4,45308981	
	Median		3,19461718	
	Variance		12,753	
	Std. Deviation		3,571193013	
	Minimum		,000000	
	Maximum		13,961219	
	Range		13,961219	
	Interquartile Range		6,071213	
	Skewness		,737	,071
	Kurtosis		-,855	,141
Makespan3	Mean		2606,15	29,074
	95% Confidence Interval for Mean	Lower Bound	2549,10	
		Upper Bound	2663,19	
	5% Trimmed Mean		2590,14	
	Median		2548,00	
	Variance		1014374,431	
	Std. Deviation		1007,162	
	Minimum		1098	
	Maximum		4446	
	Range		3348	
	Interquartile Range		1709	
	Skewness		,219	,071
	Kurtosis		-1,097	,141
Yüzde sapma3 (Rt1_Ngrd_1)	Mean		7,58248432	,123948290
	95% Confidence Interval for Mean	Lower Bound	7,33930466	
		Upper Bound	7,82566399	
	5% Trimmed Mean		7,48718487	
	Median		6,57382557	
	Variance		18,436	
	Std. Deviation		4,293694728	
	Minimum		,441501	
	Maximum		18,891967	
	Range		18,450466	
	Interquartile Range		7,141627	
	Skewness		,421	,071
	Kurtosis		-,980	,141
Makespan4	Mean		2597,86	28,939
	95% Confidence Interval for Mean	Lower Bound	2541,09	
		Upper Bound	2654,64	
	5% Trimmed Mean		2581,81	

	Median		2555,00	
	Variance		1004968,718	
	Std. Deviation		1002,481	
	Minimum		1103	
	Maximum		4419	
	Range		3316	
	Interquartile Range		1694	
	Skewness		,217	,071
	Kurtosis		-1,093	,141
Yüzde sapma4 (Rt1_Ngrd_2)	Mean		7,25854828	,121454132
	95% Confidence Interval for Mean	Lower Bound	7,02026202	
		Upper Bound	7,49683455	
	5% Trimmed Mean		7,15737730	
	Median		6,26894162	
	Variance		17,701	
	Std. Deviation		4,207294561	
	Minimum		,104785	
	Maximum		16,620499	
	Range		16,515713	
	Interquartile Range		6,770743	
	Skewness		,418	,071
	Kurtosis		-,962	,141

Extreme Values

			Case Number	Value
Makespan1	Highest	1	1003	4096
		2	1019	4074
		3	1007	4067
		4	1076	4061
		5	1012	4058
	Lowest	1	59	1089
		2	56	1089
		3	48	1089
		4	57	1090
		5	51	1091
Yüzde sapma1 (Rt2_Ngrd_2)	Highest	1	1076	9,166667
		2	1109	8,394458
		3	1142	8,127879
		4	1087	8,005540
		5	850	7,925326
	Lowest	1	794	,000000
		2	698	,000000
		3	651	,000000
		4	607	,000000
		5	182	,000000[a]
Makespan2	Highest	1	1007	4264
		2	1010	4254
		3	1005	4250
		4	1018	4250
		5	1016	4244
	Lowest	1	60	1098
		2	59	1098

		3	56	1098
		4	55	1098
		5	54	1098[b]
Yüzde sapma2 (Rt2_Ngrd_1)	Highest	1	1088	13,961219
		2	1095	13,601108
		3	1097	13,573407
		4	1096	13,490305
		5	1090	13,047091
	Lowest	1	798	,000000
		2	795	,000000
		3	30	,000000
		4	29	,000000
		5	697	,034928[c]
Makespan3	Highest	1	1004	4446
		2	1020	4442
		3	1010	4437
		4	1003	4425
		5	1006	4409
	Lowest	1	47	1098
		2	44	1115
		3	53	1125
		4	49	1128
		5	60	1132[d]
Yüzde sapma3 (Rt1_Ngrd_1)	Highest	1	1082	18,891967
		2	1094	17,506925
		3	1095	17,257618
		4	1096	17,036011
		5	1153	16,662151
	Lowest	1	25	,441501
		2	35	,515085
		3	27	,588668
		4	609	,624082
		5	40	,662252[e]
Makespan4	Highest	1	1014	4419
		2	1006	4418
		3	1005	4404
		4	1007	4396
		5	1010	4391
	Lowest	1	43	1103
		2	55	1113
		3	189	1128
		4	54	1128
		5	44	1130
Yüzde sapma4 (Rt1_Ngrd_2)	Highest	1	1098	16,620499
		2	1139	16,549676
		3	1047	16,456044
		4	1090	16,398892
		5	1097	16,232687
	Lowest	1	687	,104785
		2	603	,256975
		3	798	,359454
		4	29	,441501
		5	789	,467290

a. Only a partial list of cases with the value ,000000 are shown in the table of lower extremes.

b. Only a partial list of cases with the value 1098 are shown in the table of lower extremes.

c. Only a partial list of cases with the value ,034928 are shown in the table of lower extremes.

d. Only a partial list of cases with the value 1132 are shown in the table of lower extremes.

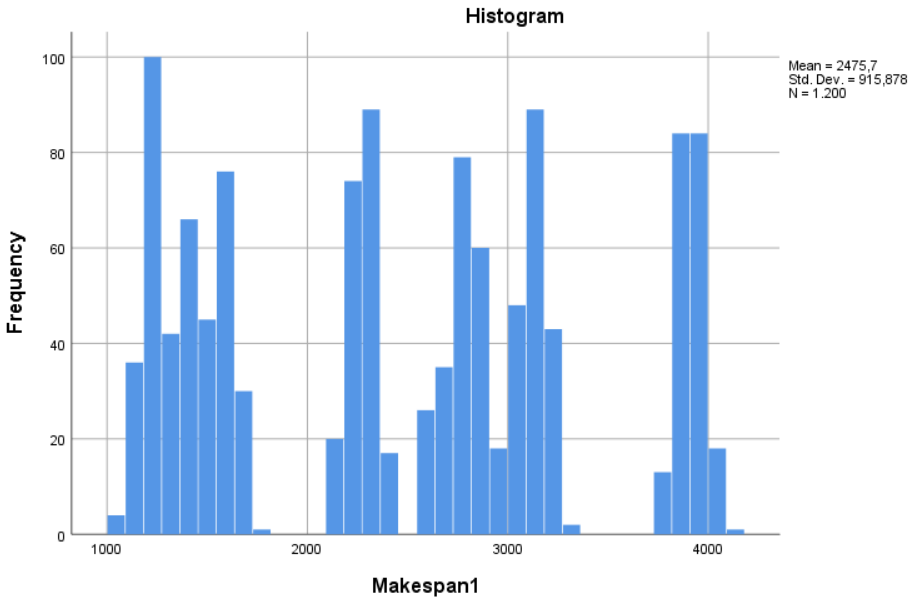
e. Only a partial list of cases with the value ,662252 are shown in the table of lower extremes.

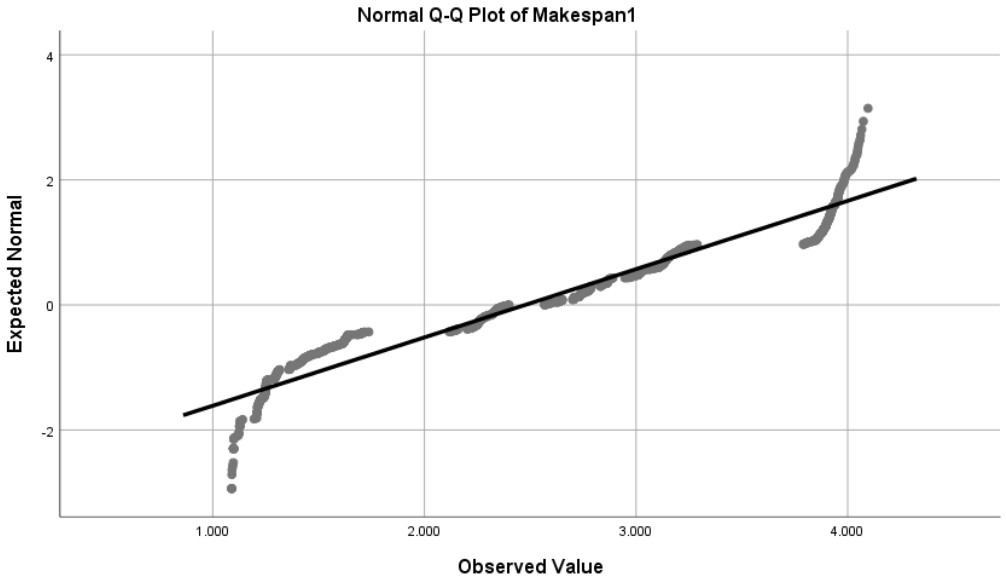
Tests of Normality

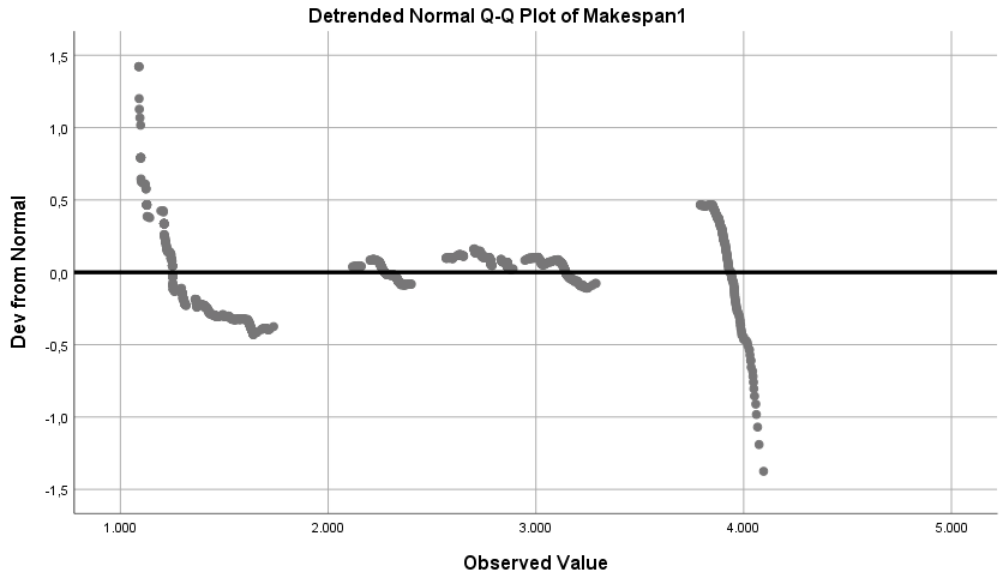
	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
Makespan1	,135	1200	,000	,927	1200	,000
Yüzde sapma1 (Rt2_Ngrd_2)	,114	1200	,000	,936	1200	,000
Makespan2	,132	1200	,000	,925	1200	,000
Yüzde sapma2 (Rt2_Ngrd_1)	,181	1200	,000	,876	1200	,000
Makespan3	,127	1200	,000	,927	1200	,000
Yüzde sapma3 (Rt1_Ngrd_1)	,101	1200	,000	,943	1200	,000
Makespan4	,124	1200	,000	,927	1200	,000
Yüzde sapma4 (Rt1_Ngrd_2)	,095	1200	,000	,946	1200	,000

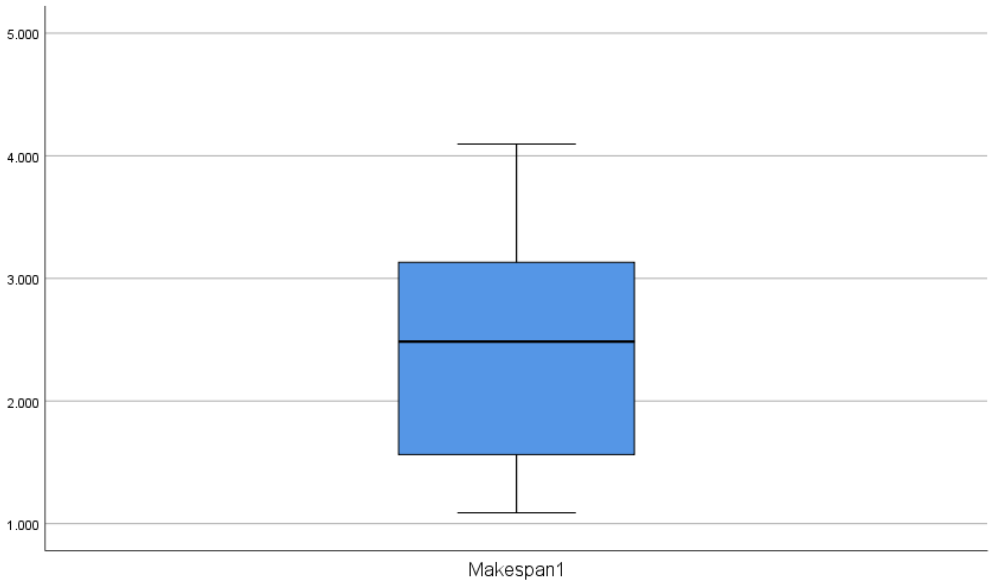
a. Lilliefors Significance Correction

Makespan1

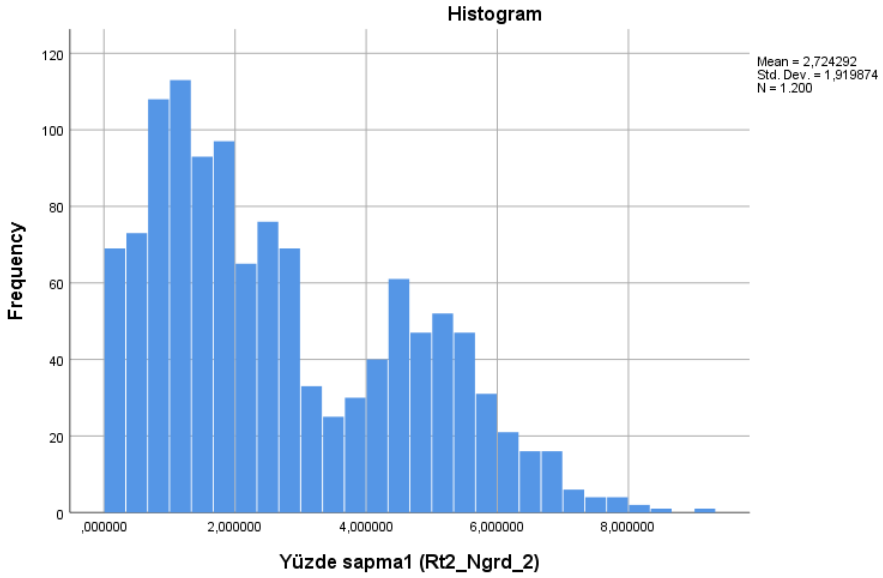


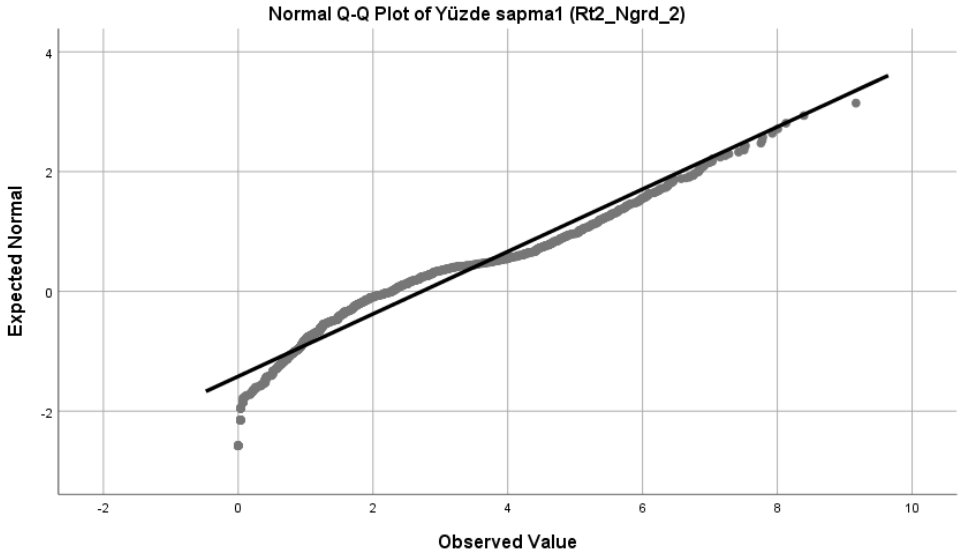

Normal Q-Q Plot of Makespan1
Expected Normal
4
2
0
-2
1.000
2.000
3.000
4.000
Observed Value

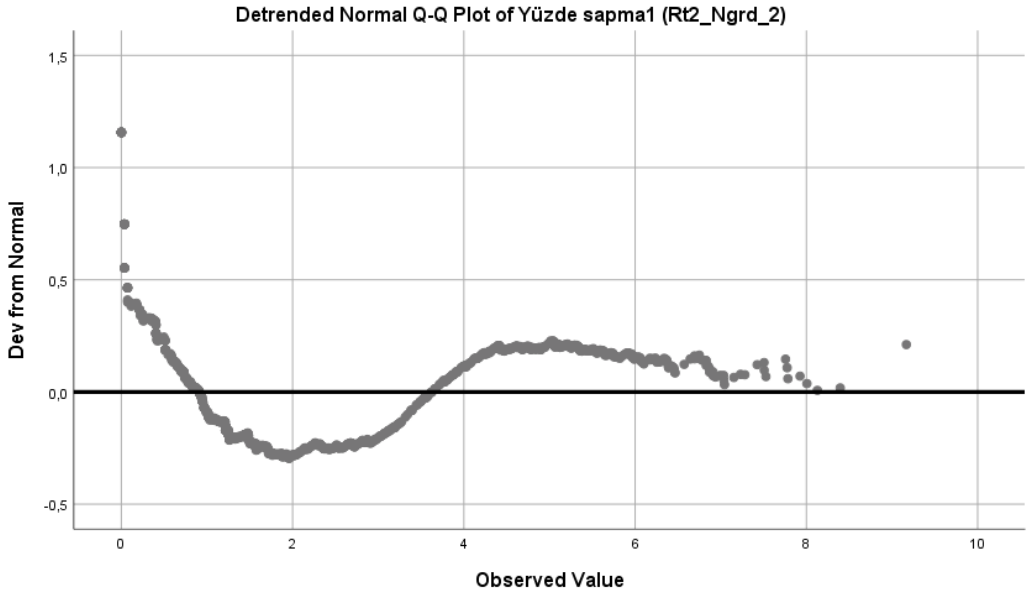

Detrended Normal Q-Q Plot of Makespan1
Dev from Normal
1,5
1,0
0,5
0,0
-0,5
-1,0
-1,5
1.000
2.000
3.000
4.000
5.000
Observed Value

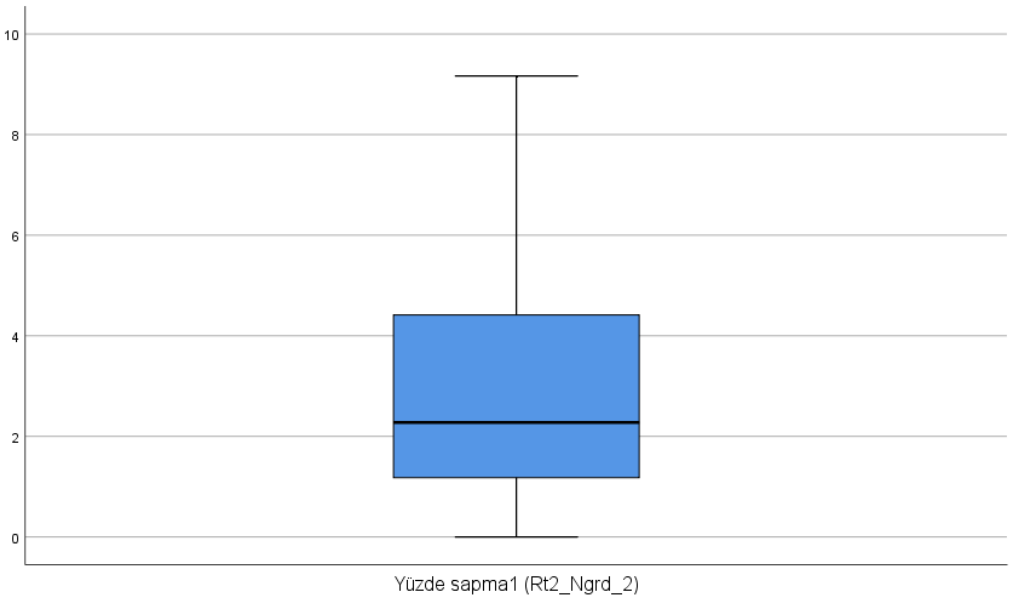


Yüzde sapma1 (Rt2_Ngrd_2)

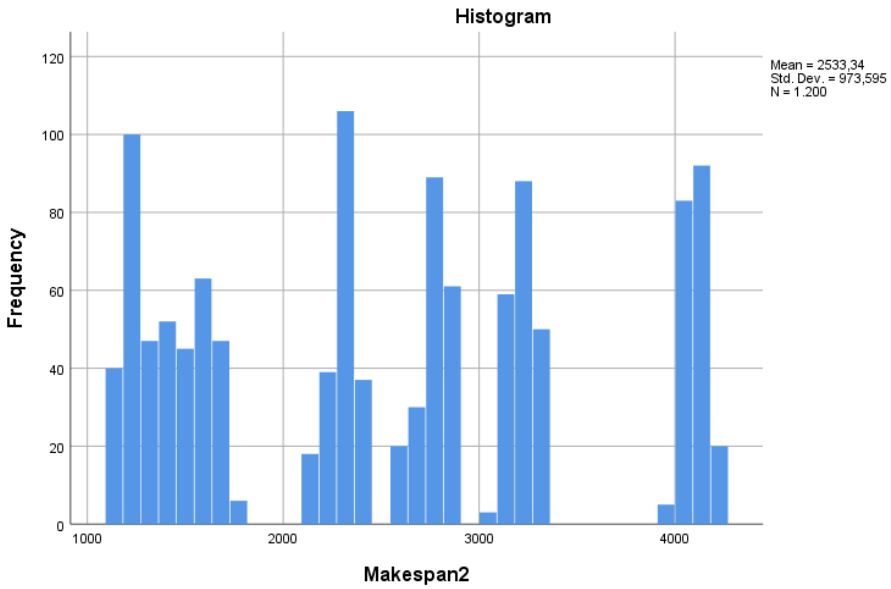


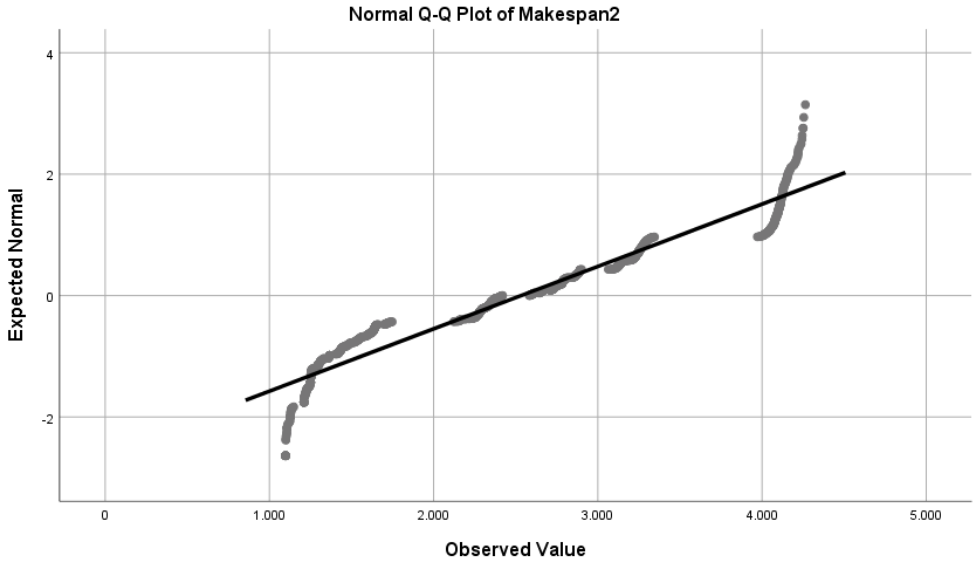

Normal Q-Q Plot of Yüzde sapma1 (Rt2_Ngrd_2)
Expected Normal
Observed Value
4
2
0
-2
-2
0
2
4
6
8
10

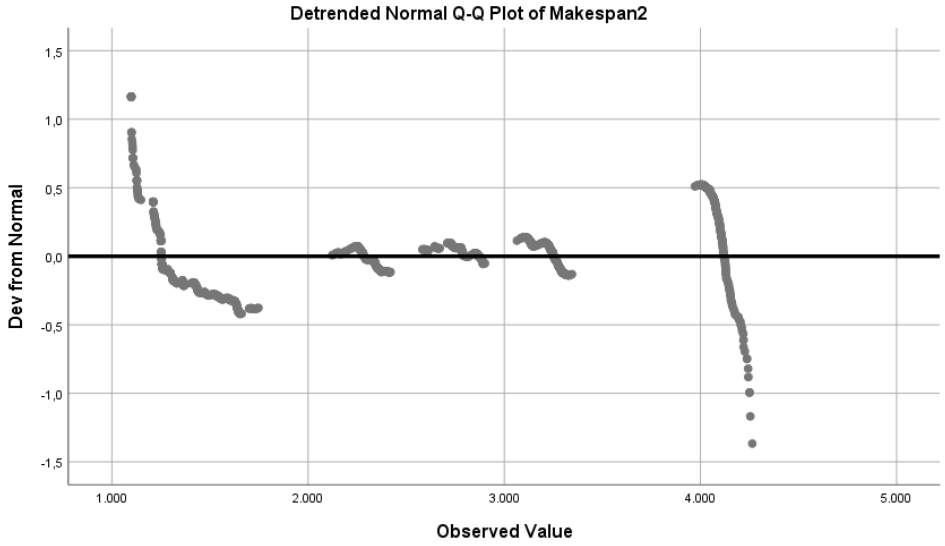

Detrended Normal Q-Q Plot of Yüzde sapma1 (Rt2_Ngrd_2)
Dev from Normal
Observed Value
1,5
1,0
0,5
0,0
-0,5
0
2
4
6
8
10

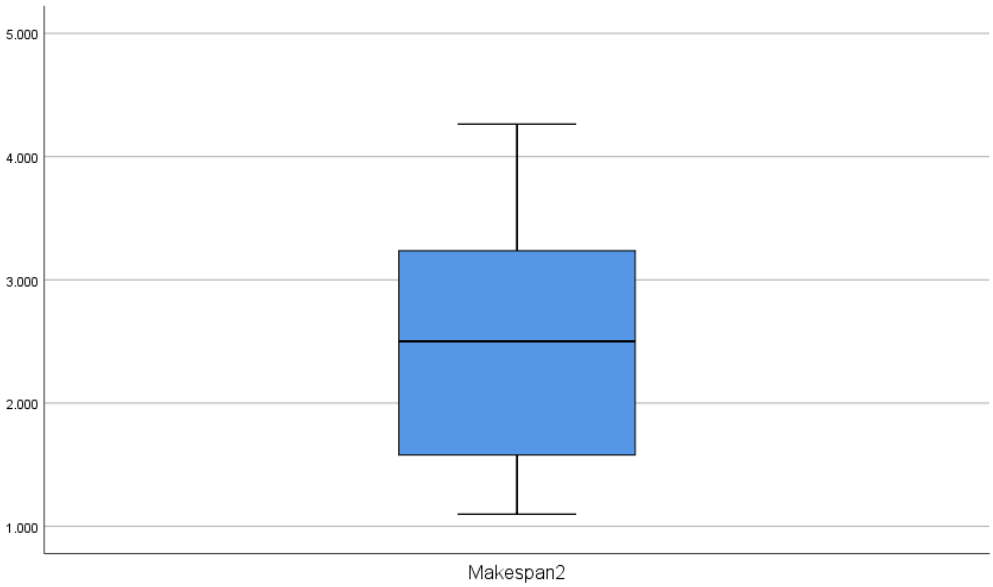

10
8
6
4
2
0
Yüzde sapma1 (Rt2_Ngrd_2)

Makespan2

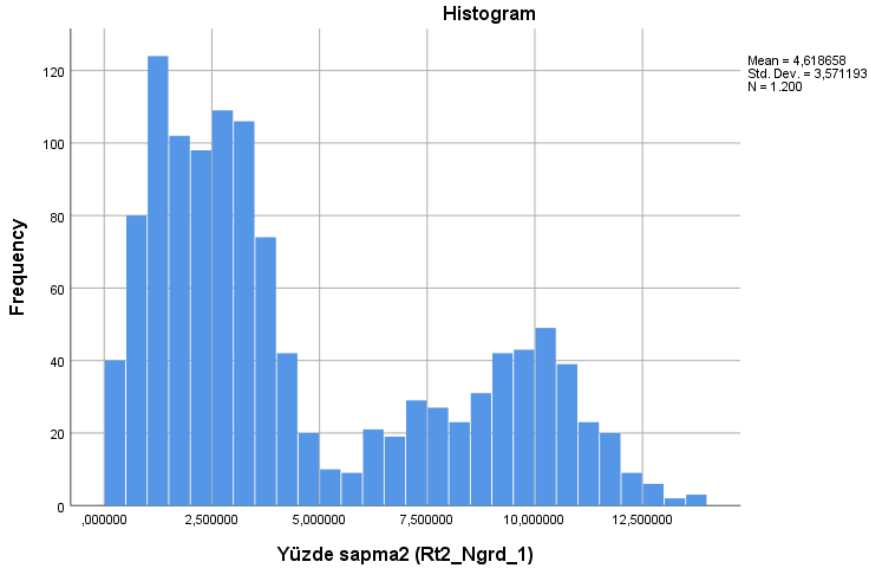

Histogram
Mean = 2533,34
Std. Dev. = 973,595
N = 1.200
Frequency
120
100
80
60
40
20
0
1000
2000
3000
4000
Makespan2

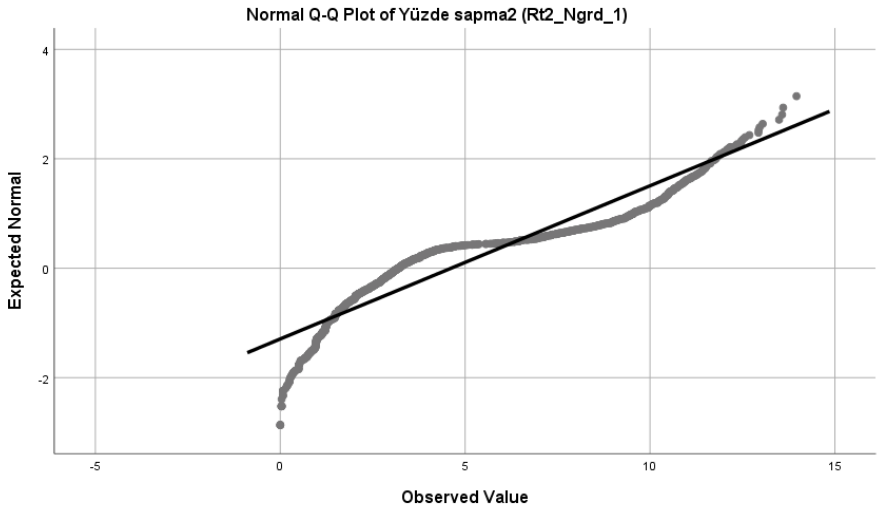

Normal Q-Q Plot of Makespan2
Expected Normal
4
2
0
-2
0
1.000
2.000
3.000
4.000
5.000
Observed Value

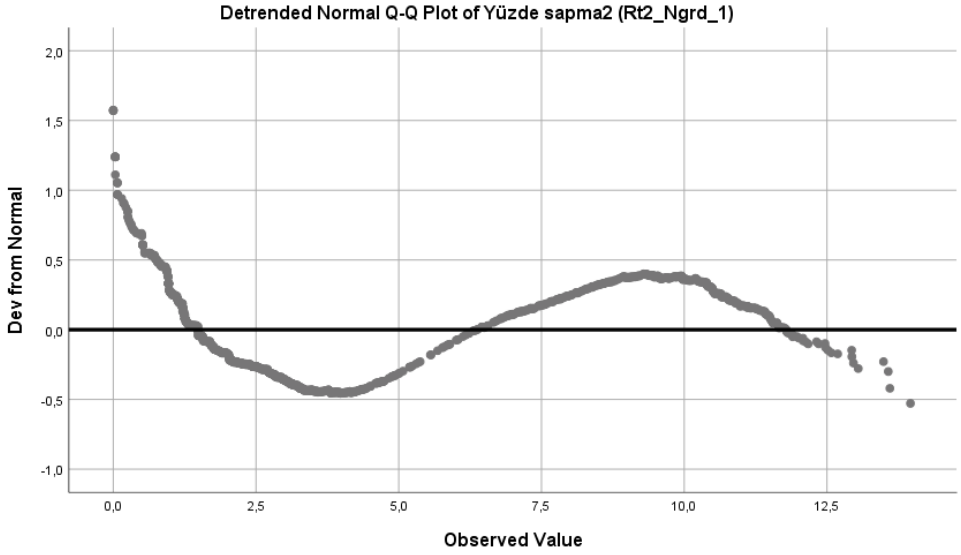

Detrended Normal Q-Q Plot of Makespan2
Dev from Normal
1,5
1,0
0,5
0,0
-0,5
-1,0
-1,5
1.000
2.000
3.000
4.000
5.000
Observed Value

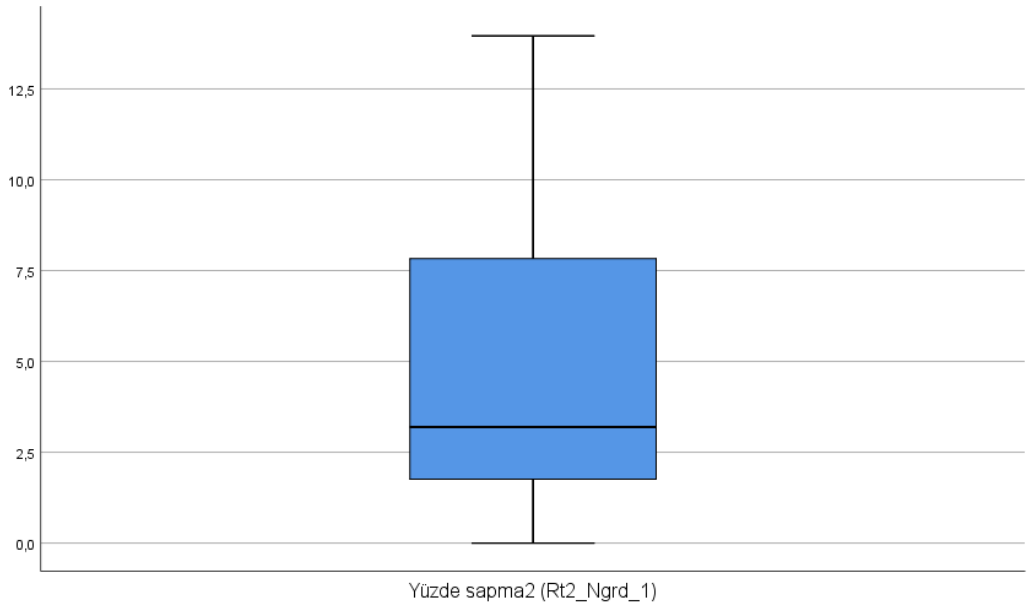


Yüzde sapma2 (Rt2_Ngrd_1)

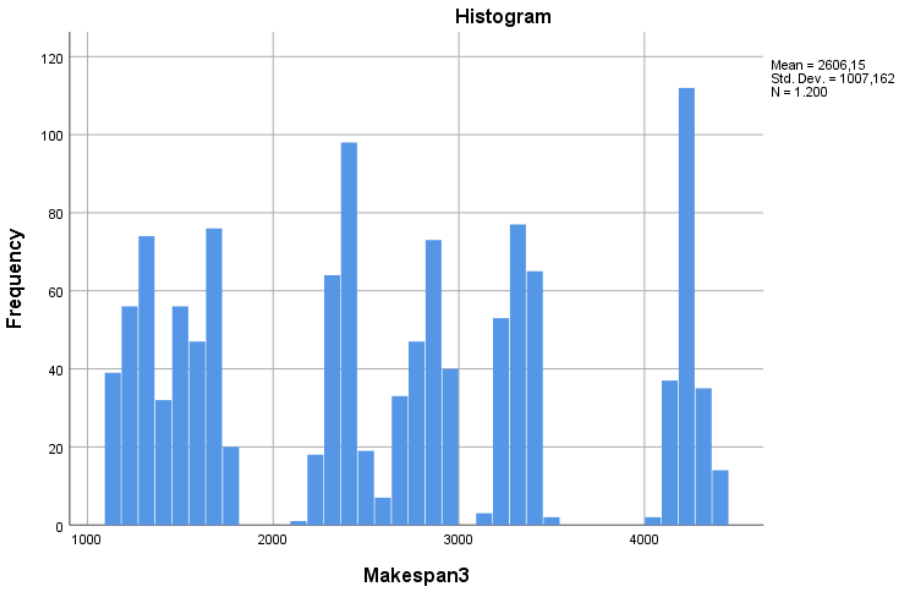


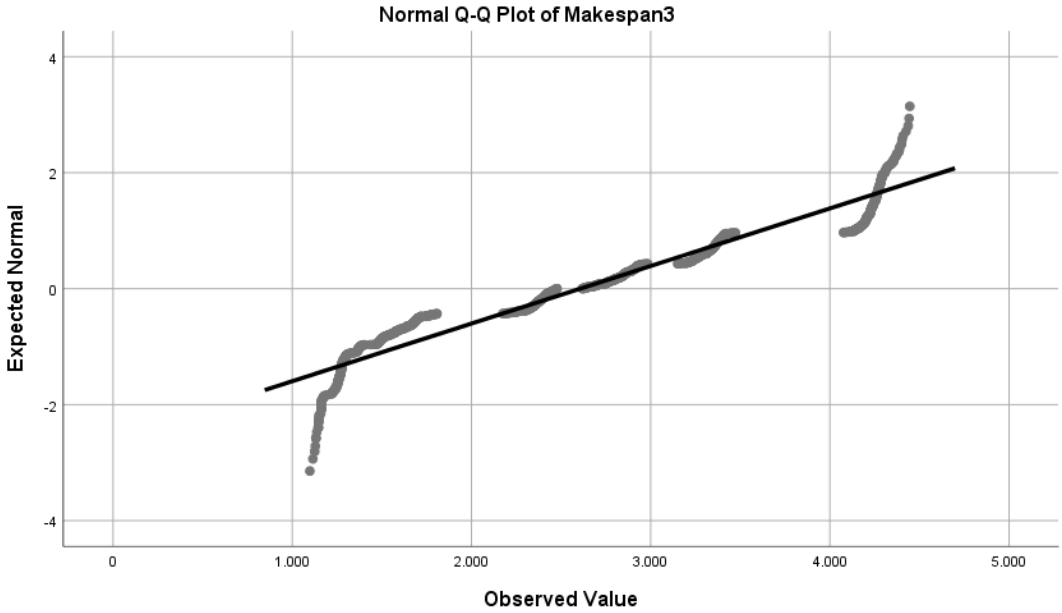

Normal Q-Q Plot of Yüzde sapma2 (Rt2_Ngrd_1)
Expected Normal
Observed Value
4
2
0
-2
-5
0
5
10
15

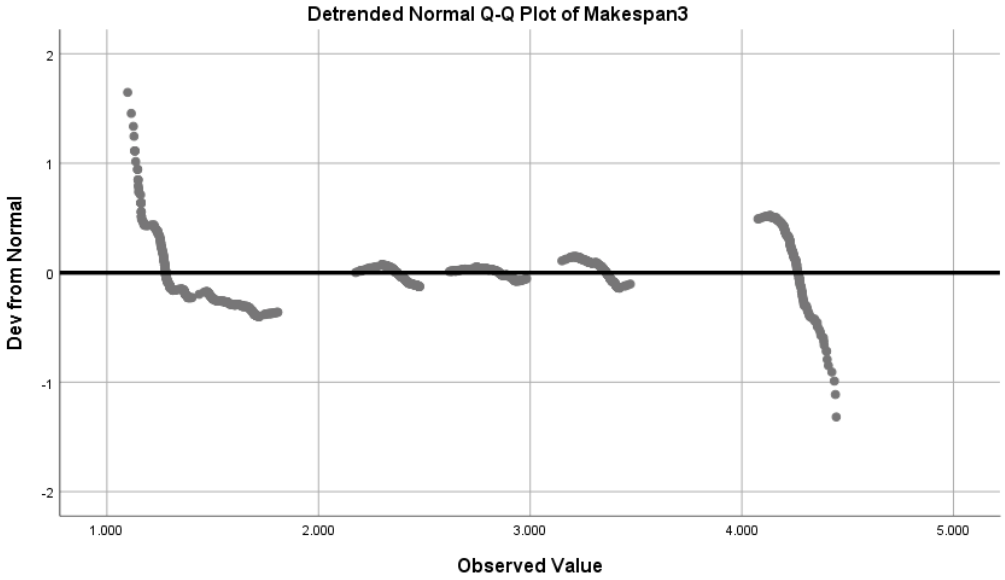

Detrended Normal Q-Q Plot of Yüzde sapma2 (Rt2_Ngrd_1)
Dev from Normal
Observed Value
2,0
1,5
1,0
0,5
0,0
-0,5
-1,0
0,0
2,5
5,0
7,5
10,0
12,5

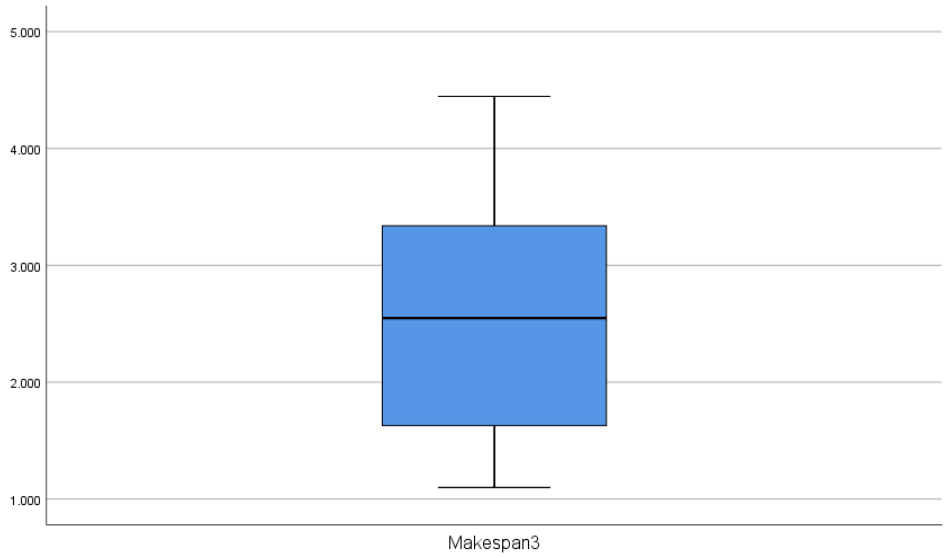

12,5
10,0
7,5
5,0
2,5
0,0
Yüzde sapma2 (Rt2_Ngrd_1)

Makespan3

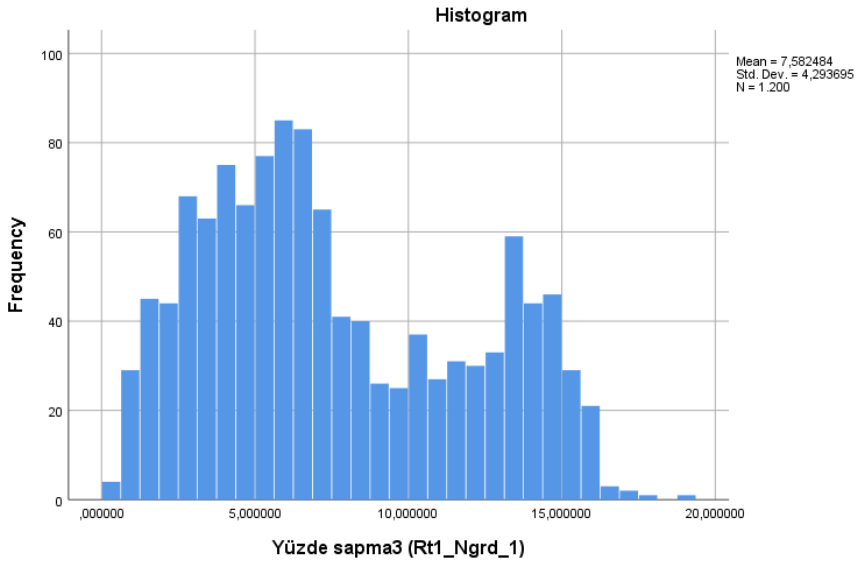

Histogram
Mean = 2606,15
Std. Dev. = 1007,162
N = 1.200
Frequency
120
100
80
60
40
20
0
1000
2000
3000
4000
Makespan3

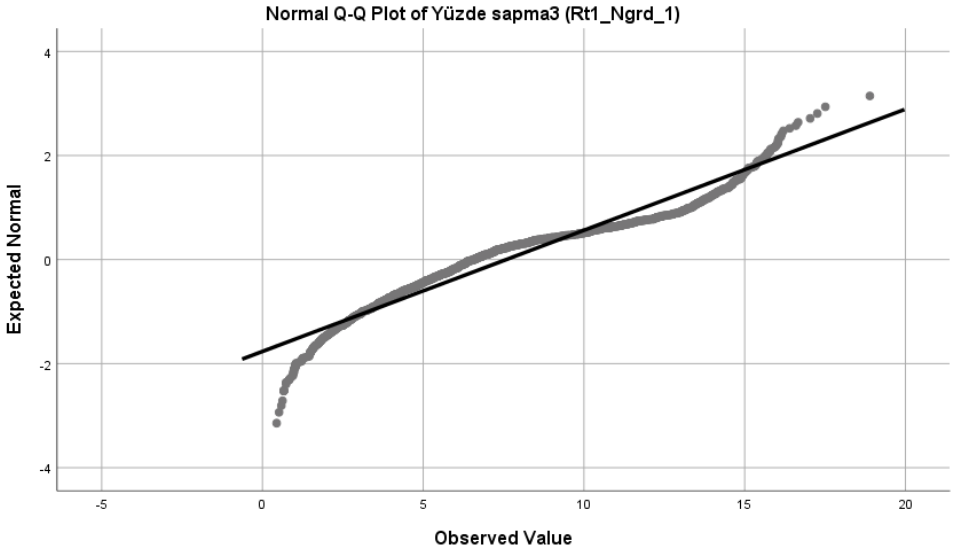

Normal Q-Q Plot of Makespan3
Expected Normal
4
2
0
-2
-4
0
1.000
2.000
3.000
4.000
5.000
Observed Value

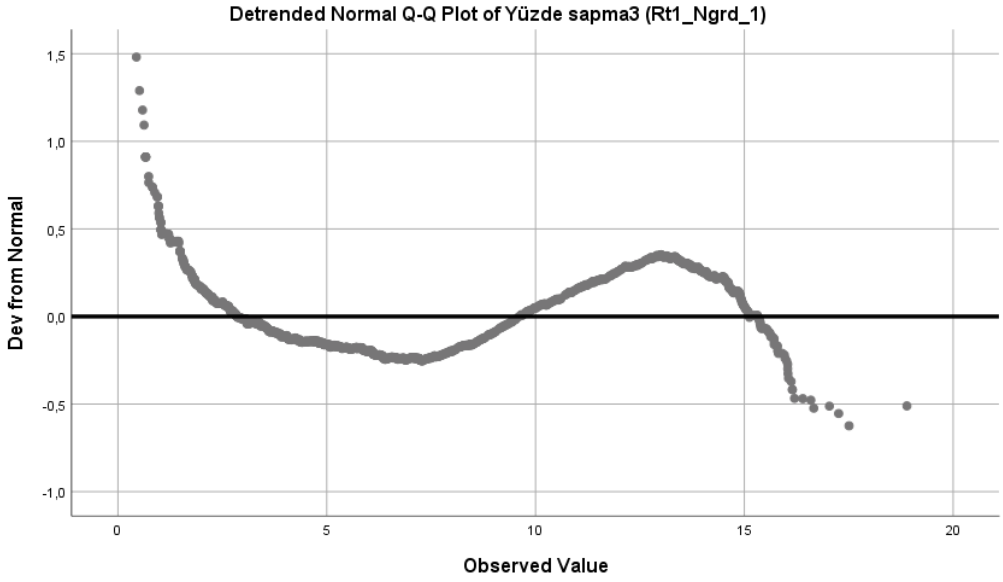

Detrended Normal Q-Q Plot of Makespan3
Dev from Normal
2
1
0
-1
-2
1.000
2.000
3.000
4.000
5.000
Observed Value

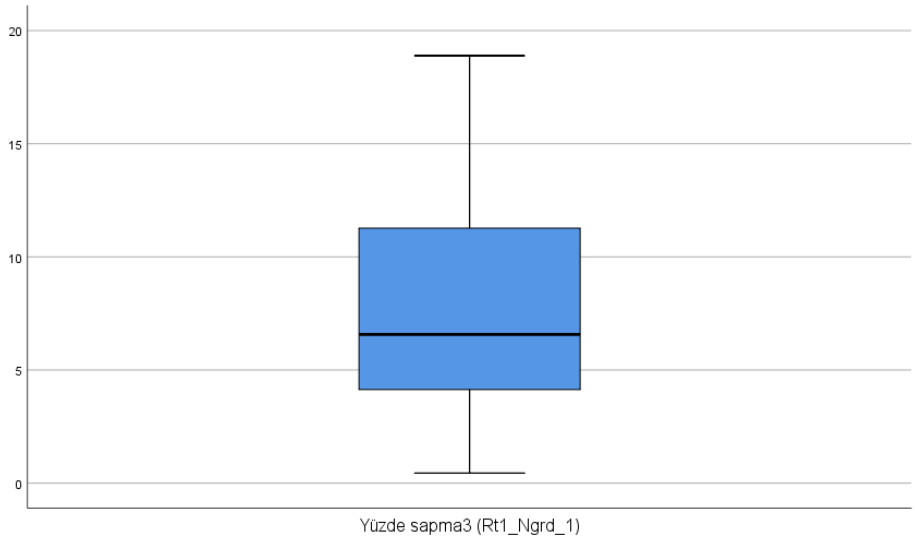


Yüzde sapma3 (Rt1_Ngrd_1)

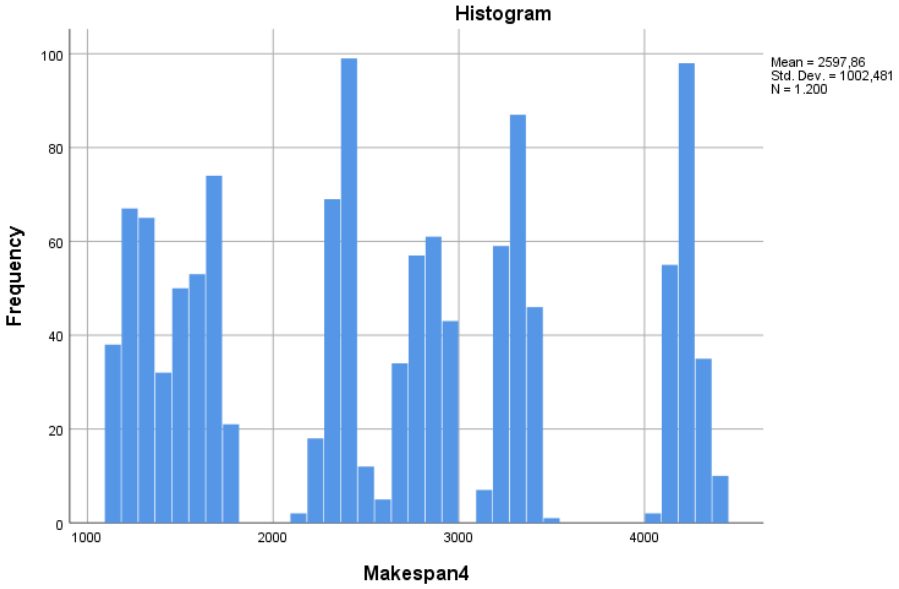


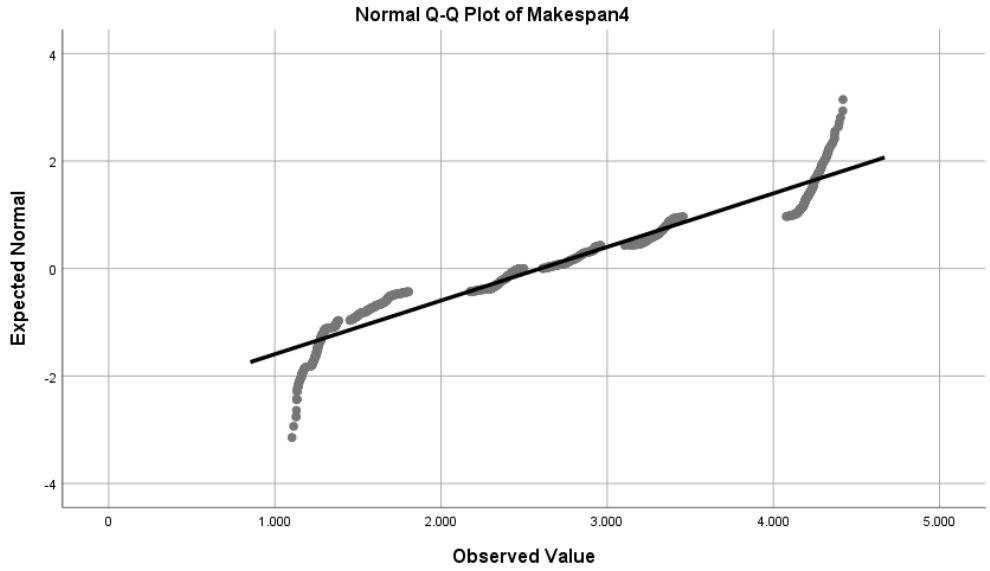

Normal Q-Q Plot of Yüzde sapma3 (Rt1_Ngrd_1)
Expected Normal
Observed Value
4
2
0
-2
-4
-5
0
5
10
15
20

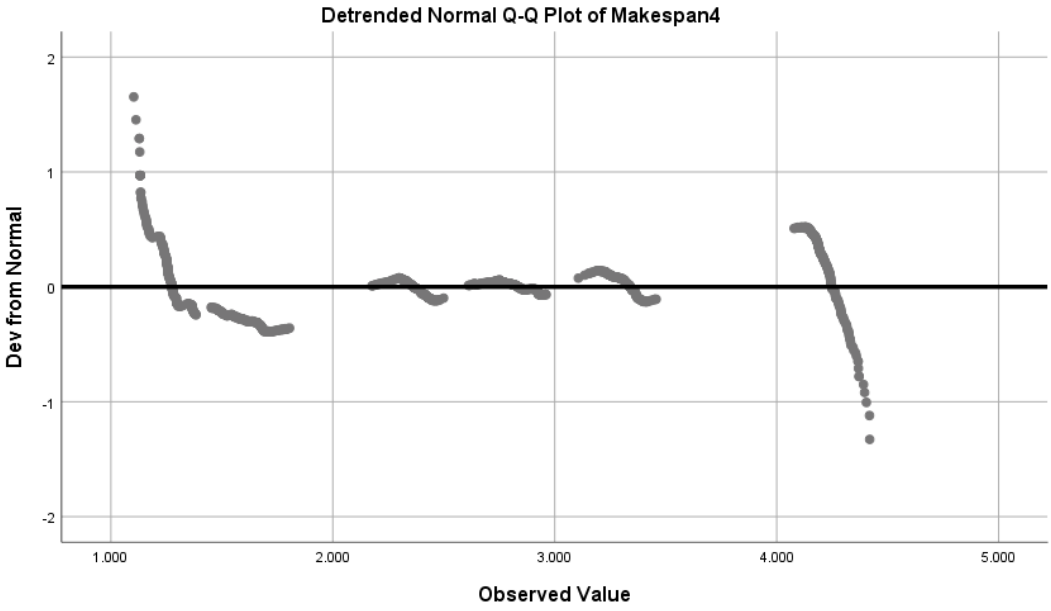

Detrended Normal Q-Q Plot of Yüzde sapma3 (Rt1_Ngrd_1)
Dev from Normal
Observed Value
1,5
1,0
0,5
0,0
-0,5
-1,0
0
5
10
15
20

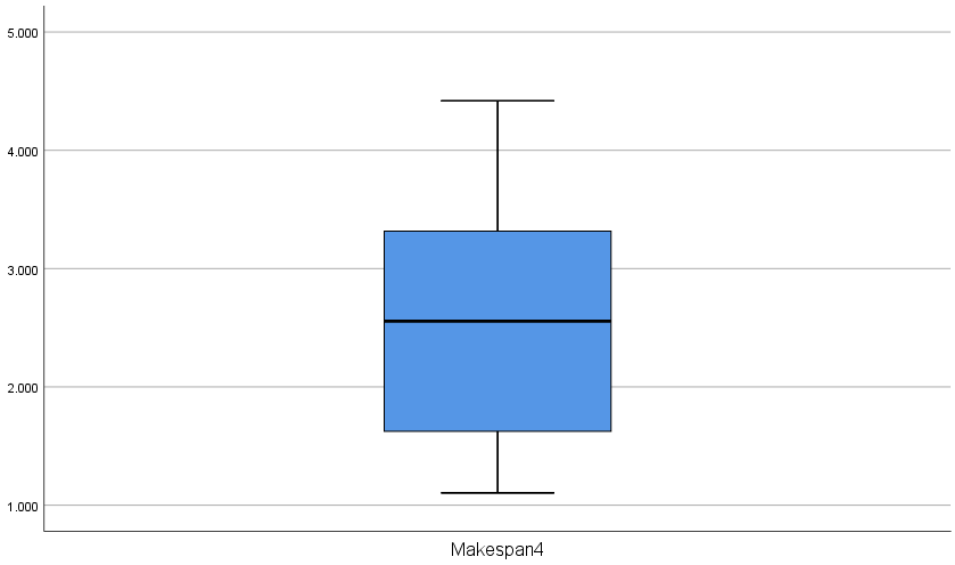

20
15
10
5
0
Yüzde sapma3 (Rt1_Ngrd_1)

Makespan4

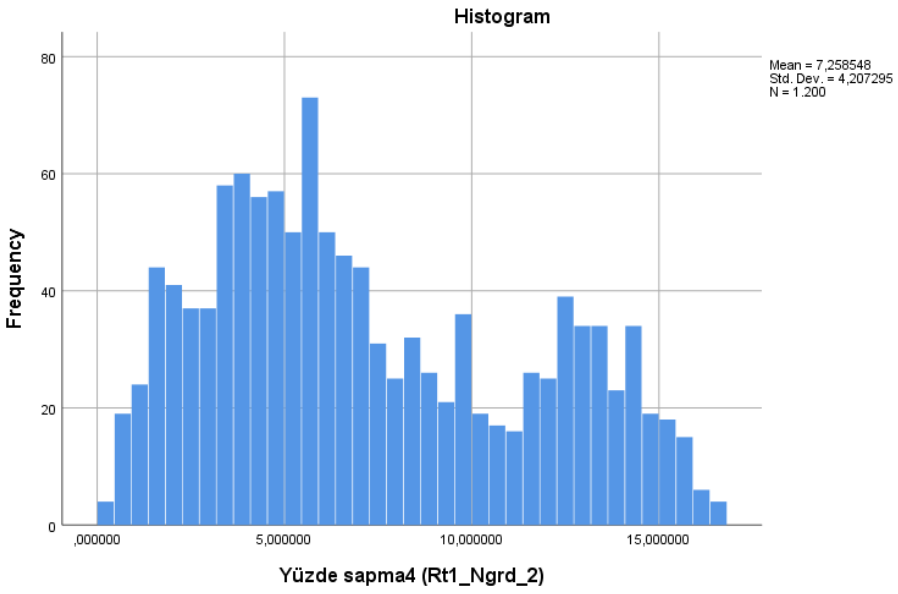

Histogram
Mean = 2597,86
Std. Dev. = 1002,481
N = 1.200
Frequency
100
80
60
40
20
0
1000
2000
3000
4000
Makespan4

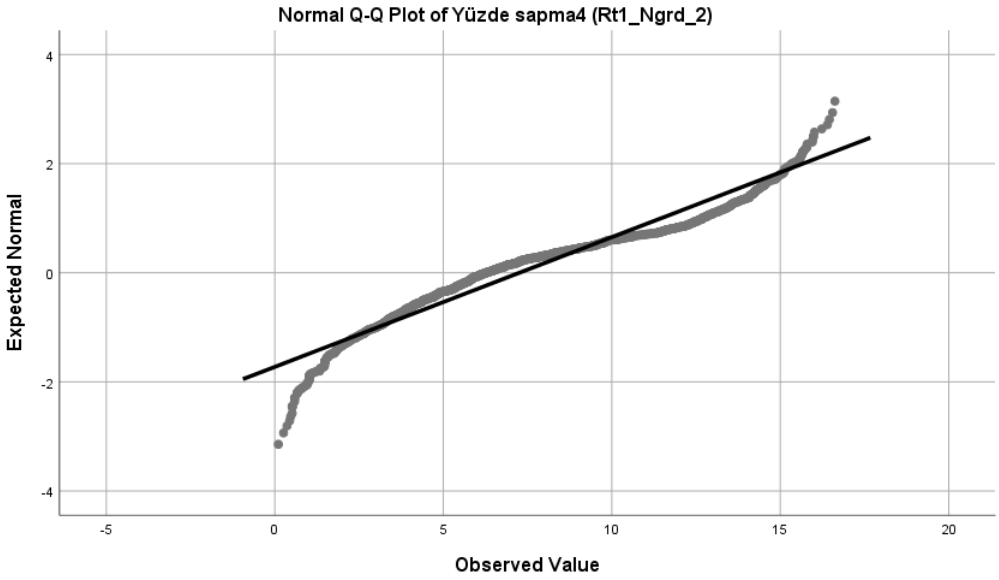

Normal Q-Q Plot of Makespan4
Expected Normal
4
2
0
-2
-4
0
1.000
2.000
3.000
4.000
5.000
Observed Value

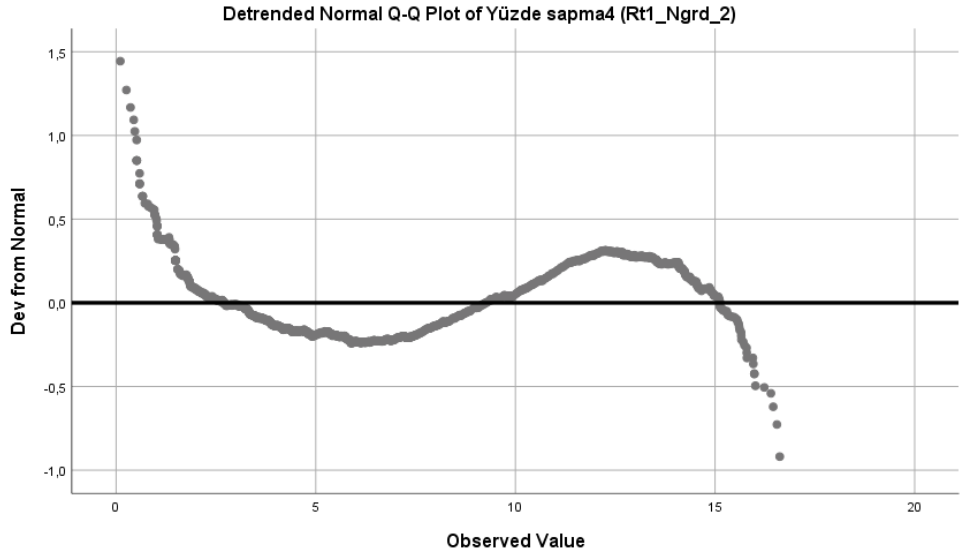

Detrended Normal Q-Q Plot of Makespan4
Dev from Normal
2
1
0
-1
-2
1.000
2.000
3.000
4.000
5.000
Observed Value

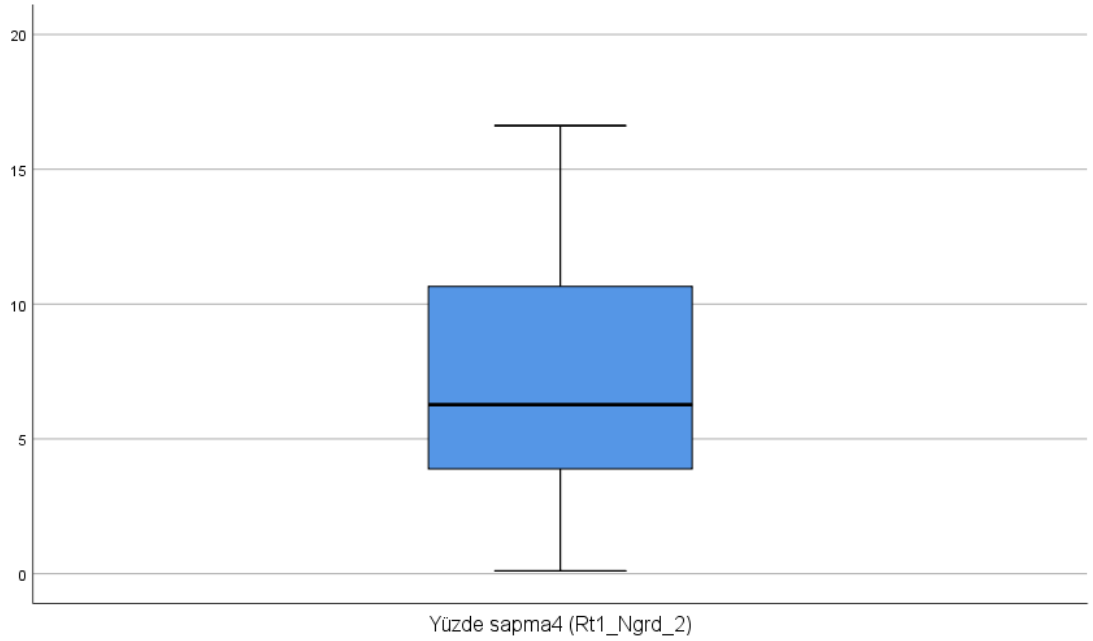


Yüzde sapma4 (Rt1_Ngrd_2)




Normal Q-Q Plot of Yüzde sapma4 (Rt1_Ngrd_2)
Expected Normal
Observed Value
4
2
0
-2
-4
-5
0
5
10
15
20


Detrended Normal Q-Q Plot of Yüzde sapma4 (Rt1_Ngrd_2)
Dev from Normal
Observed Value
1,5
1,0
0,5
0,0
-0,5
-1,0
0
5
10
15
20


20
15
10
5
0
Yüzde sapma4 (Rt1_Ngrd_2)

Ek-C

Heller (1960) tarafından üretilen iki problem grubunda üst sınır değerin alındığı kaynaktan Ancau (2012) daha iyi makespan değerlerine ulaşılmıştır. Bu problemlerde ulaşılan makespan değerleri ve çözümler aşağıda verilmiştir:

Problem: Hel1 (Heller 100x10)

Çözüm: (1, 22, 23, 74, 82, 13, 55, 87, 90, 27, 4, 73, 49, 70, 34, 54, 46, 86, 44, 51, 20, 52, 100, 84, 57, 16, 19, 95, 39, 68, 12, 61, 30, 76, 37, 25, 97, 92, 26, 32, 28, 91, 43, 9, 58, 14, 15, 50, 64, 93, 88, 56, 7, 59, 29, 24, 6, 62, 85, 48, 21, 42, 72, 83, 96, 69, 10, 81, 18, 78, 66, 77, 3, 45, 99, 8, 98, 53, 17, 38, 33, 2, 75, 79, 36, 80, 65, 67, 60, 89, 31, 41, 40, 94, 35, 71, 63, 11, 5, 47)

Üst sınır (Upper Bound):516

Bulunan maksimum tamamlanma zamanı(makespan): 515

Bağıl sapma yüzdesi (Δ): -0,19379845

Problem:Hel2 (Heller 20x10)

Çözüm: (1, 14, 20, 3, 17, 2, 13, 9, 8, 19, 15, 4, 16, 7, 6, 10, 18, 12, 11, 5)

Üst sınır (Upper Bound):136

Bulunan maksimum tamamlanma zamanı (makespan): 135

Bağıl sapma yüzdesi (Δ): -0,735294118

Ek-D

Aşağıda $EBPM_{NEH}$ ve $AEBPM_{RYA}$ yöntemlerine ait MATLAB kodları verilmiştir;

```
clear all; clc; close all;

%n_v=20;          % problem dimension

n_g_grd=2;      % No. of grenades that used OSD, equal to the number of desired minima to be found.

%ng=10;          % No. of grenades

n_p_min=40;      % No. of shrapnel

np=n_p_min;

Rrd=100;     % Reduction of territory radius

iter=300;        % No. of iteratiratio

psin=0.10; % weight of OSD ons

mmin=0.2;         % exploitation

total_runs=20;    % No. of independant runs

tolf=1e-2;       % target tolerance of the objective function

nmaks=500;

global P0  ;%cizelgelemeproblemi;deneme5

enyisonuclar=[];soncozumler=[];
sapmalar=[];ort_sapma_problem=[];ort_cpu_zamani=[];son_noktalar=[];% soncozumler her çalışmada en iyi bombanın koordinatı; son noktalar bütün bombaların ziyaret ettiği son noktalar

cpu_zamanlari=[];  I_soncozumler=[]; I_enyisonuclar=[];%test amaçlı eklendi

for problem=130:151%

disp(['Problem No.:' num2str(problem)])

[P0,obtm]=problemsec(problem);

[n,m]=size(P0);

Maks_time=n*m*0.1;%

n_v=n;
```

```
ng=round(n*m/20);

 for run_say=1:total_runs

% profile on; profile clear;

 [final_solutions,yayilmalar,cpuzamani(run_say,1),No_of_bads,
ave_func_eval,answer1]=GEM_OSD_time(n_v,n_g_grd,ng,n_p_min,Rrd,iter,mmin,psi
n,tolf,total_runs,obtm,Maks_time);

 [ed,I]=min(yayilmalar);

 enyisonuclar=[enyisonuclar;problem,run_say,ed];

 sapmalar=[sapmalar;problem,run_say,(ed-obtm)/obtm*100];

%sadece iterasyonlardaki sapmaları kaydetmek için kullanılacaksa algoritmanın ismi
içine yazılmalı

   % Sapma_iter=(answer1-obtm)/obtm*100;

   % akis_kayded_Sapma_iter(Sapma_iter,problem,run_say);

 size_eski=size(soncozumler,2);

  if n>size_eski-2 && size_eski>0

  soncozumler(:,(size_eski+1):n+2)=0;I_soncozumler(:,(size_eski+1):n+2)=0;

  son_noktalar(:,(size_eski+1):n+2)=0;

  soncozumler=[soncozumler;problem,run_say,final_solutions(I,:)];

son_noktalar=[son_noktalar;problem*ones(ng,1),run_say*ones(ng,1),final_solutions];

       [y,I_soncozum]=sort(final_solutions(I,:));

       I_soncozumler=[I_soncozumler;problem,run_say,I_soncozum];

  else

        soncozumler=[soncozumler;problem,run_say,final_solutions(I,:)];

 son_noktalar=[son_noktalar;problem*ones(ng,1),run_say*ones(ng,1),final_solutions];

           [y,I_soncozum]=sort(final_solutions(I,:));

           I_soncozumler=[I_soncozumler;problem,run_say,I_soncozum];

       end
```

```
I_enyisonuclar=[I_enyisonuclar;problem,run_say,uygunluk_yayilma_per(I_soncozum)]
;

   end

cpu_zamanlari=[cpu_zamanlari;problem*ones(total_runs,1),cpuzamani];

ort_cpu_zamani=[ort_cpu_zamani;problem,sum(cpuzamani)/total_runs];

cntrl=problem==sapmalar(:,1);

ort_sapma_problem=[ort_sapma_problem;problem,sum(sapmalar(:,3).*cntrl)/total_runs
];

end


ort_cpu_zamani,ort_sapma_problem

G_ort_cpu_zamani=sum(ort_cpu_zamani(:,2))/size(ort_cpu_zamani,1);

G_ort_sapma=sum(ort_sapma_problem(:,2))/size(ort_sapma_problem,1);

Par_List={'n','m','ng','np','Rrd','iter','mmin','total_runs','obtm','tolf','Maks_time','n_g_grd'
,'GEM_OSD_time','n*m/20'};

Par_deger=[n,m,ng,np,Rrd,iter,mmin,total_runs,obtm,tolf,Maks_time,n_g_grd];

akis_kayded(enyisonuclar,soncozumler,sapmalar,ort_sapma_problem,ort_cpu_zamani,n
p,ng,iter,mmin,Par_List,Par_deger);


akis_kayded_I(I_enyisonuclar,I_soncozumler,sapmalar,ort_sapma_problem,ort_cpu_za
mani,np,ng,iter,mmin,Par_List,Par_deger);

clear I aralik cntrl ed run_say n_g_grd ng n_p_min Rrd iter psin mmin total_runs tolf n
m np ardisik

%[y1,I2]=sort(son_noktalar(:,3:22),2);


function [final_solutions,nemoodarha,cpuzamani, No_of_bads,
ave_func_eval,answer1]=GEM_OSD_time(n_v,n_g_grd,n_g,n_p_min,shrn_co,iter,mmi
n,tavane_grd_w,tolf,total_runs,obtm,Maks_time)

t=cputime;%zamanı ölçmek için eklendi

good=0; %patlayan şarapnel sayısı
```

```
bad=0;  %hedefe ulaşılamayan çalıştırma sayısı
answer1=[]; %her itarasyonda ulaşılan en iyi uygunluk değeri;
keep_answers=[]; %en iyi uygunluk değerini veren çözüm kümesi
hadaf=[];  %kaçıncı iterasyonda hedefe ulaşıldığı
keep_nem=[]; %uygunluk değerleri
%nokta=[];nokta1=[]; %ziyaret edilen noktaları kaydetmek için ben koydum
%değişken çıktılarına ekle
for run_count=1:1%total_runs
   %disp(['run No.:' num2str(run_count)])
   n_p_inc=0;
   R_max=2*n_v^.5;  %initial value of search radius: Le çap
   shomar=zeros(1,n_g);% Le çapını güncellemede kullanılan "m" değeri
   A=.2*ones(1,n_g); %PTS=0.2  atamalarını yapar
   grd_w=0;
   p_cr=0;
   sh=shrn_co^(1/iter);
   a=sh^2/(2*sh-1);
   q=(2-1/sh)^(2/(iter-1));
   paresh=10/iter;
   R=R_max;
   [X R2]=thr(n_g,n_v,.999); % (-1 1) aralığında bomba sayısı*değişken sayısı kadar rasgele sayı (bombaları) üretir, R2 ajan yarı çapı
   p_weight=1;
   n_p=n_p_min*ones(n_g,1); %herbir bombanın başlangıçta üreteceği şarapnel sayısı belirlenir
   hadafyab=0;
   for m5=1:iter
```

```
if Maks_time<=(cputime-t); break, end %ÇIKIŞ AMAÇLI

[X ,n_p]=sort_kon(X,n_p); % uygunluk değerlerine göre sıralar, artan sıralama

forsat=iter/20*(1.5-m5/iter); %yerel optimuma takıldığında atlanacak değer kaç defa sonra z!!!!

if ((calc_fit(X(1,:))-obtm)<=(tolf) & hadafyab<.5) %.....(abs(calc_fit(X(1,:))-obtm)<=(tolf) & hadafyab<.5)

  hadaf=[hadaf;m5];

  hadafyab=1;

end

nem1(m5)=calc_fit(X(1,:)); % en iyi bombanın uygunluk değerini iterasyon saysı ile saklar

for kth=1:n_g

  if kth==1

    x_ch=10*ones(1,n_v);

  else

    x_ch=X(1:kth-1,:);

  end

  %if Maks_time<=(cputime-t); break, end %Çıkış amaçlı

  if kth<=n_g_grd

    num_g=calc_num_grd_2(X(kth,:),
R2,R2,1/n_v+(n_v^.3-1/n_v)*m5/iter,x_ch); % optmal arama yönü

    num_grd=num_g(1,:);

    num_grd_abs=(num_grd*num_grd')^.5;

    if num_grd_abs<1e-50

      num_grd_abs=1;

    end

    num_grd=num_grd/num_grd_abs;
```

```
        ezaf=num_g(3,1);

        if m5/iter<.1

           ezaf=2*n_v;

        end

        [z      new_R2]=xpld3(X(kth,:),R,R2,A(kth),n_p_min+n_p_inc*(kth-1)+ezaf,-
num_grd,grd_w,p_cr,x_ch); % generate n_p_min point arround the current grenade

        z=[z;X(kth,:)];    % nokta1=[nokta1;z]; % ben ekledim nokta yoktu

        nog_fit=calc_fit(z);   %% z_nog is the random produced point, produced points
in calculating gradient are not included

        nog_fit_min_index=find_min_n(nog_fit,1);

        z_bestia=z(nog_fit_min_index,:);

        z_bestia_dis=calc_dis(z_bestia,X(kth,:));

        if m5/iter >=.1

           %dis_cond_grd2=calc_dis(x_ch,num_g(2,:))>=R2;

           z=[z;num_g(2,:)];

        end

     else

        [z                new_R2]=xpld4(X(kth,:),R,R2,A(kth),n_p_min+n_p_inc*(kth-
1),p_cr,x_ch); % generate n_p_min point arround the current grenade

        z=[z;X(kth,:)];     %nokta1=[nokta;z]; % sonrada eklendi nokta yoktu

        nog_fit=calc_fit(z);   %% z_nog is the random produced point, produced points
in calculating gradient are not included

        nog_fit_min_index=find_min_n(nog_fit,1);

        z_bestia=z(nog_fit_min_index,:);

        z_bestia_dis=calc_dis(z_bestia,X(kth,:));

     end

     z_fit=calc_fit(z);
```

```
z_fit_min_index=find_min_n(z_fit,1); % en küçüğü buluyor.....aşğıda nokta taşınacak

if z_fit(z_fit_min_index)<calc_fit(X(kth,:)) % if the best point is better than the curent location of the grenade, move its location to that point

   z_bestia_rasta=z_bestia-X(kth,:);

   if sum(z_bestia_rasta.^2)>0

     z_bestia_rasta=z_bestia_rasta/((z_bestia_rasta*z_bestia_rasta')^.5);

     z_bestia_R_max=find_a(X(kth,:),z_bestia_rasta,R);

     p_weight=max(1/n_v,log10(R2/z_bestia_R_max)/log10(A(kth)));

     R_cr=.3^p_weight*z_bestia_R_max;%z_bestia_R_max*cr^p_weight;

     if z_bestia_dis<=R_cr

        gam=-z_bestia_dis/R_cr+1;

        vazn=.8-A(kth);

     else

        gam=-(z_bestia_dis-R_cr)/(z_bestia_R_max-R_cr); %.....Pts

        vazn=A(kth)-.05;

     end

     if ~(p_weight<=1.000001/n_v)

        A(kth)=gam*vazn*paresh+A(kth);

     end

   end

end

old_xkth_fit= calc_fit(X(kth,:));

if calc_fit(z(z_fit_min_index,:))<calc_fit(X(kth,:))

   X(kth,:)=z(z_fit_min_index,:);

end

if old_xkth_fit-calc_fit(X(kth,:))<tolf/forsat    %şarapnel sayısını azaltır
```

```
            shomar(kth)=shomar(kth)+1;
        else
            shomar(kth)=0;
            ezafe_shavad=n_p_min-n_p(kth);
            if ezafe_shavad>0
                n_p=tadil_shavad_afzayesh(n_p,kth,ezafe_shavad);   % tarif shavad
            end
        end
    end
    for mth=1:n_g
        if shomar(mth)>=forsat
            shomar(mth)=0;
            kahide_shavad=n_p(mth)-max(floor(n_p(mth)/2),1);
            n_p=tadil_shavad_kahesh(n_p,mth,kahide_shavad);   % tarif shavad
        end
    end       %şarapnel sayısı yeniden hesaplandı
    R2=new_R2;
    R2=R2/(a*q^(iter-m5));      % reduce R2
    m10=1-m5/iter*(1-mmin);     % new value of m10
    R=(R2)^(1-m10)*R_max^m10; % new value of R
    if m5/iter<.1
        grd_w=0;
    else
        grd_w=sin(((m5-.1*iter)/(.9*iter))^tavane_grd_w*pi/2);     %s379     adım     13:
grd_w=mosd, m5: itersyon no, tavane_grd_w:psin....
    end
```

```
      p_cr=grd_w*m5/iter+(1-m5/iter)*m5/iter;abs(calc_fit(X(1,:))-obtm);% NOKTALI VİRGÜL EKLENDİ

      if abs(calc_fit(X(1,:))-obtm)>tolf

         good=good+(n_p_min*n_g+2*n_v*n_g_grd)/total_runs;      %toplam şarapnel sayısı:denenen nokta sayısı

      end

     %nokta=[nokta;X];

   end % one run is finished

   if abs(min(nem1)-obtm)>tolf  %min(nem1)>tolf

      bad=bad+1;

   else

      keep_nem=[keep_nem;nem1];

   end

   answer1=[answer1 ;nem1];

   keep_answers=[keep_answers;X];

end %all runs re done

ave_func_eval=(good*total_runs-(n_p_min*n_g+2*n_v*n_g_grd)*bad*iter)/(total_runs-bad+1e-100);

No_of_bads=bad;

nemoodarha=calc_fit(X);     %answer1

final_solutions=keep_answers;

cpuzamani=cputime-t;

%abs(calc_fit(X(1,:))-obtm);


function [filename] = akis_kayded(enyisonuclar,soncozumler,sapmalar,ort_sapma_problem,ort_cpu_zamani,np,ng,iter,mmin,Par_List,Par_deger)

%UNTİTLED Summary of this function goes here
```

```
%   Detailed explanation goes here

A=[soncozumler,enyisonuclar,sapmalar];

indis=size(A,1)+2;

k=['A',num2str(indis)];k1=['F',num2str(indis)];k2=['F',num2str(indis+1)];k3=['F',num2st
r(indis+2)];

saat=clock;

dak=num2str(saat(5));

saat=num2str(saat(4));

filename=['k_',date,'_',saat,'_',dak];


 xlswrite('C:\Program
Files\MATLAB\R2009a\work30.03.2020\Testsonucları\problemlen_out_M.xlsx',      A,
filename, 'A1');

 xlswrite('C:\Program
Files\MATLAB\R2009a\work30.03.2020\Testsonucları\problemlen_out_M.xlsx',
[ort_sapma_problem,ort_cpu_zamani], filename, k);

 xlswrite('C:\Program
Files\MATLAB\R2009a\work30.03.2020\Testsonucları\problemlen_out_M.xlsx',
[np,ng,iter,mmin], filename, k1);

 xlswrite('C:\Program
Files\MATLAB\R2009a\work30.03.2020\Testsonucları\problemlen_out_M.xlsx',
Par_List, filename, k2);

xlswrite('C:\Program
Files\MATLAB\R2009a\work30.03.2020\Testsonucları\problemlen_out_M.xlsx',
Par_deger, filename, k3);

end


function                                [filename]                                =
akis_kayded(enyisonuclar,soncozumler,sapmalar,ort_sapma_problem,ort_cpu_zamani,n
p,ng,iter,mmin,Par_List,Par_deger)

%UNTİTLED Summary of this function goes here

%   Detailed explanation goes here
```

```
A=[soncozumler,enyisonuclar,sapmalar];

indis=size(A,1)+2;

k=['A',num2str(indis)];k1=['F',num2str(indis)];k2=['F',num2str(indis+1)];k3=['F',num2st
r(indis+2)];

saat=clock;

dak=num2str(saat(5));

saat=num2str(saat(4));

filename=['k_',date,'_',saat,'_',dak,'_I'];


 xlswrite('C:\Program
Files\MATLAB\R2009a\work30.03.2020\Testsonucları\problemlen_out_M.xlsx',      A,
filename, 'A1');

 xlswrite('C:\Program
Files\MATLAB\R2009a\work30.03.2020\Testsonucları\problemlen_out_M.xlsx',
[ort_sapma_problem,ort_cpu_zamani], filename, k);

 xlswrite('C:\Program
Files\MATLAB\R2009a\work30.03.2020\Testsonucları\problemlen_out_M.xlsx',
[np,ng,iter,mmin], filename, k1);

 xlswrite('C:\Program
Files\MATLAB\R2009a\work30.03.2020\Testsonucları\problemlen_out_M.xlsx',
Par_List, filename, k2);

 xlswrite('C:\Program
Files\MATLAB\R2009a\work30.03.2020\Testsonucları\problemlen_out_M.xlsx',
Par_deger, filename, k3);

end


function [X, R2]=thr(n_g,n_v,shrinkage)

% (-1 1) aralığında değişken sayısı kadar rasgele sayı üretir

%V1=2^n_v/n_g;

zaribe_hajm=[

   2.000000000000000e+000
```

3.141592653589793e+000

4.188790204786391e+000

4.934802200544679e+000

5.263789013914324e+000

5.167712780049969e+000

4.724765970331401e+000

4.058712126416768e+000

3.298508902738706e+000

2.550164039877345e+000

1.884103879389900e+000

1.335262768854589e+000

9.106287547832829e-001

5.992645293207919e-001

3.814432808233043e-001

2.353306303588931e-001

1.409811069171390e-001

8.214588661112819e-002

4.662160103008852e-002

2.580689139001405e-002

1.394915040902100e-002

7.370430945714347e-003

3.810656386852123e-003

1.929574309403922e-003

9.577224088231724e-004

4.663028057676122e-004

2.228721247212740e-004

```
    1.046381049248456e-004
    4.828782273891741e-005
    2.191535344783020e-005
    9.787139946737361e-006
    4.303069587032943e-006
    1.863467088262139e-006
    7.952054001475505e-007
    3.345288294108971e-007
    1.387895246221376e-007
    5.680828718331174e-008
    2.294842899726985e-008
    9.152230650159558e-009
    3.604730797462497e-009
    1.402564906073200e-009
    5.392664662608123e-010
    2.049436095396476e-010
    7.700707130601345e-011
    2.861552613910809e-011
    1.051847171693205e-011
    3.825460710520370e-012
    1.376864728037740e-012
    4.905322148884565e-013
    1.730219245836109e-013];
R2=n_v^.5;%n_v^.5;10^-10;%test amaçlı çok küçük bir sayı seçildi
chosen=[];
rejected=0;
```

```
while size(chosen,1)<n_g
   r1=(2*rand(1,n_v)-1);
   if size(chosen,1)>0
      dist1=calc_dis(chosen,r1); %uzaklık hesaplar
      cnd=dist1>=R2; %büyük ,eşitse se 1 değilse 0
      if ~(ismember(0,cnd))
         chosen=[chosen;r1];
         rejected=0;
      else    % bombalar uygun şekilde yerleştitrlemezse yarı capı küçültmekte
         rejected=rejected+1;
         if rejected>=(size(chosen,1)/n_g)*n_v*10
            R2=R2*shrinkage;
            rejected=0;
         end
      end
   else
     chosen=[chosen;r1];
   end
end %while
X=chosen;


function [x,y]=sort_kon(x,y)
% This function rearrange row of matrix x in order of their fitness-
% sıralama yapar -küçükten büyüğe doğru
x_size=size(x);
for m1=1:x_size(1)
```

```
   for m2=m1:x_size(1)-1
      if calc_fit(x(m2,:))>calc_fit(x(m2+1,:))% uygunluk değeri hesaplar
         z=x(m2,:);
         x(m2,:)=x(m2+1,:);
         x(m2+1,:)=z;
         z2=y(m2,:);
         y(m2,:)=y(m2+1,:);
         y(m2+1,:)=z2;
      end
   end
end

function yayilma=calc_fit(x)
% objective function. input is a matrix in which each row is a solution
% all input variables are in [-1,1], so scale yuor search range
% output is a vertical vector.  uygunluk değeri

%x=5*x-2;z=x.^2-10*cos(2*pi*x)+10;f=sum(z')';

%x1=x(:,1)*10;   x2=x(:,2)*10;  f=1/4*x1.^4-1/2*x1.^2+1/10*x1+1/2*x2.^2;
%%ap [-0.352386]

%x1=x(:,1).*2.048;     x2=x(:,2).*2.048;    f=3905.93-100*(x1.^2-x2).^2-(1-x1).^2;  f=0-
f;% dejong (maks)3905.93
```

```
%x1=x(:,1)*2;                x2=x(:,2)*2;          f=(1+(x1+x2+1).^2.*(19-14*x1+3*x1.^2-
14*x2+6*x1.*x2+3*x2.^2)).*(30+(2*x1-3*x2).^2.*(18-32*x1+12*x1.^2+48*x2-
36*x1.*x2+27*x2.^2)); (min=3)%gold stein and price


global P0;

%P0=cizelgelemeproblemi;
%TEXTREAD('C:\Users\turgay\Documents\MATLAB\work
29062010\GEM_OSD\cizproblem\tai20x5.txt')';

nokta=size(x,1);

[n m]=size(P0);

yayilma=zeros(nokta,1);

for nok=1:nokta

 [y,I]=sort(x(nok,:));

 P=P0(I,:);

   ES=zeros(n,m);

 %EB(1,1)=0;

 ES(1,1)=P(1,1);

   for i=2:n

      ES(i,1)=ES(i-1,1)+P(i,1);

   end

 for j=2:m

     ES(1,j)=ES(1,j-1)+P(1,j);

 end

 for i=2:n


      for j=2:m

         ES(i,j)=max(ES(i-1,j),ES(i,j-1))+P(i,j);

      end
```

```
 end

 yayilma(nok)=ES(n,m);


end


function calc_num_grd_2=calc_num_grd_2(x,R2,R_grd,p,x_ch)

% x: değişken problem boyutu kadar, R2:R_grd: ajan çapı,

% p:(n_v^.3-1/n_v)*m5/iter,bu değer rasal r değeriinin üstü; x_ch:10*ones(1,n_v) ve ya değişken matrisi

x_size=numel(x);

x_fit=calc_fit(x);

yeke=zeros(x_size);

x_ch_size=size(x_ch);

kharej=[];

kharej_fit=[];

for k=1:x_size

    yeke(k,k)=1;

end

ezafe=0;

r1=(.001+rand^p)/1.001; %rassal sayı üretilmiş (0-1) aralığı sağlansın diye bölme ve toplama yapılmış

for k=1:x_size

    tanaghoz=1;

    if (1-x(k))<min(.01,R_grd)

        stp(k)=-r1*min([R_grd 1+x(k)]);

        new_x=x+stp(k)*yeke(k,:);

        new_x_fit=calc_fit(new_x);
```

```
    tanaghoz=tanaghoz-1;
    rayate_fasele=calc_dis(x_ch,new_x)>=R2*ones(x_ch_size(1),1);
    if sum(rayate_fasele)==numel(x_ch_size(1))
        kharej=[kharej;new_x];
        kharej_fit=[kharej_fit;new_x_fit];
    end
    grd(k)=(new_x_fit-x_fit)/stp(k);
    ezafe=ezafe+1;
end
if (1+x(k))<min(.01,R_grd)
    stp(k)=r1*min([R_grd 1-x(k)]);
    new_x=x+stp(k)*yeke(k,:);
    new_x_fit=calc_fit(new_x);
    tanaghoz=tanaghoz-1;
    rayate_fasele=calc_dis(x_ch,new_x)>=R2*ones(x_ch_size(1),1);
    if sum(rayate_fasele)==numel(x_ch_size(1))
        kharej=[kharej;new_x];
        kharej_fit=[kharej_fit;new_x_fit];
    end
    grd(k)=(new_x_fit-x_fit)/stp(k);
    ezafe=ezafe+1;
end
if min([1-x(k) 1+x(k)])>=min(.01,R_grd)
    stp(k)=r1*min([R_grd 1-x(k) 1+x(k)]);
    new_x1=x+stp(k)*yeke(k,:);
    new_x2=x-stp(k)*yeke(k,:);
```

```
        new_x1_fit=calc_fit(new_x1);
        new_x2_fit=calc_fit(new_x2);
        tanaghoz=tanaghoz-1;
        rayate_fasele=calc_dis(x_ch,new_x1)>=R2*ones(x_ch_size(1),1);
        if sum(rayate_fasele)==numel(x_ch_size(1))
           kharej=[kharej;new_x1];
           kharej_fit=[kharej_fit;new_x1_fit];
        end
        rayate_fasele=calc_dis(x_ch,new_x2)>=R2*ones(x_ch_size(1),1);
        if sum(rayate_fasele)==numel(x_ch_size(1))
           kharej=[kharej;new_x2];
           kharej_fit=[kharej_fit;new_x2_fit];
        end
        grd(k)=(new_x1_fit-new_x2_fit)/2/stp(k);
    end
end
if numel(kharej)==0
    kharej=x;
    kharej_fit=calc_fit(x);
end
x_best=kharej(1,:);
x_best_fit=kharej_fit(1);
best_x_index=find_min_n(kharej_fit,1); %en küçük değerin endeksini bulur
x_best=kharej(best_x_index,:);
calc_num_grd_2=[grd;x_best;ezafe*ones(1,numel(x))];
```

function [xpld3, new_R2]=xpld3(x,R,R2,A,n,grd,grd_w,p_cr,x_all)

%this function receive one solution for the optimization problem (x) and produce n allowable points arround

% it. R is range of exposion, R2 is radius of grenade's teritory and p is the exponent of the random number.

if numel(x)>50;tayeen_L=linspace(0.9,0.2,numel(x));else

tayeen_L=[

0

0.88622692545276

0.80599597700823

0.74522504471455

0.69699505843873

0.65743537038329

0.62418028536489

0.59568756662165

0.57090106804921

0.54906886292645

0.52963848612130

0.51219330719764

0.49641216596048

0.48204281799261

0.46888391610888

0.45677245647503

0.44557482623781

0.43518028929654

0.42549616044291

0.41644417228429

0.40795770086983

0.39997961914039

0.39246061684141

0.38535787087175

0.37863398324658

0.37225612443675

0.36619533712754

0.36042596586049

0.35492518628087

0.34967261414701

0.34464997865027

0.33984084772103

0.33523039596492

0.33080520732587

0.32655310691953

0.32246301659102

0.31852483062722

0.31472930852562

0.31106798144739

0.30753307122108

0.30411741922203

0.30081442431162

0.29761798783233

0.29452246547954

```
  0.29152262457562
  0.28861360604755
  0.28579089075777
  0.28305026949185
  0.28038781600247
  0.27779986302943]';
end
L=tayeen_L(numel(x))*R;
rejected=0;
xpld0=[];
cnd1=(calc_dis(x_all,x)-R2)==abs(calc_dis(x_all,x)-R2);
   cond_dis=sum(cnd1)==numel(cnd1);
if cond_dis
xpld0=x;
xpld0_size=size(xpld0);
end

xpld0_size=size(xpld0);

while xpld0_size(1)<n % n is the number of point to be generated.
   game_mojaz=0;
   while game_mojaz<1e-5*R2
      if rand<p_cr

         ds=prd_rand_dir_num(numel(x));
```

```
        rasta= ds*(1-grd_w)*rand+grd*rand*grd_w; % z is the random point generated,
but it is possible that this point be unacceptableandom

        rasta_abs=(rasta*rasta')^.5;

        rasta=rasta/rasta_abs;

        game_mojaz=find_a(x,rasta,R);

      else



        mojz=calc_share(x,L);

        ds=prd_rand_dir_sq(x,mojz);

        ds=ds/(ds*ds')^.5;

        rasta= ds*(1-grd_w)*rand+grd*rand*grd_w; % z is the random point generated,
but it is possible that this point be unacceptableandom

        rasta_abs=(rasta*rasta')^.5;

        rasta=rasta/rasta_abs;

        game_mojaz=find_a_sq(x,rasta,mojz);

      end

   end %while

   p_weight=max(1/numel(x),log(R2/game_mojaz)/log(A)); % max(1/n,(log(Rt/Re)/log
(Pts)))).....

   z=x+rasta*game_mojaz*rand^p_weight;

   cnd2=abs(1-abs(z))==1-abs(z); %

   cond_range=sum(cnd2)==numel(cnd2);

   if ~cond_range

     'error: outside bound'

   end

   cnd1=(calc_dis(x_all,z)-R2)==abs(calc_dis(x_all,z)-R2);

   cond_dis=sum(cnd1)==numel(cnd1);
```

```
    if cond_dis
      rejected=0;
      xpld0=[xpld0;z];
      xpld0_size=size(xpld0);
    else
       rejected=rejected+1;
    end
    if rejected>(size(xpld0,1)/n)*numel(x)*10
       R2=R2*.999;
    end
end
xpld3=[xpld0]; %(rand(numel(x),numel(x))-.5)*2];
new_R2=R2;


function [xpld4, new_R2]=xpld4(x,R,R2,A,n,p_cr,x_all)
%this function receive one solution for the optimization problem (x) and produce n allowable points arround
% it. R is range of exposion, R2 is radius of grenade's teritory and p is the exponent of the random number.
%x= etrafında şarapnel üretilecek bomba,    R2 = ajan çapı; R= patlama çapı,    n=
%şarapnel sayısı,    A=PTS; P_cr=???? ;     x_all bombalar
%p_weight= p eşitlik eşitlik (2.7)
%rasta=yön vektörü;       game_mojaz=patlama çapı hesapnanan
xpld0=[];
cnd1=(calc_dis(x_all,x)-R2)==abs(calc_dis(x_all,x)-R2);
    cond_dis=sum(cnd1)==numel(cnd1);
if cond_dis
```

```
xpld0=x;
xpld0_size=size(xpld0);
end
xpld0_size=size(xpld0);
if numel(x)>50;tayeen_L=linspace(0.9,0.2,numel(x));else
tayeen_L=[
                0
   0.88622692545276
   0.80599597700823
   0.74522504471455
   0.69699505843873
   0.65743537038329
   0.62418028536489
   0.59568756662165
   0.57090106804921
   0.54906886292645
   0.52963848612130
   0.51219330719764
   0.49641216596048
   0.48204281799261
   0.46888391610888
   0.45677245647503
   0.44557482623781
   0.43518028929654
   0.42549616044291
   0.41644417228429
```

0.40795770086983

0.39997961914039

0.39246061684141

0.38535787087175

0.37863398324658

0.37225612443675

0.36619533712754

0.36042596586049

0.35492518628087

0.34967261414701

0.34464997865027

0.33984084772103

0.33523039596492

0.33080520732587

0.32655310691953

0.32246301659102

0.31852483062722

0.31472930852562

0.31106798144739

0.30753307122108

0.30411741922203

0.30081442431162

0.29761798783233

0.29452246547954

0.29152262457562

0.28861360604755

```
  0.28579089075777
  0.28305026949185
  0.28038781600247
  0.27779986302943]';
end
L=tayeen_L(numel(x))*R;
rejected=0;

while xpld0_size(1)<n % n is the number of point to be generated.
   game_mojaz=0;
   while game_mojaz<1e-5*R2
      if rand<p_cr
         ds=prd_rand_dir_num(numel(x));
         rasta=ds;
         game_mojaz=find_a(x,rasta,R);
      else
         mojz=calc_share(x,L);%2*nmatris
         ds=prd_rand_dir_sq(x,mojz);
         ds=ds/(ds*ds')^.5;
         rasta=ds;%vektör
         game_mojaz=find_a_sq(x,rasta,mojz);%sayı
      end
   end %while
   p_weight=max(1/numel(x),log(R2/game_mojaz)/log(A));% eşitlik (2.7)
   z=x+rasta*game_mojaz*rand^p_weight; % bir şarapnel üretildi şartları sağlıyor mu
test edilcek
```

```
    cnd2=abs(1-abs(z))==1-abs(z); %
    cond_range=sum(cnd2)==numel(cnd2);
    if ~cond_range
      'mage mihse?'
    end
    cnd1=(calc_dis(x_all,z)-R2)==abs(calc_dis(x_all,z)-R2);%tüm boblara uzaklık ölçüldü
    cond_dis=sum(cnd1)==numel(cnd1);
    if cond_dis
      xpld0=[xpld0;z];
      xpld0_size=size(xpld0); %uzaklıklar uyugun şarapnel kabul
      rejected=0;
    else
       rejected=rejected+1; %şarapnel kabul değil şartlar sağlanan kadar tekrar dene
    end
    if rejected>(size(xpld0,1)/n)*numel(x)*10 %belli sayı boyunca şart sağnamaz sa çap küçültülur
       R2=R2*.999;
    end
end
xpld4=[xpld0]; %(rand(numel(x),numel(x))-.5)*2];
new_R2=R2;
```

```
clear all; clc; close all;


%n_v=20;          % problem dimension

n_g_grd=5; %5       % No. of grenades that used OSD, equal to the number of desired minima to be found.

ng=10;          % No. of grenades

n_p_min=40;      % No. of shrapnel

np=n_p_min;

Rrd=100;     % Reduction of territory radius

iter=1000000;       % No. of iteratiratio 100

psin=0.10; % weight of OSD ons

mmin=0.2;         % exploitation

total_runs=20;    % No. of independant runs

tolf=1e-2;       % target tolerance of the objective function

nmaks=500;

global P0  ;%cizelgelemeproblemi;deneme5

enyisonuclar=[];soncozumler=[];
sapmalar=[];ort_sapma_problem=[];ort_cpu_zamani=[];son_noktalar=[];% soncozumler her çalışmada en iyi bombanın koordinatı; son noktalar bütün bombaların ziyaret ettiği son noktalar

cpu_zamanlari=[];

%profile on; profile clear;

for problem=130:151%     97

disp(['Problem No.:' num2str(problem)])


[P0,obtm]=problemsec(problem);

[n,m]=size(P0);

Maks_time=n*m*0.06;%n*m*15*10^-3;%0.030- 0.10%n*m*0.1 %n*m*0.06
```

```
%ng=round(n*m/20);

 for run_say=1:total_runs

 [final_solutions,yayilmalar,cpuzamani(run_say,1)]=GEM_AKIS_PER5(n,m,ng,np,Rrd,
iter,mmin,total_runs,obtm,tolf,Maks_time,n_g_grd);%                    _PER11
GEM_AKIS_OSD(n,m,ng,np,Rrd,iter,mmin,total_runs,obtm,tolf,Maks_time,n_g_grd)

 [ed,I]=min(yayilmalar);

 enyisonuclar=[enyisonuclar;problem,run_say,ed];

 sapmalar=[sapmalar;problem,run_say,(ed-obtm)/obtm*100];


 size_eski=size(soncozumler,2);

  if n>size_eski-2 && size_eski>0

  soncozumler(:,(size_eski+1):n+2)=0;

  %son_noktalar(:,(size_eski+1):n+2)=0;

  soncozumler=[soncozumler;problem,run_say,final_solutions(I,:)];

      %
son_noktalar=[son_noktalar;problem*ones(ng,1),run_say*ones(ng,1),final_solutions];

  else

        soncozumler=[soncozumler;problem,run_say,final_solutions(I,:)];


 %son_noktalar=[son_noktalar;problem*ones(ng,1),run_say*ones(ng,1),final_solutions]
;

    end

   end

cpu_zamanlari=[cpu_zamanlari;problem*ones(total_runs,1),cpuzamani];

ort_cpu_zamani=[ort_cpu_zamani;problem,sum(cpuzamani)/total_runs];

cntrl=problem==sapmalar(:,1);

ort_sapma_problem=[ort_sapma_problem;problem,sum(sapmalar(:,3).*cntrl)/total_runs
];

end
```

```
%profile report


ort_cpu_zamani,ort_sapma_problem

G_ort_cpu_zamani=sum(ort_cpu_zamani(:,2))/size(ort_cpu_zamani,1);

G_ort_sapma=sum(ort_sapma_problem(:,2))/size(ort_sapma_problem,1);

Par_List={'n','m','ng','np','Rrd','iter','mmin','total_runs','obtm','tolf','Maks_time','n_g_grd'
};

Par_deger=[n,m,ng,np,Rrd,iter,mmin,total_runs,obtm,tolf,Maks_time,n_g_grd];


akis_kayded(enyisonuclar,soncozumler,sapmalar,ort_sapma_problem,ort_cpu_zamani,n
p,ng,iter,mmin,Par_List,Par_deger);

clear I aralik cntrl ed run_say n_g_grd ng n_p_min Rrd iter psin mmin total_runs tolf n
m np ardisik

%[y1,I2]=sort(son_noktalar(:,3:22),2);


function [bomba,b_uy,cpuzamani] =
GEM_AKIS_PER5(n,m,ng,np,Rrd,iter,mmin,total_runs,obtm,tolf,Maks_time,ng1)

%UNTİTLED Summary of this function goes here

%   Detailed explanation goes here

t=cputime;

%yayilmalar=[];%P_bas=1;

%final_solutions=[];%tabu=zeros(ng,n);ng1=ng;

%for runs=1:total_runs

  tabu=zeros(ng,n);

   %t1=cputime;

   %Le=(2*n)/ng;

   Re=((1*n)^0.5)/ng;  %disp(['run No.:' num2str(runs)])*n^2

   %Le_bas=Le;P=P_bas;
```

```
 % ng tane bomba üret
[bomba yayilma]=NEHbas();
bomba=repmat(bomba,ng,1);
 %bomba=repmat(randperm(n),ng,1);
 o_bomba=1;
 red=0;
 while o_bomba< ng
    bomba(o_bomba+1,:)=randperm(n);
   % d1=zeros(1,o_bomba-1);
    for d=1:o_bomba, %uzaklık hesaplar
       d1=(sum(((bomba(d,:)-bomba(o_bomba+1,:))./n).^2))^0.5;
       if d1<=Re;break,end;
    end
    %cnd=d1>=Re;
    if d1>=Re%~(ismember(0,cnd))
       o_bomba=o_bomba+1;
       red=0;
    else %bombalar uygun şekilde yerleştitrlemezse yarı capı küçültmekte
       red=red+1;
       if red>=o_bomba/ng*n*10;
          Re=Re*0.99;
          red=0;
       end
    end
 end %bomba üretimi bitti
 %bomba(ng,:)=[5,3,20,11,8,6,9,4,17,13,7,19,2,10,12,15,16,1,18,14];
```

```
Re_bas=Re;
b_uy=uygunluk_yayilma_per(bomba);
eniyisay=zeros(ng,1);
%i=0;
%İterasyonlar
for i=1:iter
%while (cputime-t1)<n*m*0.030   %n*m/10
  %i=i+1;
  [b_uy,I]=sort(b_uy);
  bomba=bomba(I,:);
  eniyisay=eniyisay(I);
  %ilerleme olmassa durdurma
  %if b_uy(1)<eniyi, eniyisay=0; eniyi=b_uy(1);else eniyisay=eniyisay+1;end
  %if eniyisay==200; break, end

  if Maks_time<=(cputime-t); break, end
  %sayaç son
 if b_uy(1)==obtm, break, end
  %yerel arama
  %if i>10;ng1=10;end
  if i<600 && mod(i,1)==0
     for iris=1:ng1,

[bomba(iris,:),b_uy(iris),tabu]=yerel_ris1_per_insert_NEHT(bomba(iris,:),b_uy(iris),n,t
abu,iris,i);%iris,i,

     end
```

```
        end
        if i>=2 && mod(i,1)==0
            for iris=1:10;%10,

%[bomba(iris,:),b_uy(iris),tabu]=yerel_ris1_per_insert_NEHT1(bomba(iris,:),b_uy(iris)
,n,tabu,iris,i);
              [bomba(iris,:),b_uy(iris)]=yerel_neht1(bomba(iris,:),b_uy(iris),n);
            end
        end
        %yerel arama bitti
       for bom=1:ng %her bomba için şarapnel üretir
          X01=bomba(bom,:);
          srpnl=repmat(X01,np,1);
          rn=randi(n,np,2);

         for i1=1:np
           srpnl(i1,rn(i1,1))=X01(rn(i1,2));
           srpnl(i1,rn(i1,2))=X01(rn(i1,1));

         end
             s_uy=uygunluk_yayilma_per(srpnl);
            [min_s_uy,K]=sort(s_uy);
       %uzaklıkları ölçer uygun uzaklıkta olmayanları tekar bulur
       oi=1;o_srpnl=0;
      while o_srpnl==0
       if min_s_uy(oi)<b_uy(bom)&&bom>1,
```

```
            for d=1:bom-1 %uzaklık hesaplar
               d1=(sum(((bomba(d,:)-srpnl(K(oi),:))./n).^2))^.5;
               %d2(d)=sum(d1.*conj(d1)).^.5;
               if d1<Re;break, end;
            end
            if d1>=Re; o_srpnl=1;else oi=oi+1;end
     else break,
     end
        %o_srpnl=1;red=0;
    end
      % end % bir bomba için şarapnel üretimi bitti
       %
        if min_s_uy(oi)<b_uy(bom); b_uy(bom)=min_s_uy(oi); bomba(bom,:)=srpnl(K(oi),:);eniyisay(bom)=0;
        else eniyisay(bom)=eniyisay(bom)+1;
        end
          if eniyisay(bom)>10 && bom>1;bomba(bom,:)=randperm(n); b_uy(bom)=uygunluk_yayilma_per(bomba(bom,:));eniyisay(bom)=0; end %[bomba(bom,:) b_uy(bom)]=yerel_neht1(bomba(bom,:),b_uy(bom),n)
      end %bütün bombaların şarapnel üretimi ve bombaların uygun yere taşınması bitti
      %pts güncelleme ve şarapnel dağıtımı yapılmalı
      Re=Re_bas/Rrd^(i/iter);
      %P=P_bas/log2((30/iter)*i+1)/2;    %3^(i/iter);
      end %iterasyon bitti
%yayilmalar=[yayilmalar;b_uy];
%final_solutions=[final_solutions;bomba];
%end
```

```
cpuzamani=cputime-t;
end

function [x min_yay]=NEHbas()

global P0  ;%cizelgelemeproblemi;
t=cputime;

%P0=TEXTREAD(problemadlari(problem,:))';
%obtm=obtm1(problem,2);
[n,m]=size(P0);
[P_sum I]=sort(sum(P0,2),'descend');
x=I(1);
for k=2:n
   x0=I(k);
   yayilma=NEHT_yayilma3(x,x0);
   [min_yay i]=min(yayilma);
   if i==1;
      x=[x0,x];
   elseif i==k,
         x=[x,x0];
   else
      x=[x(1:i-1),x0,x(i:k-1)];
   end
end
t=cputime-t;
```

```
%akis_kayded(enyisonuclar,soncozumler,sapmalar,ort_sapma_problem,ort_cpu_zamani
,np,ng,iter,mmin);


function [bomba,bomuy,tabu] =
yerel_ris1_per_insert_NEHT1(bomba,bomuy,n,tabu,iris,m)

t=cputime;

if ismember(bomba,tabu,'rows')%bomba==tabu(iris,:),

   [IR bomuy1]=yerel_neht(bomba,bomuy,n);

else

   IR=bomba;bomuy1=bomuy;

end,

Icur=IR;Ibest=IR;

h=1;i=1;tabu1=[];

while i<=n

     h=mod(h,n);if h==0,h=n;end

     in= Icur==IR(h);

     Icur(in)=[];


   yayilma=NEHT_yayilma3(Icur,IR(h));

   [buy_test test_in]=min(yayilma);



     %[buy_test test_in]=min(uygunluk_yayilma_per(I1));

     if buy_test<=bomuy1

       if test_in==1;

          Ibest=[IR(h),Icur];
```

```
        elseif test_in==n-1,
            Ibest=[Icur,IR(h)];
        else
        Ibest=[Icur(1:test_in-1),IR(h),Icur(test_in:n-1)];
        end;
        bomuy1=buy_test;tabu1=[tabu1;Ibest];if (cputime-t)>=0.01*n*m,i=n+1;end
%bomuy1,t,
      else
       if ismember(Ibest,tabu1,'rows');i=i+1;else i=1;end
      %i=i+1;
      end
      h=h+1;
      Icur=Ibest;
end
if bomuy1<bomuy
   bomuy=bomuy1;bomba=Ibest;
else
tabu(iris,:)=Ibest;
end
zaman=cputime-t;


function [bomba bomuy] = yerel_neht1(bomba,bomuy,n)
%UNTİTLED2 Summary of this function goes here
%   Detailed explanation goes here
sil=5;
x=bomba;
```

```
rn=randi(n,1,sil);
rn=unique(rn);
I=x(rn);
x(rn)=[];
k1=size(x,2)+1;k2=1;
for k=k1:n
   x0=I(k2);
   yayilma=NEHT_yayilma3(x,x0);
   [min_yay,i]=min(yayilma);
   if i==1;
      x=[x0,x];
   elseif i==k,
         x=[x,x0];
   else
      x=[x(1:i-1),x0,x(i:k-1)];
   end
   k2=k2+1;
end
if min_yay<bomuy
  bomuy=min_yay;bomba=x;
%else
  %bomuy=bomuy;bomba=bomba;
end
end
```

KİŞİSEL YAYIN VE ESERLER

Özkale, C., Baynal, K., ve **Öztürk , T.** (2023). Permütasyon Akış Tipi Çizelgeleme Probleminin El Bombası Patlatma Metodu ile Çözümü. *Pamukkale Üniversitesi İşletme Araştırmaları Dergisi*, *10(2*), 491-510.

Öztürk, T., ve Baynal, K. (2013). Akış Tipi Çizelgeleme Problemlerinin El Bombası Patlatma Metodu ile Çözümü. *International IIE Conference ve YA/EM 2013 Kongresi*, (s. 116-117). İstanbul.

Öztürk, T., Baynal, K., ve Aladağ, Z. (2011). Sağlık Tesislerinin ANP Tekniği Kullanılarak Planlanması. 12th International *Sympozium on Econometrics, Statistics and Operation Research*, (s. 430-439). Denizli.

ÖZGEÇMİŞ

İlk ve orta öğretimini Ordu' da tamamladı. 2004 yılında Kocaeli Üniversitesi Endüstri Mühendisliği Bölümü'nden mezun oldu. 2006 yılında askerlik görevini tamamladıktan sonra Kocaeli Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Bölümü'nde yüksek lisansa başladı. 2008 yılında yüksek lisans programını tamamdı.

2005 yılında profesyonel iş hayatına Sağlık Bakanlığında başladı. 2011 yılına kadar bu bakanlığın merkez ve taşra teşkilatında farklı görevlerde bulundu. 2011 yılında Türk Akreditasyon Kurumunda çalışmaya başladı. Kurum adına; çok sayıda laboratuvar ve belgelendirme kuruluşunun denetimine denetçi ve ekip lideri olarak katıldı, akreditasyon standartları, iyi laboratuvar uygulamaları, risk bazlı denetim teknikleri vb. konularda eğitimler verdi, Filistin ve Özbekistan'da yürütülen projelerde görev aldı. Ayrıca Kurumu temsilen Avrupa Akreditasyon Birliği adına bazı akreditasyon kurumlarının eş değerlendirme denetimlerine eş değerlendirici olarak katıldı. 2018-2020 yıllarında Personel Akreditasyon Başkanlığı görevinde bulundu. Halen Türk Akreditasyon Kurumunda Akreditasyon uzmanı olarak çalışma hayatına devam etmektedir.