

175813

**ÇOK-AMAÇLI AKIŞ TİPİ ÇİZELGELEME PROBLEMİ İÇİN
KARINCA KOLONİLERİ ENİYİLEME**

**DOKTORA TEZİ
Y. Müh. Betül YAĞMAHAN
(507992104)**

**Tezin Enstitüye Verildiği Tarih : 16 Mayıs 2005
Tezin Savunulduğu Tarih : 26 Temmuz 2005**

Tez Danışmanı :

Yrd.Doç.Dr. Mehmet Mutlu YENİSEY

Diğer Jüri Üyeleri

Prof.Dr. Füsun ÜLENGİN (İTÜ)

Prof.Dr. Erdal EMEL (U.Ü.)

Yrd.Doç.Dr.Yusuf İlker TOPCU (İTÜ)

Prof.Dr.Alpaslan FIĞLALI (K.Ü.)

TEMMUZ 2005

ÖNSÖZ

Son yıllarda yeni uygulamalara olan talep ve artan bilgisayar gücü, NP tipi birleşik eniyileme problemleri üzerine olan ilgiyi arttırmıştır. Bu tez kapsamında da NP-zor tipi problem den biri olan çok-amaçlı akış tipi çizelgeleme problemi ele alınmıştır. Bu probleme çözüm sağlamak üzere yeni bir metasezgisel olan karınca kolonileri eniyileme tekniğı önerilmektedir.

Önerilen tekniğın problem açısından performansını değerlendirebilmek amacıyla kapsamlı bir deneysel çalışma yapılmıştır. Ayrıca, çok-amaçlı akış tipi probleminin çözümü için önerilen karınca kolonileri eniyileme algoritması ile elde edilen çözüm, bugüne kadar aynı problem için literatürde mevcut temel teknikler ile elde edilen çözümler ile karşılaştırılmış ve bu tekniklere göre sağladığı avantajlar ortaya konmuştur.

Bu çalışmanın her aşamasında değerli bilgilerini ve yardımlarını esirgemeyen tez danışmanım Sayın Yrd.Doç.Dr.Mehmet Mutlu YENİSEY' e; maddi destek sağlayan Türkiye Bilimsel ve Teknik Araştırma Kurumuna (TÜBİTAK); bu zorlu çalışmaya sabırla katlanan, beni cesaretlendiren aileme ve destek olan herkese çok teşekkür ederim.

Temmuz 2005

Betül YAĞMAHAN

İÇİNDEKİLER

KISALTMALAR	vii
TABLO LİSTESİ	ix
ŞEKİL LİSTESİ	xii
ÖZET	xiv
SUMMARY	xvi
1. GİRİŞ	1
1.1 Birleşik Eniyileme Problemi	2
1.2 Karmaşıklık Teorisi	2
1.2.1 Algoritma Karmaşıklığı	2
1.2.2 Problem Karmaşıklığı (P ve NP Sınıfları)	4
1.3 Birleşik Eniyileme Problemleri Çözüm Yöntemleri	4
1.3.1 Kesin Çözümler	5
1.3.2 Yaklaşık Çözümler	5
1.3.2.1 Geliştirici Algoritmalar	5
1.3.2.2 Yerel Arama Algoritması	5
2. METASEZGİSEL YAKLAŞIMLAR	8
2.1 Metasezgisellerin Sınıflandırılması	9
2.2 Yörünge Yöntemleri	11
2.2.1 Tavlama Benzetimi	11
2.2.2 Yasak Arama	12
2.2.3 Keşfedici Yerel Arama Yöntemleri	12
2.2.3.1 Hırslı Rasgele Adaptif Arama Yordamı	12
2.2.3.2 Değişken Komşuluk Arama	14
2.2.3.3 Rehber Yerel Arama	14
2.2.3.4 Ardaşık Yerel Arama	16
2.3 Nüfus-Tabanlı Yöntemler	17
2.3.1 Evrimsel Hesaplama	17
2.3.1.1 Genetik Algoritmalar	18

2.3.1.2	Evrimsel Programlama	21
2.3.1.3	Evrimsel Stratejiler	21
2.3.2	Dağıtarak Arama ve Yeniden Yol İlişkilendirme	22
2.3.3	Dağılım Tahmini Algoritmaları	23
2.3.4	Diferansiyel Gelişim	23
2.3.5	Memetik Algoritmalar	26
2.3.6	Küme Zekası Algoritmaları	26
2.3.6.1	Karınca Kolonileri Eniyileme	27
2.3.6.2	Partikül Küme Eniyileme	28
2.3.7	Yapay Bağışıklık Sistemleri	30
2.4	Metasezgisellerin Melezleştirilmesi	36
3.	KARINCA KOLONİLERİ ENİYİLEME	37
3.1	KKE Metasezgiselinin Temel Özellikleri	37
3.2	Problem Gösterimi	38
3.3	Karıncaların Davranışları (Biyolojik Esaslar)	39
3.4	Gerçek ve Yapay Karıncaların Benzerlik ve Farklılıkları	43
3.5	KKE Metasezgiseli	44
3.6	KKE Metasezgiseli Uygulamaları	46
3.7	Mevcut KKE Algoritmaları	52
3.7.1	İlk KKE Algoritması: Karınca Sistemi	52
3.7.2	Karınca Koloni Sistemi	57
3.7.3	Ant-Q Algoritması	60
3.7.4	Max-Min Karınca Sistemi	60
3.7.5	Sıralamaya Dayalı Karınca Sistemi	62
4.	AKIŞ TİPİ ÇİZELGELEME PROBLEMİ	63
4.1	Basit Tanımlamalar	63
4.2	Amaçlar	64
4.3	Matematiksel Formül	66
4.4	Problemin Karmaşıklığı	67
4.5	Akış Tipi Çizelgeleme Problemi Çözüm Algoritmaları	67
4.5.1	Kesin Çözüm Algoritmaları	68
4.5.1.1	Johnson Algoritması	68
4.5.1.2	Revize Johnson Algoritması	69
4.5.1.3	Dal-Sınır Algoritması (Ignall-Schrage Algoritması)	69
4.5.2	Yaklaşık Çözüm Algoritmaları	71

4.5.2.1	Palmer Sezgiseli	71
4.5.2.2	CDS Sezgiseli	71
4.5.2.3	Gupta Sezgiseli	71
4.5.2.4	NEH Sezgiseli	72
4.5.3	Diğer Algoritmalar	72
4.6	İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Problemi	75
4.7	İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Problemi Çözüm Algoritmaları	76
5.	AKIŞ TİPİ ÇİZELGELEME PROBLEMİ İÇİN KKE YAKLAŞIMI	79
5.1	Problemin Tanımı ve Çözüm Yöntemi	79
5.2	Literatür Araştırması	80
5.3	Amaçlar	82
5.4	Problemin Gösterimi	83
5.5	Yapay Karıncaların Özellikleri	84
5.6	Otokatalitik Süreç	84
5.7	Yasaklar Listesi	85
5.8	Karıncı Sayısı	85
5.9	Geçiş Olasılığı	85
5.10	Pheromone İzi Güncelleme Kuralları	86
5.11	Sezgisel Bilgi	87
5.12	Aday Liste	88
5.13	Yerel Geliştirme Yöntemleri	89
5.14	Geliştirilen KKE Yaklaşımının Genel Yapısı	90
6.	DENEYSEL ÇALIŞMALAR	93
6.1	Tek-Amaçlı Durum için Denemeler ve Hesaplama Sonuçları	93
6.1.1	Test Problemleri	93
6.1.2	Başlangıç Çözümü ve Parametre Değerleri	94
6.1.3	Parametre Analizi	96
6.1.3.1	Geçiş Kuralı Parametre Analizi	96
6.1.3.2	α ve β Parametrelerinin Çözümüne Etkisi	104
6.1.3.3	pl ve pg Parametrelerinin Çözümüne Etkisi	106
6.1.3.4	İterasyon Sayısı Parametresinin Çözümüne Etkisi	108

6.1.3.5	Karınca Sayısı Parametresinin Çözümüne Etkisi	112
6.1.4	Karıncalar Arasındaki İşbirliğinin Çözümüne Etkisi	114
6.1.5	Geçiş Olasılığının Çözümüne Etkisi	116
6.1.6	Pheromone İzi Güncelleme Kuralının Çözümüne Etkisi	119
6.1.7	Sezgisel Bilginin Önemi ve Çözümüne Etkisi	124
6.1.8	Aday Liste Kullanımının Çözümüne Etkisi	126
6.1.9	Yerel Geliştirme Yönteminin Çözümüne Etkisi	129
6.2	Performans Değerlendirmesi	130
6.3	Çok-Amaçlı Durum için Denemeler ve Hesaplama Sonuçları	132
7.	SONUÇLAR ve TARTIŞMA	142
	KAYNAKLAR	146
	EKLER	162
	ÖZGEÇMİŞ	291



KISALTMALAR

KKE	: Karınca Kolonileri Eniyileme (Ant Colony Optimization)
AS	: Karınca Sistemi (Ant System)
TSP	: Gezgin Satıcı Problemi (Traveling Salesman Problem)
QAP	: Kuadratik Atama Problemleri (Quadratic Assignment Problem)
CO	: Birleşik Eniyileme (Combinatorial Optimization)
P	: Polinomsal Problem (Polynomial Problem)
NP	: Deterministik-Olmayan Polinomsal Problem (Non-Deterministic Polynomial Problem)
SA	: Tavlama Benzetimi (Simulated Annealing)
TS	: Yasak Arama (Tabu Search)
GRASP	: Hırslı Rasgele Adaptif Arama Yordamı (Greedy Randomized Adaptive Search Procedures)
VNS	: Değişken Komşuluk Arama (Variable Neighborhood Search)
GLS	: Rehber Yerel Arama (Guided Local Search)
ILS	: Ardaşık Yerel Arama (Iterated Local Search)
EC	: Evrimsel Hesaplama (Evolutionary Computation)
GA	: Genetik Algoritmalar (Genetic Algorithms)
EP	: Evrimsel Programlama (Evolutionary Programming)
ES	: Evrimsel Stratejiler (Evolutionary Strategies)
SS	: Dağıtarak Arama (Scatter Search)
PR	: Yeniden Yol İlişkilendirme (Path Relinking)
EDA	: Dağılım Tahmini Algoritmaları (Estimation of Distribution Algorithms)
DE	: Diferansiyel Gelişim (Differential Evolution)
MA	: Memetik Algoritmalar (Memetic Algorithms)
PSO	: Partikül Küme Eniyileme (Particle Swarm Optimization)
AIS	: Yapay Bağışıklık Sistemleri (Artificial Immune Systems)
IS	: Bağışıklık Sistemi (Immune System)
ANN	: Yapay Sinir Ağları (Artificial Neural Network)
FIFO	: İlk Giren İlk Çıkar Stratejisi (First In First Out)
RCL	: Sınırlandırılmış Aday Listesi (Restricted Candidate List)
JSS	: Atölye Çizelgeleme (Job Shop Scheduling)
ACS	: Karınca Koloni Sistemi (Ant Colony System)
MMAS	: Max-Min Karınca Sistemi (Max-Min Ant System)
AS_{rank}	: Sıralamaya Dayalı Karınca Sistemi (Rank Based Ant System)
AS_{elite}	: Elitist Stratejisine Dayalı Karınca Sistemi
RL	: Güçlendirme Öğrenimi (Reinforcement Learning)
SPT	: En Kısa İşlem Süresi Kuralı (Shortest Processing Time)
LPT	: En Uzun İşlem Süresi Kuralı (Longest Processing Time)
CDS	: Campbell Dubek Smith Sezgiseli
NEH	: Namaz Ensore Ham Sezgiseli
HC	: Ho Chang Sezgiseli
MD	: Katlı İniş (Multiple Descent)

API	: Bitişik Komşuluk Değişirme (Adjacent Pairwise Interchange)
NS	: Komşuluk Arama (Neighbourhood Search)
PGK	: Pheromone İzi Güncelleme Kuralı
TSPT	: En Kısa İşlem Süresi Toplamı (Total Shortest Processing Time)
GJI	: Gupta İş İndeksi
SPIRIT	: Entegre Edilmiş Yasak Arama Tekniği ile Yeni bir Çözüm Oluşturan Sıralama Problemi (Sequencing Problem Involving a Resolution by Integrated Taboo Search Techniques)
PACO	: Rajendran Zeigler KKE Sistemi
EDD	: En Erken Teslim Tarihi Kuralı (Earliest Due Date)



TABLO LİSTESİ

	<u>Sayfa No</u>
Tablo 3.1. KKE Algoritmaları Çalışmaları-1.....	48
Tablo 3.2. KKE Algoritmaları Çalışmaları-2.....	50
Tablo 3.3. KKE Metasezgiseli Uygulamaları.....	51
Tablo 4.1. Toplam Tamamlanma Zamanı Amacı için Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları.....	73
Tablo 4.2. Farklı Amaçlar için Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları.....	75
Tablo 4.3. İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Çalışmaları.....	77
Tablo 6.1. İlk Çözüm için Algoritma Performans Değerlendirmesi.....	95
Tablo 6.2. Taguchi Analizinde Kullanılan Faktör ve Seviyeleri.....	100
Tablo 6.3. Taguchi Analizi L8 Ortogonal Diziler.....	100
Tablo 6.4. L8 Tablosuna Bağlı Deney Tasarımı.....	100
Tablo 6.5. En İyi Değerlere Göre Deney Tasarımı Sonuçları.....	101
Tablo 6.6. Ortalama Değerlere Göre Deney Tasarımı Sonuçları.....	102
Tablo 6.7. En Kötü Değerlere Göre Deney Tasarımı Sonuçları.....	102
Tablo 6.8. Parametre Analizi Sonrası Algoritma Performans Değerlendirmesi.....	103
Tablo 6.9. Problem Hacmi Bazında Parametre Analizi Öncesi ve Sonrası Ortalama Değerler.....	104
Tablo 6.10. α ve β Parametrelerinin En İyi Değere Göre Çözüm Etkisi.....	105
Tablo 6.11. α ve β Parametrelerinin Ortalama Değere Göre Çözüm Etkisi.....	105
Tablo 6.12. pl ve pg Parametrelerinin En İyi Değere Göre Çözüm Etkisi....	107
Tablo 6.13. pl ve pg Parametrelerinin Ortalama Değere Göre Çözüm Etkisi.....	107
Tablo 6.14. İterasyon Sayısının Çözüm Etkisi.....	109
Tablo 6.15. Karınca Sayısının Çözüm Etkisi.....	113
Tablo 6.16. Karıncalar Arası İşbirliğinin Çözüm Etkisi.....	114
Tablo 6.17. Problem Hacmi Bazında Geçiş Olasılıklarının Çözüm Etkisi....	118
Tablo 6.18. Problem Hacmi Bazında Pheromone İzi Güncelleme Kurallarının (AS, AS _{elite} , ACS _{ib} , ACS _{gb}) Çözüm Etkisi.....	123
Tablo 6.19. Problem Hacmi Bazında Pheromone İzi Güncelleme Kurallarının (ANTQ, <i>MMAS</i> , AS _{rank}) Çözüm Etkisi.....	123
Tablo 6.20. Problem Hacmi Bazında Sezgisel Bilginin Çözüm Etkisi.....	126
Tablo 6.21. Aday Liste Büyüklüğünün Çözüm Etkisi.....	127
Tablo 6.22. Aday Liste Kullanımının Çözüm Etkisi.....	128
Tablo 6.23. Problem Hacmi Bazında Yerel Geliştirme Yöntemlerinin Çözüm Etkisi.....	129
Tablo 6.24. Algoritmanın Toplam Tamamlanma Zamanı Amacı için Performans Değerlendirmesi.....	131
Tablo 6.25. Toplam Tamamlanma Zamanı Amacı için Algoritmaların Performans Sonuçları.....	137

Tablo 6.26.	Toplam Akış Zamanı Amacı için Algoritmaların Performans Sonuçları.....	138
Tablo 6.27.	Toplam Aylak Kalma Zamanı Amacı için Algoritmaların Performans Sonuçları.....	139
Tablo 6.28.	Tüm Amaçların Toplamı için Algoritmaların Performans Sonuçları.....	140
Tablo A.1.	Tek Amaçlı Akış Tipi Çizelgeleme Çalışmaları.....	163
Tablo B.1.	İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Çalışmaları.....	191
Tablo C.1.	İlk Çözüm için Algoritma Deneme Sonuçları.....	209
Tablo C.2.	İlk Çözüm için Algoritma Performans Sonuçları.....	210
Tablo D.1.1.	α ve β Parametreleri Eğilim Belirleme Deneme Sonuçları.....	211
Tablo D.1.2.	pl , pg ve q_0 Parametreleri Eğilim Belirleme Deneme Sonuçları..	213
Tablo D.2.1.	α Parametresi Eğilim Belirleme Bağlı Hata Sonuçları.....	215
Tablo D.2.2.	β Parametresi Eğilim Belirleme Bağlı Hata Sonuçları.....	216
Tablo D.2.3.	pl Parametresi Eğilim Belirleme Bağlı Hata Sonuçları.....	217
Tablo D.2.4.	pg Parametresi Eğilim Belirleme Bağlı Hata Sonuçları.....	218
Tablo D.2.5.	q_0 Parametresi Eğilim Belirleme Bağlı Hata Sonuçları.....	219
Tablo D.3.1.	α Parametresi Değerini Belirleme Deneme Sonuçları.....	220
Tablo D.3.2.	β Parametresi Değerini Belirleme Deneme Sonuçları.....	221
Tablo D.3.3.	pl Parametresi Değerini Belirleme Deneme Sonuçları.....	222
Tablo D.3.4.	pg Parametresi Değerini Belirleme Deneme Sonuçları.....	223
Tablo D.3.5.	q_0 Parametresi Değerini Belirleme Deneme Sonuçları.....	224
Tablo D.4.1.	α Parametresi Değerini Belirleme Bağlı Hata Sonuçları.....	225
Tablo D.4.2.	β Parametresi Değerini Belirleme Bağlı Hata Sonuçları.....	225
Tablo D.4.3.	pl Parametresi Değerini Belirleme Bağlı Hata Sonuçları.....	226
Tablo D.4.4.	pg Parametresi Değerini Belirleme Bağlı Hata Sonuçları.....	226
Tablo D.4.5.	q_0 Parametresi Değerini Belirleme Bağlı Hata Sonuçları.....	227
Tablo D.5.	Taguchi Analizi Deneme Sonuçları.....	228
Tablo D.6.1.	Parametre Analizi Sonrası Algoritma Deneme Sonuçları.....	232
Tablo D.6.2.	Parametre Analizi Sonrası Algoritma Performans Sonuçları.....	233
Tablo D.7.	α ve β Parametreleri Etkisini Belirleme Deneme Sonuçları.....	234
Tablo D.8.	pl ve pg Parametreleri Etkisini Belirleme Deneme Sonuçları.....	238
Tablo D.9.	İterasyon Sayısı Parametresi Etkisini Belirleme Deneme Sonuçları.....	242
Tablo D.10.	Karınca Sayısı Parametre Etkisini Belirleme Deneme Sonuçları	246
Tablo E.1.	Karıncalar Arası İşbirliği Etkisini Belirleme Deneme Sonuçları.	248
Tablo F.1.1.	R-Prob Geçiş Kuralı Deneme Sonuçları.....	249
Tablo F.1.2.	PR-Prob Geçiş Kuralı Deneme Sonuçları.....	250
Tablo F.1.3.	Psum Geçiş Kuralı Deneme Sonuçları.....	251
Tablo F.1.4.	R-Eval Geçiş Kuralı Deneme Sonuçları.....	252
Tablo F.2.	Geçiş Kuralları Bağlı Hata Sonuçları.....	253
Tablo G.1.1.	AS İz Güncelleme Kuralı Deneme Sonuçları.....	254
Tablo G.1.2.	AS _{elite} İz Güncelleme Kuralı Deneme Sonuçları.....	255
Tablo G.1.3.	ACS _{ib} İz Güncelleme Kuralı Deneme Sonuçları.....	256
Tablo G.1.4.	ACS _{gb} İz Güncelleme Kuralı Deneme Sonuçları.....	257
Tablo G.1.5.	ANT-Q İz Güncelleme Kuralı Deneme Sonuçları.....	258
Tablo G.1.6.	MMAS İz Güncelleme Kuralı Deneme Sonuçları.....	259
Tablo G.1.7.	AS _{rank} Güncelleme Kuralı Deneme Sonuçları.....	260
Tablo G.2.	İz Güncelleme Kuralları (AS, AS _{elite} , ACS _{ib} , ACS _{gb}) Bağlı Hata Sonuçları.....	261

Tablo G.3.	İz Güncelleme Kuralları (ANTQ, $MMAS$, AS_{rank}) Bağlı Hata Sonuçları.....	262
Tablo H.1.1.	TSPT Sezgisel Bilgisi Deneme Sonuçları.....	263
Tablo H.1.2.	GJI Sezgisel Bilgisi Deneme Sonuçları.....	264
Tablo H.1.3.	SPIRIT Sezgisel Bilgisi Deneme Sonuçları.....	265
Tablo H.1.4.	Sezgisel Bilgi Kullanılmadığı Durumda Deneme Sonuçları.....	266
Tablo H.2.	Sezgisel Bilgi Kuralları Bağlı Hata Sonuçları.....	267
Tablo I.1.	Aday Liste Büyüklüğünü Belirleme Deneme Sonuçları.....	268
Tablo I.2.	Aday Listenin Kullanımının Etkisini Belirleme Deneme Sonuçları.....	269
Tablo J.1.1.	API Yerel Arama Yöntemi Deneme Sonuçları.....	270
Tablo J.1.2.	Interchange Yerel Arama Yöntemi Deneme Sonuçları.....	271
Tablo J.1.3.	Insert Yerel Arama Yöntemi Deneme Sonuçları.....	272
Tablo J.2.	Yerel Arama Yöntemleri Bağlı Hata Sonuçları.....	273
Tablo K.1.1.	20x5-10-20 Problem Boyutu için Algoritma Deneme Sonuçları.	274
Tablo K.1.2.	50x5-10-20 Problem Boyutu için Algoritma Deneme Sonuçları.	275
Tablo K.1.3.	100x5-10-20 Problem Boyutu için Algoritma Deneme Sonuçları.....	276
Tablo K.2.1.	20x5-10-20 Problem Boyutu için Algoritma Performans Sonuçları.....	277
Tablo K.2.2.	50x5-10-20 Problem Boyutu için Algoritma Performans Sonuçları.....	278
Tablo K.2.3.	100x5-10-20 Problem Boyutu için Algoritma Performans Sonuçları.....	281
Tablo L.1.	NEH ve HC Algoritmaları için Deneme Sonuçları.....	285
Tablo L.2.	KKE Algoritması için Deneme Sonuçları.....	286
Tablo L.3.	Problem Bazında Algoritma Bağlı Hata Sonuçları.....	289

ŞEKİL LİSTESİ

	<u>Sayfa No</u>
Şekil 1.1 : Zaman Karmaşıklığı.....	3
Şekil 1.2 : Yerel Arama Algoritması.....	6
Şekil 1.3 : İteratif Gelişim Algoritması.....	7
Şekil 2.1 : Tavlama Benzetimi Algoritması.....	11
Şekil 2.2 : Yasak Arama Algoritması.....	12
Şekil 2.3 : Hırslı Rasgele Adaptif Arama Yordamı.....	13
Şekil 2.4 : Hırslı Rasgele Çözüm Yapısı.....	13
Şekil 2.5 : Değişken Komşuluk Arama Algoritması.....	14
Şekil 2.6 : Rehber Yerel Arama Algoritması.....	15
Şekil 2.7 : Ardaşık Yerel Arama Algoritması.....	16
Şekil 2.8 : Evrimsel Hesaplama Algoritması.....	17
Şekil 2.9 : Genetik Algoritmalar.....	18
Şekil 2.10 : Dağıtarak Arama ve Yeniden Yol İlişkilendirme Algoritmaları	22
Şekil 2.11 : Dağılım Tahmini Algoritması.....	23
Şekil 2.12 : Diferansiyel Gelişim Algoritması.....	26
Şekil 2.13 : Memetik Algoritmalar.....	27
Şekil 2.14 : Partikül Küme Eniyileme Algoritması.....	30
Şekil 2.15 : Bağışıklık Sisteminde Ayırt Etme.....	31
Şekil 2.16 : Negatif Seçim Algoritması.....	32
Şekil 2.17 : Klonal Seçim, Çoğalma, Benzerlik Olgunlaştırma ve Hafıza Hücrelerinin Korunması.....	33
Şekil 2.18 : Klonal Seçim Algoritması.....	34
Şekil 2.19 : Bağışıklık Ağ Teorisi.....	34
Şekil 2.20 : Bağışıklık Ağ Algoritması.....	35
Şekil 3.1 : Engelle Karşılaşan Karıncaların Seçimi ve En Kısa Yolu Bulmaları.....	40
Şekil 3.2 : Tek Köprü Deneyi.....	41
Şekil 3.3 : Çift Köprü Deneyi.....	42
Şekil 3.4 : KKE Metasezgiseli.....	47
Şekil 3.5 : Karınca-Yoğunluk ve Karınca-Miktar Algoritmaları.....	55
Şekil 3.6 : Karınca-Çevrim Algoritması.....	56
Şekil 4.1 : Kuramsal Akış Tipi Sistemi.....	64
Şekil 4.2 : Genel Akış Tipi Sistemi.....	64
Şekil 4.3 : Akış Tipi Çizelgeleme Problemi için Gantt Şeması.....	65
Şekil 4.4 : Johnson Algoritması.....	68
Şekil 5.1 : Akış Tipi Çizge Modeli.....	83
Şekil 5.2 : Komşuluk Yapıları.....	90
Şekil 5.3 : KKE Yaklaşımının Genel Yapısı.....	91
Şekil 5.4 : Akış Tipi Çizelgeleme Problemi için KKE Algoritması.....	92
Şekil 6.1 : İlk Çözüm için Algoritma Ortalama Performans Değerlendirmesi.....	95

Şekil 6.2	: α Parametresi Eğilim Sonuçları.....	97
Şekil 6.3	: β Parametresi Eğilim Sonuçları.....	97
Şekil 6.4	: pl Parametresi Eğilim Sonuçları.....	98
Şekil 6.5	: pg Parametresi Eğilim Sonuçları.....	98
Şekil 6.6	: q_0 Parametresi Eğilim Sonuçları.....	99
Şekil 6.7	: Parametre Analizi Sonrası Algoritma Ortalama Performans Değerlendirmesi.....	103
Şekil 6.8	: Problem Hacmi Bazında Parametre Analizinin Etkisi.....	104
Şekil 6.9	: α ve β Parametrelerinin Etkisi.....	106
Şekil 6.10	: pl ve pg Parametrelerinin Etkisi.....	108
Şekil 6.11	: İterasyon Sayısının Çözümüne Etkisi.....	110
Şekil 6.12	: ta011 için İterasyon Sayısı Analizi.....	110
Şekil 6.13	: ta012 için İterasyon Sayısı Analizi.....	111
Şekil 6.14	: ta013 için İterasyon Sayısı Analizi.....	111
Şekil 6.15	: ta014 için İterasyon Sayısı Analizi.....	112
Şekil 6.16	: Karınca Sayısının Çözümüne Etkisi.....	113
Şekil 6.17	: Karıncalar Arası İşbirliğinin Çözümüne Etkisi.....	114
Şekil 6.18	: ta011 için Karıncalar Arası İşbirliği.....	115
Şekil 6.19	: ta012 için Karıncalar Arası İşbirliği.....	115
Şekil 6.20	: ta013 için Karıncalar Arası İşbirliği.....	116
Şekil 6.21	: ta014 için Karıncalar Arası İşbirliği.....	116
Şekil 6.22	: Problem Hacmi Bazında Geçiş Olasılıklarının Çözümüne Etkisi..	119
Şekil 6.23	: Problem Hacmi Bazında Pheromone İzi Güncelleme Kurallarının Çözümüne Etkisi.....	124
Şekil 6.24	: Problem Hacmi Bazında Sezgisel Bilginin Çözümüne Etkisi.....	126
Şekil 6.25	: Aday Liste Büyüklüğünün Çözümüne Etkisi.....	127
Şekil 6.26	: Aday Liste Kullanımının Çözümüne Etkisi.....	128
Şekil 6.27	: Problem Bazında Yerel Geliştirme Yöntemlerinin Çözümüne Etkisi.....	130
Şekil 6.28	: Algoritmanın Toplam Tamamlanma Zamanı Amacı için Ortalama Performans Değerlendirmesi.....	131
Şekil 6.29	: HC Sezgiseli.....	133
Şekil 6.30	: NEH Sezgiseli.....	134
Şekil 6.31	: Toplam Tamamlanma Zamanı Amacı için Algoritmaların Ortalama Performans Sonuçları.....	138
Şekil 6.32	: Toplam Akış Zamanı Amacı için Algoritmaların Ortalama Performans Sonuçları.....	139
Şekil 6.33	: Toplam Aylak Kalma Zamanı Amacı için Algoritmaların Ortalama Performans Sonuçları.....	140
Şekil 6.34	: Tüm Amaçların Toplamı için Algoritmaların Ortalama Performans Sonuçları.....	141

ÇOK-AMAÇLI AKIŞ TİPİ ÇİZELGELEME PROBLEMİ İÇİN

KARINCA KOLONİLERİ ENİYİLEME

ÖZET

Bu tez kapsamında, NP-zor tipi problem olarak bilinen permütasyon akış tipi çizelgeleme problemini çözmek için Karınca Kolonileri Eniyileme yaklaşımı önerilmektedir.

Permütasyon akış tipi çizelgeleme problemi, tüm makinelerde aynı sıraya sahip işlerin çizelgelenmesinden oluşur. Bir iş aynı anda en çok bir makine üzerinde işlenebilirken, bir makine aynı anda en çok bir iş üzerinde çalışabilir.

Günümüzde bir çok problem çok amaçlı olarak değerlendirilmektedir. Bu amaçla bu çalışmada akış tipi problemi için çok kullanılan toplam tamamlanma zamanı amacından başka, toplam akış zamanı ve toplam makine aylak kalma zamanını da göz önüne alacak şekilde çok-amaçlı akış tipi problemi ele alınmıştır.

Karınca kolonileri eniyileme, birleşik eniyileme problemleri için son zamanlarda kullanılan ve umut verici metasezgisel bir yaklaşımdır. Biyolojik tabanlı diğer sezgisel yöntemlere (genetik algoritmalar, evrimsel stratejiler, yapay sinir ağları, yapay bağışıklık sistemleri gibi) benzer şekilde, karınca algoritmaları, gerçek karınca kolonilerinden esinlenilerek oluşturulmuştur. Karınca kolonileri eniyileme metasezgiselinde, yapay karınca kolonisi eniyileme problemine iyi sonuçlar elde edebilmek için işbirliği yaparlar.

Bu çalışmanın ilk bölümünde, eniyileme problemi ve karmaşıklığından sözedilmiş, ayrıca çözüm yöntemleri tanımlanmıştır.

İkinci bölümde metasezgisel yaklaşımlar sınıflandırılmış ve özet olarak açıklanmıştır.

Üçüncü bölümde ise, karınca kolonileri eniyileme yaklaşımının özellikleri tanımlanmış ve algoritmanın çalışma prensibi ayrıntılı şekilde açıklanmıştır. Ayrıca literatürdeki çeşitli problemler için karınca kolonileri eniyileme yaklaşımı kullanılarak yapılan çalışmalar sınıflandırılmıştır.

Dördüncü bölümde, akış tipi çizelgeleme problemi tanımlanmış, özellikleri ve varsayımlarından bahsedilmiştir. Ayrıca, bu problemin çözümü için literatürdeki mevcut yöntemlerin amaç tipi açısından bir sınıflandırması yapılmıştır.

Beşinci bölümde ise, karınca kolonileri eniyileme yaklaşımının akış tipi çizelgeleme problemi için nasıl uyarlandığı gösterilmiştir. Ayrıca bu problem için çözüm sağlayacak bir karınca koloni sistemi algoritması önerilmiştir.

Son kısımda, akış tipi çizelgeleme problemi için önerilen karınca koloni sistemi algoritmasının iyi çözümler vermesini sağlamak üzere deneysel çalışmalar yapılmıştır. Bunun için öncelikle parametre analizi gerçekleştirilmiştir. En uygun

parametreleri tespit etmek üzere iki seviyeli bir deneysel tasarım uygulanmıştır. Daha sonra algoritmanın güçlenmesini sağlayacak mekanizmalar incelenmiş, her birinin çözüm performansı üzerindeki etkileri araştırılmıştır.

Bu çalışmalar sırasında, tek-amaçlı problem için 5, 10 ve 20 makine ve 20, 50 ve 100 işten oluşan 90 test problemi, çok-amaçlı problem için 5, 10 ve 15 makine ve 20, 30 ve 50 işten oluşan 18 test problemi üzerinde toplam 13930 adet deneme yapılmıştır.

Önerilen yöntem ile elde edilen çözümlerin performansı, bu problem için literatürde mevcut diğer tekniklerle elde edilen çözümlerin performansı ile karşılaştırılmıştır. Karşılaştırma sonuçları, önerilen algoritmanın bu problem için oldukça etkin olduğunu göstermiştir. Önerilen algoritma karşılaştırıldığı diğer algoritmalara göre de daha iyidir.



ANT COLONY OPTIMIZATION FOR MULTI OBJECTIVE FLOW SHOP SCHEDULING PROBLEM

SUMMARY

In this thesis, Ant Colony Optimization (ACO) approach is proposed to solve permutation flow shop scheduling problem which is known as NP-hard type.

Permutation flow shop scheduling problem consists of scheduling works with same order at all machinery. The work can be processed on at most one machine; meanwhile one machine can process at most one work.

Today, many problems are considered to be of with multi objectives type. Hence, this study investigates multi objective flow shop problem which considers total machine idle time and total flow time, besides the makespan objective, which is common for flow shop problems.

Ant colony optimization is one of the most recent and promising metaheuristic for combinatorial optimization problems. Similar to other heuristics based on biological systems (such as genetic algorithms, evolutionary strategies, artificial neural network, artificial immune systems) ant algorithms have been inspired by the behavior of real ant colonies. In the ant colony optimization metaheuristic, a colony of artificial ants cooperate in finding good solutions to optimization problems.

In the first chapter of this study, optimization problem and its complexity are mentioned. In addition, solution methods are described.

In the second chapter, metaheuristic approaches are classified and explained summarily.

In the third chapter, characteristics of ant colony optimization approach are described and study principals are clarified in detail. Furthermore, studies using ant colony optimization approach for several problems in literature are classified.

In the fourth chapter, flow shop scheduling problem is defined; properties and assumptions are mentioned. Furthermore, a classification of existing methods based on objective types is given.

In the fifth chapter, how to adapt the ant colony optimization approach for flow shop scheduling problem is demonstrated. In addition, to solve this problem an ant colony system algorithm is proposed.

In the last chapter, an extensive experimental study to obtain a good solution using the proposed ant colony system algorithm for flow shop scheduling problem is presented. Therefore, following a parameter analysis a two-level experimental design is applied for fine-tuning of parameters. Later, each mechanism is examined to make algorithm strong and their influence on solution performance is searched.

During this study, a total of 13930 runs is carried on 90 test problems with 5, 10 and 20 machines and 20, 50 and 100 works for makespan objectives and 18 test problems with 5, 10 and 15 machines and 20, 30 and 50 works for multiple objectives.

Solution performance obtained here by the proposed algorithm is compared with solution performance obtained by the available methods reported in the literature. As a result, computational results show that proposed algorithm is more effective and better than other methods compared.



1. GİRİŞ

Karınca kolonileri eniyileme (KKE, Ant Colony Optimization-ACO), zor çözümlü birleşik eniyileme (Combinatorial Optimization-CO) problemlerinin çözümü için son zamanlarda önerilen metasezgisel (metaheuristic) bir yaklaşımdır. Karınca algoritmaları gerçek karıncalardan esinlenilerek oluşturulmuştur. Karınca koloni algoritmalarının ana fikri, karıncalar arasında iletişim ve geri besleme aracı olarak gerçek karıncalar tarafından kullanılan pheromone izlerini takip etmektir.

Karınca kolonileri eniyileme ilk olarak Dorigo ve diğ. (1991) tarafından ortaya atılmıştır. Algoritma olarak ilk örnek, gezgin satıcı problemi (Traveling Salesman Problem-TSP) olarak bilinen örnek bir uygulama için kullanılan karınca sistemidir (Ant System-AS) (Dorigo ve diğ., 1991a, 1991b, 1996; Colormi ve diğ., 1991, 1992). Bu başarısından hareketle, KKE metasezgiseli mevcut uygulamalar ve algoritmik değişiklikler için genel bir yapı olarak görülmektedir. KKE metasezgiselini kullanan algoritmalar, KKE algoritmaları olarak adlandırılır.

Farklı birleşik eniyileme problemlerinde kullanılan karınca kolonileri eniyilemenin önemli özellikleri şöyle sıralanabilir: çok yönlü (versitail), güçlü (robust) ve nüfus-tabanlı (population-based) yaklaşım olması, dağıtık hesaplama (distributed computation) yapabilmesi ve geliştirici hırslı (constructive greedy) sezgisel olmasıdır. Aynı zamanda, sezgisel bilginin kullanımı, karıncaların daha umut verici çözüme ulaşmalarında yol gösterici olabilir. Daha da önemlisi, karınca araştırmaları, algoritmaların ileriki iterasyonlarında çözüm yapısını öğrenmeyi destekleyici bir şekilde kullanılabilir.

KKE algoritmalarının şu anki uygulamaları, iki önemli problem sınıfı olan statik ve dinamik birleşik eniyileme problemleri için yapılmaktadır. Bu problemlerin çözümü için kullanılan KKE algoritmaları geniş çerçevede birbirine oldukça benzerdir, ancak detayda önemli farklılıklar gösterir.

1.1 Birleşik Eniyileme Problemi

Birçok eniyileme probleminin çözümü, amaçları “en iyi” biçimde sağlayan bir değişkenler kümesi aramaktadır. Bu değişkenler kümesi iki kısma ayrılabilir: gerçek-değerli değişkenler (real-valued variables) ve kesikli (sürekli) değişkenler (discrete variables). İkinci kısımda yer alan problemler arasında, birleşik eniyileme problemleri olarak adlandırılan problemler ile karşılaşılır. Birleşik eniyileme problemlerinde karar değişkenleri kesikli, çözüm ise tamsayıların veya diğer kesikli nesnelerin bir kümesi veya sıralamasıdır.

Tanım 1.1 Bir birleşik eniyileme problemi $P = (S, f)$, şöyle tanımlanır:

- değişkenler kümesi $x = \{x_1, \dots, x_n\}$
- değişken alanı $D_1, \dots, D_n$
- değişkenler arasındaki kısıtlar
- en küçüklenecek bir f amaç fonksiyonu, burada $f : D_1 \times \dots \times D_n \rightarrow \mathbb{R}^+$

Tüm mümkün uygun atamaların kümesi:

$$S = \{s = \{(x_1, v_1), \dots, (x_n, v_n)\} \mid v_i \in D_i, s \text{ tüm kısıtları sağlar}\}$$

S , arama (ya da çözüm) alanı olarak adlandırılır. Kümedeki her eleman ise, aday çözüm olarak görülebilir. Birleşik eniyileme problemini çözmek için, $f(s^*) \leq f(s) \quad \forall s \in S$ olacak şekilde amaç fonksiyonunu en küçükleyen bir $s^* \in S$ çözümünü bulmak gerekir. s^* , (S, f) 'nin global eniyi çözümü ve $S^* \subseteq S$ kümesi, global eniyi çözüm kümesi olarak adlandırılır.

Gezgin satıcı problemi, kuadratik atama problemi (Quadratic Assignment Problem-QAP), ders programlama (timetabling) ve çizelgeleme problemi (scheduling problem) birleşik eniyileme problemi örnekleridir (Blum ve Roli, 2003).

1.2 Karmaşıklık Teorisi

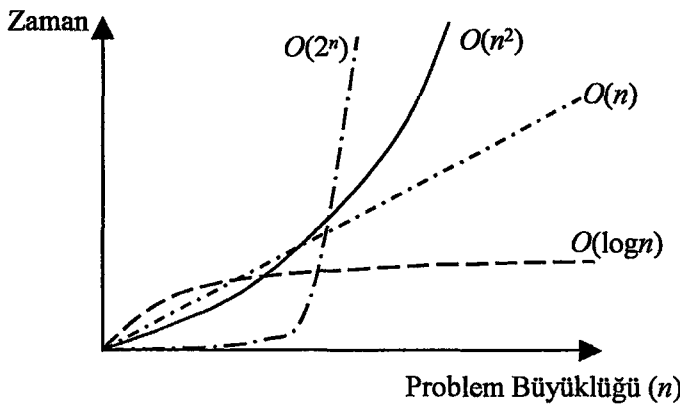
1.2.1 Algoritma Karmaşıklığı

Algoritma karmaşıklığı teorisi, çözülmesi kolay ve zor olan problemleri belirlemekle ilgili bir teoridir. Ayrıca bu teori, algoritmaların hesaplama açısından etkinliği ve

etkinsizliği ile de ilgilenmektedir. Bu açıdan, bir algoritmanın etkinliği bu algoritmaları yürütmek için gerekli hesaplama kaynakları ile tayin edilir ve işlem zamanı ve alanını kapsar. İşlem zamanı, algoritma girdi ve çıktı süreçlerini kapsayan adımların sayısıdır. Alan ise, algoritmayı çalıştırmak için gerekli hafıza büyüklüğüdür. Ancak algoritma karmaşıklığı teorisinde, algoritmaların etkinliği genellikle zaman karmaşıklığı (time complexity) açısından ifade edilir.

Zaman karmaşıklığı, problemin büyüklüğüne göre gerekli olan girdi büyüklüğü ile ilgilidir. Dahası, bir algoritma için zaman karmaşıklığı, en-kötü durumdaki davranışı ile tanımlanır. Bu, n büyüklüğüne sahip bir girdi için algoritmanın gerçekleştirmesi beklenen temel işlemlerin maksimum sayısıdır. Bir algoritmanın zaman karmaşıklığı, $O(g(n))$ notasyonu (Big-Oh notation) kullanılarak ifade edilir. Eğer, $n \geq 0$ için $|f(n)| \leq k \cdot |g(n)|$ olacak şekilde bir k sabiti varsa, bir $f(n)$ fonksiyonunun $O(g(n))$ olduğu söylenir. Diğer bir deyişle, $O(g(n))$, $g(n)$ ' den daha hızlı olmayan fonksiyonları gösterir ve $O(g(n))$ notasyonu, $g(n)$ ile sınırlandırılan algoritmaların zaman karmaşıklığını ifade eder. Şekil 1.1' de problem büyüklüklerine göre zaman karmaşıklığı görülmektedir.

Polinomsal bir fonksiyon ile tanımlanan (yani $O(4n)$, $O(n^3)$ vb.) bir zaman karmaşıklığına sahip algoritmalar etkin olarak ele alınır. Çünkü, önemli büyüklükteki girdilerle makul zamanda çalışırlar. Ancak, eğer algoritmaların zaman karmaşıklığı, üstel bir fonksiyon ile (yani $O(3^n)$, $O(n^{\log n})$ vb.) tanımlanıyorsa, algoritmalar etkin olarak ele alınmaz. Çünkü, sadece küçük büyüklüğe sahip girdiler için makul zamanda çalışabilirler, önemli büyüklükteki girdiler için uygulanamazlar.



Şekil 1.1 Zaman Karmaşıklığı

Bir problemin hesaplama karmaşıklığı, problemi çözmek için bulunabilen en iyi algoritma ile belirlenir. Eğer verilen bir problem için, bir polinomsal zamanlı algoritma bulunabilirse, problem kontrol edilebilirdir veya zor değildir. Ancak, problem için, böyle bir algoritma yoksa, yani sadece üstel zamanlı algoritmalar oluşturulabiliyorsa, problem, kontrol edilemezdir veya zordur (Silva, 2003).

1.2.2 Problem Karmaşıklığı (P ve NP Sınıfları)

Problemler polinomsal (Polynomial- P) ve deterministik olmayan polinomsal (Non-deterministic Polynomial- NP) problemler olmak üzere iki sınıfta toplanabilir. P sınıfı problemleri, etkin (polinomsal zaman) deterministik algoritmaların bulunabildiği tüm problemleri kapsar. NP sınıfı problemleri ise, problemin çözümü için deterministik olmayan polinomsal zamanlı algoritmaların bulunabildiği tüm problemleri kapsar. $P \subseteq NP$ olduğu açıktır, yani deterministik-olmayan algoritmalar, deterministik algoritmalarından daha güçlüdür. NP' de olan, ancak P' de olmayan bir algoritma P' ye taşınabilir. Ancak NP' den P' ye taşınamayacak problemler vardır. Yani, etkin bir algoritma bulunmayan ve NP' de olduğu bilinen bu problemler, NP -zor olarak tanımlanır.

NP problemlerinin önemli bir alt sınıfı, NP -tam sınıfıdır. NP' deki herhangi bir problem tatmin edilebilirlik (satisfiability) problemine indirgenebilir (tatmin edilebilirlik problemi, NP -tam olduğu ispatlanan ilk problem). Yani, tatmin edilebilirlik problemini çözmek için etkin bir çözüm varsa, NP' deki herhangi bir problem de etkin bir algoritma ile çözülebilir. Bir NP -tam problemini çözmek için etkin bir algoritma yoksa, NP' deki hiçbir problem için etkin bir algoritma yoktur ve bunların tümü kontrol edilemezdir. Birçok problemin NP -tam olduğu ispatlanmıştır.

60' ların sonunda, birçok birleşik problemin NP olduğu açık hale gelmişken, bunlardan sadece birkaçı P sınıfına aittir. Birleşik problemler, ayrıca NP -zor olarak tanımlanır. Yani, çözümü oldukça zordur (Silva, 2003).

1.3 Birleşik Eniyileme Problemleri Çözüm Yöntemleri

Birleşik eniyileme problemlerinin önemi nedeniyle, çözümü için birçok algoritma geliştirilmiştir. Bu algoritmalar, kesin ve yaklaşık algoritmalar olarak iki sınıfta toplanabilir.

1.3.1 Kesin Çözümler

Kesin algoritmalar, her sonlu büyüklüğe sahip birleşik eniyileme problemi için sınırlı bir zamanda bir eniyi çözümü bulmayı garanti eder. Kesin çözümler arasında doğrusal programlama, doğrusal olmayan programlama, tamsayı programlama, hedef programlama, dinamik programlama, şebeke programlama, sayma (enumerative) yöntemleri (dal-sınır, ışın arama (beam search)) sayılabilir.

1.3.2 Yaklaşık Çözümler

NP-zor olan birleşik eniyileme problemleri için, polinomsal zamanlı algoritmalar mevcut değildir. Kesin çözümler, üstel hesaplama süresi gerektirirler. Bu nedenle, birleşik eniyileme problemleri için yaklaşık çözümlerin kullanımı son 30 yıldır daha fazla önem kazanmıştır. Yaklaşık çözümlerde, önemli ölçüde kısalan zamanda iyi çözümler elde etmek amacıyla, eniyi çözümü bulma garantisinden fedakarlık edilir. Basit yaklaşık çözüm yöntemleri, geliştirici yöntemler ve yerel arama yöntemleri olarak ikiye ayrılabilir.

1.3.2.1 Geliştirici Algoritmalar

Geliştirici algoritmalar (hırslı yöntemler), birleşik eniyileme problemleri için kullanılır ve makul bir süre içinde kabul edilebilir iyi çözümler üretebilirler (Silva, 2003). Geliştirici algoritmalar, çok sayıda iterasyonla bir çözüm oluştururlar. Genellikle, birleşik eniyileme problemlerinde iterasyon sayısı değişken sayısına eşittir. Sezgisel algoritmalar, tam bir çözüm oluşturulana dek, her iterasyonda değişkenlerden birine bir değer atar. Sezgisel algoritma, çözümün uygunluğunu sağlayan değeri seçer ve önceden tanımlanmış bir kritere göre en iyi sonucu üretir (Blum ve Roli, 2003).

1.3.2.2 Yerel Arama Algoritması

Yerel arama algoritmaları, başlangıç çözümüyle başlar ve iteratif olarak o anki çözümün komşuluğundaki en iyi çözümle o anki çözümü yer değiştirmeye çalışır. Algoritma, yerel minimum bulunduğu durur. Algoritma, Şekil 1.2' de özetlenmiştir. Komşuluk şöyle tanımlanır:

Tanım 1.2. Bir komşuluk yapısı, her $s \in S'$ ye bir komşular kümesi $N(s) \subseteq S$ atayan bir $N : S \rightarrow 2^S$ fonksiyonudur. $N(s)$ ' ye, s ' nin komşuluğu adı verilir.

Bir komşuluk yapısının oluşturulması, bize yerel minimum çözümler kavramını tanımlamaya imkan tanır.

Tanım 1.3. N komşuluk yapısına göre bir yerel minimum çözüm, $\forall s \in N(\hat{s}) : f(\hat{s}) \leq f(s)$ olacak şekildeki $\hat{s}$ çözümdür (Blum ve Roli, 2003).

```
s ← BaşlangıçÇözümüOluştur()
repeat
    s ← Gelişim(N(s))
until Gelişim mümkün değil
```

Şekil 1.2 Yerel Arama Algoritması (Blum ve Roli, 2003)

Bir CO probleminde çözümler, bir sıralama veya bölümlene olarak gösterilebilir. Bu tip çözüm gösterimi k -değişim komşuluk yapısına izin verir. Yani, verilen bir sıralamada veya bölümde k elemanın değişimi bir komşuluk yapısını oluşturur. Yerel aramada hareket bir komşuluk çözümü oluşturmak amacıyla, o anki çözüm için oluşturulan komşuluk yapısı ile tanımlanan bir değişimdir.

Yerel arama sezgiselleri, komşulukların araştırılması ile o anki çözümü geliştirmeye çalışırlar. İlk önemli seçim, komşuluk yapısıdır. Yönetilebilir bir büyüklüğe sahip verilen bir komşuluk yapısı, belli bir dayanıklılığa sahiptir. Güçlü bir komşuluk yapısı, büyük ölçüde başlangıç çözümünün kalitesinden bağımsız yerel eniyi üretirken, zayıf bir komşuluk yapısı, yüksek ölçüde başlangıç çözümüyle ilişkili bir yerel eniyi üretir. Diğer bir seçim, komşuluk(lar) içindeki çözümlerin nasıl araştırılacağıdır. Bunun için aşağıda birkaç yol tanımlanmıştır (Silva, 2003):

Deterministik İteratif Gelişim (Deterministic Iterative Improvement):

Temel yerel arama stratejisi ya da deterministik iteratif gelişim, verilen bir komşuluk veya bir başlangıç çözümü alır. Her iterasyonda bir komşuluk üretilir ve eğer daha iyi ise, o anki çözümle yer değiştirilir. Arama, daha iyi bir çözüm bulunmadığında biter.

İlk ve En İyi İteratif Gelişim (First and Best Iterative Improvement):

Sadece tek bir komşuluğun araştırılması genellikle zayıf bir yerel eniyiye yol açar. Komşuluk çözümlerinin bir alt kümesiyle veya komşuluğun büyüklüğüne bağlı

olanların tümünün üretilmesiyle bir arama yapılabilir. İlk komşuluk yapısı daha iyiye, o anki çözüm kabul edilir. Bu ilk iteratif gelişim algoritmasını oluşturur. Tüm komşulukların en iyisi seçilirse, bu yaklaşıma, en iyi iteratif gelişim adı verilir. Eğer en iyi gelişim hareketi alınır, bir eniyi çözüme, ilk gelişim hareketine göre daha az adımda ulaşılır. Diğer taraftan, hangi hareketin en iyi olduğunun belirlenmesi, ilk hareketin seçilmesine göre daha fazla gayret gerektirir.

İteratif gelişim algoritmaları, en büyükleme problemleri için, tırmanma (hill-climbing) yöntemi olarak, en küçükleme problemleri için, iniş (descent) yöntemi olarak tanımlanır. İteratif gelişim algoritması Şekil 1.3' de gösterilmiştir (der Besten, 2000; Silva, 2003).

Adım1. $s \leftarrow \text{BaşlangıçÇözümüOluştur}()$
Adım2. O anki çözüm s ' nin komşuluğunu araştır ve s' aday çözümü oluştur.
Adım3. **If** $f(s') < f(s)$ **then** $s \leftarrow s'$
Adım4. Eğer durma kriteri sağlanıyorsa, dur; aksi halde Adım2' ye git.

Şekil 1.3 İteratif Gelişim Algoritması¹ (Silva, 2003)

Son 20 yılda, bir arama alanının etkin şekilde araştırmayı amaçlayan ve temel sezgisel yöntemleri birleştirmeye çalışan yeni bir yaklaşık algoritma oluşmuştur. Bu yöntemler, metasezgisel olarak adlandırılır (Blum ve Roli, 2003).

¹ Adım 2' de deterministik gelişim, sadece tek bir komşuluk oluştururken, ilk ve en iyi gelişim komşulukların bir kümesini araştırır.

2. METASEZGİSEL YAKLAŞIMLAR

Bir metasezgisel algoritma, arama alanı araştırılması (exploration) ve devamlılığı (exploitation) için kullanılan kavramları birleştirerek ikincil bir sezgisel algoritmaya rehberlik eden iteratif bir süreç olarak tanımlanabilir. Bir metasezgisel, her iterasyonda, tek bir çözümü veya biriktirilmiş çözümleri işleyebilir.

Metasezgisellerin özellikleri şöyle özetlenebilir:

- Metasezgiseller, arama sürecine yol gösteren bir stratejidir.
- Amaç, eniyi (veya yakın) çözümü bulmak için, arama alanını etkin şekilde araştırmaktır.
- Metasezgisel algoritmaları oluşturan teknikler, basit yerel arama süreçlerinden kompleks öğrenme süreçlerine kadar değişebilir.
- Metasezgisel algoritmalar, yaklaşıktır ve genellikle deterministik değildir.
- Metasezgiseller, arama alanındaki tuzak bölgelere gitmeyi önleyecek mekanizmalarla birleştirilebilirler.
- Metasezgisellerin temel düşüncesi, soyut düzeyli tanımlamalara izin verir.
- Metasezgiseller, probleme-özel (problem-specific) değildir.
- Metasezgiseller, yüksek düzeyli stratejilerle kontrol edilen sezgiseller gibi alana-özel bilginin kullanımını sağlarlar.
- Son zamanlarda, daha ileri metasezgiseller, aramaya rehberlik edecek arama deneyimlerini kullanmaktadırlar.

Metasezgisellerin performansını güçlendirmek için en önemli faktörlerden biri yoğunlaştırma (intensification) ve çeşitlendirme (diversification) arasındaki karmaşık takastır. Yoğunlaştırma, o ana kadar yapılan araştırmada bulunan en iyi çözümün iyileştirilmesini sağlar. Bu amaçla iyi çözümün komşuluklarında araştırma yapılmasına olanak sağlar. Çeşitlendirme ise, araştırmayan bölgelere gidilmesini sağlayarak yeni yapıda çözümler üretilmesini sağlar (Blum ve Roli, 2003).

Bu sınıftaki algoritmalar arasında yer alanlar şunlardır:

- Tavlama benzetimi (Simulated Annealing-SA)
- Yasak arama (Tabu Search-TS)
- Keşfedici yerel arama yöntemleri (Explorative Local Search Methods):
 - Hırslı rasgele adaptif arama yordamı (Greedy Randomized Adaptive Search Procedures-GRASP)
 - Değişken komşuluk arama (Variable Neighborhood Search-VNS)
 - Rehber yerel arama (Guided Local Search-GLS)
 - Ardaşık yerel arama (Iterated Local Search-ILS)
- Evrimsel hesaplama (Evolutionary Computation-EC):
 - Genetik algoritmalar (Genetic Algorithms-GA)
 - Evrimsel programlama (Evolutionary Programming-EP)
 - Evrimsel stratejiler (Evolutionary Strategies-ES)
- Dağıtarak arama ve Yeniden yol ilişkilendirme (Scatter Search-SS ve Path Relinking-PR)
- Dağılım tahmini algoritmaları (Estimation of Distribution Algorithms-EDA)
- Diferansiyel gelişim (Differential Evolution-DE)
- Memetik algoritmalar (Memetic Algorithms-MA)
- Küme zekası algoritmaları (Swarm Intelligence Algorithms):
 - Karınca kolonileri eniyileme (Ant Colony Optimization-ACO)
 - Partikül küme eniyileme (Particle Swarm Optimization-PSO)
- Yapay bağışıklık sistemleri (Artificial Immune Systems-AIS)

2.1 Metasezgisellerin Sınıflandırılması

Metasezgiselleri sınıflandıran ve tanımlayan çok sayıda biçim vardır. Aralarında ayırım sağlayan özelliklerine göre, birkaç sınıflandırma mümkündür. En önemli sınıflandırma özellikleri aşağıdaki gibi özetlenebilir (Blum ve Roli, 2003):

- Doğadan-esinlenen ve doğadan-esinlenilmeyen

Doğadan-esinlenen algoritmalar: Genetik algoritmalar ve karınca algoritmaları,

Doğadan-esinlenilmeyen algoritmalar: Yasak arama ve ardaşık yerel arama.

- Tek nokta (single point) ve nüfus-tabanlı arama

Tek çözüm algoritmaları, yörünge yöntemleri (trajectory methods) olarak adlandırılır ve tavlama benzetimi, yasak arama, ardaşık yerel arama ve değişken komşuluk arama gibi yerel arama tabanlı metasezgisellerdir. Arama süreci boyunca arama alanında bir yörünge tanımlama özelliğini paylaşırlar,

Nüfus-tabanlı metasezgiseller, arama alanındaki nokta kümesinin gelişimini tanımlayan bir arama sürecini takip eder. Evrimsel hesaplama ve karınca algoritmaları bu tip metasezgisellerdir.

- Dinamik ve statik amaç fonksiyonu

Metasezgiseller amaç fonksiyonu kullanım biçimine göre sınıflandırılabilir. Bazı algoritmalar amaç fonksiyonunu problem gösteriminde verildiği şekilde kullanırken, bazıları ise (rehber yerel arama gibi) amaç fonksiyonunu arama süresince değiştirirler.

- Tek ve çok komşuluk yapıları

Birçok metasezgisel algoritma tek komşuluk yapısı ile çalışırken, diğer metasezgiseller (değişken komşuluk arama gibi) bir komşuluk yapı kümesi kullanırlar.

- Hafıza kullanan ve hafıza kullanmayan yöntemler

Metasezgiselleri sınıflandırmak için önemli bir özellik, arama hafızasının kullanımı açısındandır. Hafıza kullanımı durumunda birkaç çeşit hafıza oluşturma biçimi mevcuttur. Genellikle kısa ve uzun dönem hafıza şeklinde kullanılmaktadır.

Bu çalışmada tek nokta ve nüfus-tabanlı arama sınıflandırması dikkate alınacaktır.

2.2 Yörünge Yöntemleri

Yörünge yöntemleri terimi, bu yöntemlerle yapılan arama süreci, arama alanındaki bir yörünge ile tanımlandığı için kullanılır. Takip eden çözüm, o anki çözümün komşuluğuna ait olabilir yada olmayabilir. Algoritma başlangıç durumundan (başlangıç çözümü) başlar ve alanda bir yörünge tanımlar (Blum ve Roli, 2003).

2.2.1 Tavlama Benzetimi

En eski metasezgisellerden olan tavlama benzetimi, yerel eniyiden kurtulmak için, belli bir stratejiye sahiptir. Algoritma, istatistiksel mekanik için kullanılan Metropolis algoritmasından esinlenilmiş ve ilk olarak birleşik eniyileme problemleri için Kirkpatrick ve diğ. (1983) ve Cerny (1985) tarafından sunulmuştur. Temel düşünce, yerel eniyiden kurtulmak için, o anki çözümden daha iyi elde edilen çözümlere hareketi (yokuş hareketleri (uphill moves)) sağlamaktır. Böyle bir hareketi gerçekleştirme olasılığı, arama boyunca azalır. Şekil 2.1' de tavlama benzetimi algoritmasının yapısı özetlenmiştir.

Algoritma, bir başlangıç çözümü (rasgele yada sezgisel yapıda) oluşturularak ve T sıcaklık parametresi belirlenerek başlar. Daha sonra, her iterasyonda $s' \in N(s)$ çözümü rasgele denenir ve $f(s)$, $f(s')$ ve T değerlerine göre yeni çözüm tanımlı bir kabul olasılığına göre kabul edilir. T sıcaklığı, arama süreci boyunca azalır. Böylece, aramanın başında kabul edilen yokuş hareketinin olasılığı yüksektir ve yavaş yavaş azalır (Ogbu ve Smith, 1990; Blum ve Roli, 2003).

```
s ← BaşlangıçÇözümüOluştur()
T ← T0
While sıcaklık koşullarına ulaşılmamış do
    s' ← RasgeleSeç(N(s))
    If (f(s') < f(s))
        then s ← s'
        else p(T, s', s) olasılığı ile s' yeni çözüm olarak kabul edilir
    EndIf
    Güncelle(T)
EndWhile
```

Şekil 2.1 Tavlama Benzetimi Algoritması (Blum ve Roli, 2003)

2.2.2 Yasak Arama

Yasak arama, geçmişteki aramaları kullanır ve hem yerel eniyiden kurtulur, hem de bir araştırma stratejisini uygular. Algoritma ilk olarak, Glover (1986) tarafından formüle edilmiştir. Şekil 2.2' de yasak arama algoritması özetlenmiştir. Basit yasak arama algoritması, temel bir bileşen olarak bir en iyi geliştirici yerel arama uygular ve yerel eniyiden kurtulmak ve tur sağlamak için kısa dönemli bir hafıza kullanır. Kısa dönemli bu hafıza, daha önce gidilen çözümlerin izini tutan ve tekrar bu çözümlere hareketi izin vermeyen bir yasaklar listesi (tabu list) ile gerçekleştirilir. O anki çözümün yasaklar listesine ait olmayan komşuluğu, çözümleri sınırlandırır. Bu kümeye izin verilenler kümesi (allowed set) adı verilir. Her iterasyonda izin verilenler kümesindeki en iyi çözüm, yeni çözüm olarak kabul edilir. Daha sonra, bu çözüm yasaklar listesine eklenir ve yasaklar listesindeki çözümlerden biri (genellikle FIFO (First In First Out) sırasına göre) listeden çıkarılır. İterasyon sayısı, yasaklar listesinin büyüklüğü (tabu list size) ile belirlenir. Bu büyüklük, belli bir uzunlukla sınırlandırılır. Algoritma bitiş koşullarına ulaşıldığında durur. Ayrıca, izin verilenler kümesi boşsa, yani, $N(s)$ ' deki tüm çözümler yasaklar listesinde yer alıyorsa da bitiş gerçekleşebilir (Gupta ve diğ., 1999; Blum ve Roli, 2003).

```
s ← BaşlangıçÇözümüOluştur()
Yasaklar listesi ←  $\phi$ 
While bitiş koşullarına ulaşılmamış do
    s' ← SeçEnİyi( $N(s)$  / Yasaklar listesi)
    Güncelle( Yasaklar listesi )
EndWhile
```

Şekil 2.2 Yasak Arama Algoritması (Blum ve Roli, 2003)

2.2.3 Keşfedici Yerel Arama Yöntemleri

Son zamanlarda önerilen bu yöntemler, hırslı rasgele adaptif arama yordamı, değişken komşuluk arama, rehber yerel arama ve ardışık yerel arama olarak tanımlanır (Blum ve Roli, 2003).

2.2.3.1 Hırslı Rasgele Adaptif Arama Yordamı

Hırslı rasgele adaptif arama yordamı, geliştirici sezgiseller ile yerel arama yöntemini birleştiren basit bir metasezgiseldir. Şekil 2.3' de hırslı rastgele adaptif arama

yordamı algoritmasının yapısı görülmektedir. GRASP, iki aşamalı iteratif bir süreçtir: çözüm yapıcı ve çözüm güçlendirici. Arama süreci biter bitmez en iyi çözüme dönlür.

Çözüm yapıcı mekanizma (Şekil 2.4) iki bileşenden oluşur: dinamik geliştirici bir sezgisel ve rasgelelik (randomizasyon). Bir s çözümü, elemanların (çözüm bileşenleri) bir kümesinin alt kümesinden oluştuğu varsayılarak, çözüm, her bir zamanda yeni bir elemanın eklenmesiyle adım adım oluşturulur. Bir sonraki elemanın seçimi, aday listesinden rasgele seçilerek yapılır. Elemanlar sezgisel bir kritere göre sıralanır. Bu sıralama amaç fonksiyonuna göre bir sonuç değer alır. Sınırlandırılmış aday listesi (Restricted Candidate List-RCL) adı verilen aday listesi, en iyi α elemanından oluşur. Sezgisel değerler, her adımda güncellenir, böylece, elemanların sonuç değerleri yapıcı süreç boyunca değişir.

Algoritmanın ikinci aşaması, iteratif gelişme gibi basit yerel arama veya SA, TS gibi daha ileri tekniklerdir (Blum ve Roli, 2003).

```
While bitiş koşullarına ulaşılmamış do
     $s \leftarrow \text{HırslıRasgeleÇözümüOluştur}()$  (Şekil 2.4)
     $\text{YerelAramaUygula}(s)$ 
     $\text{EnİyiÇözümüHafızayaAl}()$ 
EndWhile
```

Şekil 2.3 Hırslı Rasgele Adaptif Arama Yordamı (Blum ve Roli, 2003)

```
 $s \leftarrow \emptyset$       {  $s$ , bu durumda kısmi çözümü gösterir }
 $\alpha \leftarrow \text{AdayListeUzunluğunuBelirle}()$ 
While çözüm tamam değil do
     $RCL_{\alpha} \leftarrow \text{SınırlandırılmışAdayListesiniOluştur}(s)$ 
     $x \leftarrow \text{RasgeleBirElemanSeç}(RCL_{\alpha})$ 
     $s \leftarrow s \cup \{x\}$ 
     $\text{HırsFonksiyonunuGüncelle}(s)$ 
EndWhile
```

Şekil 2.4 Hırslı Rasgele Çözüm Yapısı (Blum ve Roli, 2003)

2.2.3.2 Değişken Komşuluk Arama

Mladenović (1995) tarafından önerilen değişken komşuluk arama, sistematik olarak değişen bir komşuluk yapısına dayanır. Başlangıç aşamasında, bir komşuluk yapı kümesi N_k ($k = 1, \dots, k_{\max}$) tanımlanır (birçok yerel arama sezgiseli sadece tek bir komşuluk yapısı kullanır; yani, $k_{\max} = 1$). Bu komşuluklar, keyfi olarak seçilebilir. Ancak, daha çok artış sırasına göre komşulukların $|N_1| < |N_2| < \dots < |N_{k_{\max}}|$ sıralaması belirlenir. Başlangıç çözümü oluşturulduktan sonra, komşuluk indekslleme başlatılır ve bir bitiş koşulu gerçekleşene dek algoritma sürdürülür. Şekil 2.5’ de değişken komşuluk arama algoritmasının yapısı görülmektedir (Hansen ve Mladenović, 2003; Blum ve Roli, 2003).

```
Bir komşuluk kümesi  $N_k$  seç.  $k = 1, \dots, k_{\max}$ 
 $s \leftarrow$  BaşlangıçÇözümüOluştur()
While bitiş koşullarına ulaşılmamış do
     $k \leftarrow 1$ 
    While  $k < k_{\max}$  do
         $s' \leftarrow$  RasgeleSeç  $\{N_k(s)\}$ 
         $s'' \leftarrow$  YerelArama( $s'$ )
        If ( $f(s'') < f(s)$ )
            then
                 $s \leftarrow s''$ ;  $k \leftarrow 1$ 
            else
                 $k \leftarrow k + 1$ 
        EndIf
    EndWhile
EndWhile
```

Şekil 2.5 Değişken Komşuluk Arama Algoritması (Blum ve Roli, 2003)

2.2.3.3 Rehber Yerel Arama

Aramaya yardımcı olacak farklı bir yaklaşım ise, amaç fonksiyonunun dinamik olarak değişimidir. Bu yaklaşımı kullanan yöntemler arasında en genel yöntem rehber yerel aramadır. GLS’ nin ana prensibi, arama çevresini değiştirerek yerel eniyiden uzaklaşmaktır. GLS’ de, çözüm kümesi ve komşuluk yapısı sabit

tutulurken, amaç fonksiyonu f , daha az istenilen o anki yerel eniyiyi oluşturmak amacıyla dinamik olarak değiştirilir. Şekil 2.6' da rehber yerel arama algoritmasının yapısı görülmektedir (Voudouris, 1997; Blum ve Roli, 2003).

```

 $k \leftarrow 0$  ;  $s \leftarrow \text{BaşlangıçÇözümüOluştur}()$ ;

For  $i \leftarrow 1$  to  $m$  do
     $p_i \leftarrow 0$ 
EndFor
While bitiş koşullarına ulaşılmamış do
    Denklem(2.2)' ye göre yeni fonksiyonu ( $f'$ ) belirle;
     $s' \leftarrow \text{YerelArama}(s, f')$ ;
    For  $i \leftarrow 1$  to  $m$  do
        Denklem(2.3)' e göre maksimum faydayı belirle;
    EndFor
    For  $Util(s, i)$  max.fayda ile tüm özellikler do
         $p_i \leftarrow p_i + 1$ 
    EndFor
     $k \leftarrow k + 1$ 
EndWhile
Güncelle( $f', p$ )

```

Şekil 2.6 Rehber Yerel Arama Algoritması (Blum ve Roli, 2003)

GLS tarafından kullanılan mekanizma, çözüm özelliklerine dayanır. Bu çözüm özellikleri, çözümler arasında ayırım yapmak için kullanılan nitelik veya karakteristikler olabilir. Bir gösterge fonksiyonu $I_i(s)$, i özelliğinin s çözümünde olup olmadığını gösterir:

$$I_i(s) = \begin{cases} 1 & : i, s \text{ çözümünde ise} \\ 0 & : \text{diğer} \end{cases} \quad (2.1)$$

Amaç fonksiyonu f , m özelliğine bağlı bir terim eklenerek yeni bir amaç fonksiyonu f' elde etmek için değiştirilir:

$$f'(s) = f(s) + \lambda \sum_{i=1}^m p_i \cdot I_i(s) \quad (2.2)$$

Burada p_i , ceza parametresi, λ , düzenleme parametresi olarak adlandırılır. Ceza parametresi özelliklerin önemine göre ağırlıklandırılır. Düzenleme parametresi, orijinal amaç fonksiyonuna göre özelliklerin uygunluğunu denetler.

Algoritma başlangıç çözümü ile başlar ve yerel minimuma ulaşana dek bir yerel arama yöntemi kullanır. Daha sonra, $\mathbf{p} = (p_1, \dots, p_m)$ cezaları, cezalar arttırılarak güncellenir ve yerel arama tekrar başlar. Maksimum fayda:

$$Util(s, i) = I_i(s) \cdot \frac{c_i}{1 + p_i} \quad (2.3)$$

Burada c_i , her i özelliğine atanan maliyetlerdir (Blum ve Roli, 2003).

2.2.3.4 Ardaşık Yerel Arama

Ardaşık yerel arama, basit ancak güçlü bir arama yöntemidir. Başlangıç aşamasında, yerel eniyi bulunana dek yerel arama uygular, daha sonra çözümü karıştırır ve tekrar yerel aramaya başlar. Şekil 2.7’ de ardaşık yerel arama algoritmasının özeti görülmektedir (Lourenço ve diğ., 2003; Blum ve Roli, 2003). Prensip olarak, herhangi bir yerel arama mekanizması kullanılabilir. Ancak, çözüm kalitesi ve hesaplama hızı açısından, ardaşık yerel aramanın performansı, seçilen mekanizmanın performansına bağlıdır. Karıştırma stratejisi o anki yerel minimumdan ayrılmaya izin verecek, yeni ve daha iyi yerel minimumu bulmak için yerel arama sağlayacak kadar güçlü olmalıdır. Aynı zamanda, o anki yerel minimumun özelliklerini korumada ise yeterince zayıf olmalıdır (Stützle, 1998b).

```

 $s_0 \leftarrow \text{BaşlangıçÇözümüOluştur}()$ 
 $\hat{s} \leftarrow \text{YerelArama}(s_0)$ 
While bitiş koşullarına ulaşılmamış do
     $s' \leftarrow \text{Karıştır}(\hat{s}, \text{geçmiş})$ 
     $\hat{s}' \leftarrow \text{YerelArama}(s')$ 
     $\hat{s} \leftarrow \text{KabulKriteriniUygula}(\hat{s}, \hat{s}', \text{geçmiş})$ 
EndWhile

```

Şekil 2.7 Ardaşık Yerel Arama Algoritması (Blum ve Roli, 2003)

2.3 Nüfus-Tabanlı Yöntemler

Nüfus-tabanlı metasezgiseller, algoritmanın her aşamasında tek bir çözümle değil, çözüm kümesi ile ilgilenirler. Çözümlerin nüfusu ile ilgilendiği için, nüfus-tabanlı yöntemler, arama alanının araştırılmasında doğal ve gerçek bir yol sağlar (Blum ve Roli, 2003).

2.3.1 Evrimsel Hesaplama

Evrimsel hesaplama algoritmaları, evrimsel süreçlerin hesaplama modelleri olarak tanımlanabilir. Her iterasyonda, gelecek nesil (generation) nüfusuna ait bireyleri oluşturmak için, o anki nüfusun bireylerine çok sayıda operatör uygulanır. Genellikle, evrimsel hesaplama algoritmaları, yeni bireyler oluşturmak için iki yada daha fazla bireyi birleştiren çaprazlama (recombination yada crossover) adı verilen operatörü kullanırlar. Ayrıca, bireylerin uyarlanmasına yol açan değişim (mutation yada modification) operatörünü kullanırlar. Evrimsel algoritmalarındaki yönlendirici güç, uygunluklarına göre bireylerin seçimidir.

Yıllardır çok sayıda evrimsel hesaplama algoritmaları önerilmiştir. Temel olarak, üç kısma ayrılabilir: genetik algoritmalar, evrimsel programlama ve evrimsel stratejiler. Bazen evrimsel hesaplama algoritmaları olarak görülen diğer iki nüfus-tabanlı yöntemler dağıtarak arama/yeniden yol ilişkilendirme ve dağılım tahmini algoritmalarıdır (Blum ve Roli, 2003). Diferansiyel gelişim algoritması da evrimsel hesaplama algoritmaları içinde sınıflandırılabilir (Lampinen, 1999).

Şekil 2.8' de evrimsel hesaplama algoritmalarının temel yapısı görülmektedir. Burada P , birey nüfusunu gösterir (Blum ve Roli, 2003).

```
P ← BaşlangıçÇözümüOluştur()
Değerlendir(P)
While bitiş koşullarına ulaşılmamış do
    P' ← Çaprazlama(P)
    P'' ← Değişim(P')
    Değerlendir(P'')
    P ← Seç(P'' ∪ P)
EndWhile
```

Şekil 2.8 Evrimsel Hesaplama Algoritması (Blum ve Roli, 2003)

2.3.1.1 Genetik Algoritmalar

Genetik algoritmalar, John Holland (1975) tarafından bulunmuş ve onun öğrencileri ve arkadaşları tarafından geliştirilmiştir. Genetik algoritmalar, kromozomlar ile çalışan doğa gibi ikili dizilerin nüfusu ile çalışır. İkili diziler, algoritmanın bulunacağı veya belirleyeceği parametrelerin kodlanmasından oluşturulmaktadır. İkili dizilerin oluşturduğu gruba nüfus adı verilmektedir (t nesli için, $P(t)$). İlk nüfus rasgele oluşturulmaktadır. Nüfusu oluşturan ikili dizilerin sayısı nüfus büyüklüğü olarak tanımlanır ve nüfus büyüklüğü işlem boyunca sabit kalmaktadır. Nüfus büyüklüğü geniş tutulduğu zaman ilk birkaç nesilde sonucu kapsama olasılığı artmakta fakat genetik algoritmanın çalışma hızı azalmaktadır.

Nüfusu oluşturan her dizinin çözüme uygunluğunu belirleyen uygunluk fonksiyonu vardır. Algoritma, sadece nüfusu oluşturan her diziye ait bir uygunluk değerine ihtiyaç duyar. Amaç fonksiyonu ile eniyilenecek problem tanımlanır. Birçok durumda uygunluk fonksiyonu amaç fonksiyonu olabilmektedir.

Nüfusta yeni bireyler oluşturmak için, bazı bireyler genetik operatörler tarafından bazı dönüşümlere maruz bırakılır. İki genetik operatör vardır: çaprazlama ve değişim. Daha sonra oluşturulan yeni bireyler değerlendirilir. Yeni nüfus, ebeveyn nüfusu ve yeni bireylerin nüfusundan en uygun olan bireylerin seçilmesi ile oluşturulur. Şekil 2.9' da genetik algoritmaların yapısı görülmektedir (Gen ve Cheng, 2000).

```
t ← 0
Başla( $P(t)$ )
Değerlendir( $P(t)$ )
While bitiş koşullarına ulaşılmamış do
     $P'(t) \leftarrow \text{Çaprazla}(P(t))$ 
     $P''(t) \leftarrow \text{Değişim}(P'(t))$ 
    Değerlendir( $P''(t)$ )
     $P(t+1) \leftarrow \text{Seç}(P''(t) \cup P(t))$ 
    t ← t + 1
EndWhile
```

Şekil 2.9 Genetik Algoritmalar (Gen ve Cheng, 2000)

Etkin bir genetik algoritma dizayn edildiğinde dikkatle seçilmesi gerekli bileşenler şunlardır (Silva, 2003):

- Bireylerin kodlanması
- Seçim mekanizması
- Genetik operatörler
- Yineleme şeması
- Kısıt yönetme mekanizması

Bireylerin Kodlanması:

Genetik algoritmalarda birey, bir çözüm, bir kısmi çözüm veya bunların bir kümesidir. Genetik algoritmalarda bireylerin gösterimi kromozom olarak adlandırılır. Kromozom temsil ettiği çözümle ilgili bilgi içermelidir. Her kromozom ikili (binary) bir diziden oluşur. Bu dizi içindeki bit adı verilen her sayı çözümün bir karakteristiğini temsil edebilir veya bir dizi bütünüyle bir sayıya işaret edebilir.

Kromozomu ikili düzendeki sayılar dizisiyle ifade etmek çok tercih edilen bir temsil şeklidir ancak bunun yerine tamsayı veya reel sayılar da kullanılabilir. İkili düzenin tercih edilmesinin sebebi basit olması ve bilgisayar tarafından daha kolay ve hızlı bir biçimde işlenebilmesidir.

Seçim Mekanizması:

Seçim işlemi, daha geniş uygunluk değerine sahip dizilerin daha yüksek olasılık ile yeni nesilde geniş sayıda kopyalarını üretebilen işlemdir. Seçme kriterleri uyumluluğu esas alarak birbiriyle uyumlu olan bireyleri seçer. Daha sonra çaprazlama ve mutasyon uygulanacak olan bireylerden daha uyumlu yeni bireylerin ortaya çıkması olasıdır. Bireylerin tamamı uyumluluğa göre seçilebilir veya bir kısmı rasgele seçilerek yeni kuşağa aktarılabilir.

Seçim işlemi için çok sayıda yöntem önerilmiştir (Gen ve Cheng, 2000):

- Rulet tekerleği seçimi
- $(\mu+\lambda)$ seçimi
- Turnuva seçimi
- Sabit durumun yeniden üretimi

- Sıralama ve dengeleme
- Paylaştırma

Genetik Operatörler:

Genetik algoritmalar iki temel işlemden oluşmaktadır: çaprazlama ve değişim.

- **Çaprazlama:** Çaprazlama işlemi, bilgilerin iki dizi arasındaki değişimi ile ilgilidir. Çaprazlama işleminde yeni üretilmiş iki dizi eşleştirme havuzundan seçilerek rasgele seçilmiş çaprazlama pozisyonuna göre ikili dizilerdeki bilgi değişimini gerçekleştirebilmektedir. Bu işlem tercih edilmiş iyi dizilerin arasında daha iyi özellikleri birleştirir.
- **Değişim:** Değişim, seçim ve çaprazlama işlemlerini tamamlayıcı bir işlemdir. Değişim işlemi, basitçe bit durumlarını tersine çevirmektir. Rasgele seçilen bit pozisyonundaki değerin arasına değişikliği sayesinde eniyi noktaya ulaşmayı kuvvetlendirmiş olmaktadır.

Yineleme şeması:

Çaprazlama ve değişim operatörleri uygulanır uygulanmaz, son nesildeki bireylerden hangilerinin yeni nüfusu oluşturacak yeni döllere yer değiştireceğine karar vermek gerekir.

Elitist stratejisi, genetik materyaller bir sonraki nesle aktarılacak şekilde en iyi bireyleri devam ettirirken, non-elitist stratejisi, o anki nüfustaki tüm bireyleri yineler.

Kısıt yönetme mekanizması:

Bireylere çaprazlama ve değişim işlemleri uygulanması, uygun olmayan çözümleri oluşturabilir. Genetik algoritmalar için kısıt yönetme teknikleri üç grupta toplanabilir:

1. Kısıtların bozulmasına izin vermek ancak ceza uygulamak.
2. Uygun olmayan çözümleri düzeltecek özel onarıcı sezgiseller uygulamak.
3. Sadece uygun çözümlerin oluşma olasılığını arttıran veya garanti eden özel birey gösterimleri kullanmak veya uygun çözümlere yönelik probleme özel operatörler kullanmak.

2.3.1.2 Evrimsel Programlama

Evrimsel programlama, Fogel (1962) tarafından zaman serilerinin tahmininde kullanılmak üzere oluşturulmuştur. EP algoritmasında, çaprazlama işlemi kullanılmaz ve değişim işlevine dayanır. Değişim işlevi, bir düzgün olasılık dağılımı kullanılarak uygulanır. Bu işlevde, her iterasyonda her ebeveyn tek döl (offspring) oluşturur. Yani, $\lambda = 2\mu$ olmaktadır (λ : ebeveyn nüfus büyüklüğü, μ : döl nüfus büyüklüğü).

Modern EP, çaprazlamanın olmadığı, böylece sadece değişim ve seçime dayanan evrimsel hesaplama gibi tanımlanır. EP, gerçek-değerli eniyileme problemlerine uygulandığı zaman, normal-dağılmış bir değişim kullanır ve genellikle, aynı zamanda kendi stratejik parametrelerini geliştirir (van den Bergh, 2001).

2.3.1.3 Evrimsel Stratejiler

Rechenberg (1973) tarafından oluşturulan evrimsel stratejiler, genellikle gerçek-değerli eniyileme problemlerine uygulanmaktadır. ES, değişim ve çaprazlama uygularken, aynı zamanda arama uzayını ve stratejik parametre uzayını araştırır.

Ebeveyn ve döl nüfus büyüklüğü genellikle farklıdır ve döl nüfus büyüklüğü en az ebeveyn nüfusu kadar geniştir. Sıklıkla iki farklı şema tipine rastlanır: virgül stratejisi (comma strategy) ve artı stratejisi (plus strategy). (μ, λ) notasyonu ile tanımlanan virgül stratejisinde, μ ebeveyni, λ dölünü oluşturmak için kullanılır. λ dölllerinin ve onların μ ebeveynleri, algoritmanın bir sonraki iterasyonunda ebeveyn nüfusu olacak şekilde seçilecektir. Bu stratejinin anlamı, orijinal ebeveyn nüfusu, korunmadığı için, önceki neslin iyi sonuçlarının kaybedilebilmesidir. Diğer taraftan, bu yaklaşım nüfusun farklılaştırılmasını artırır. $(\mu + \lambda)$ notasyonu ile tanımlanan artı stratejisinde ise, μ ebeveynleri ve λ dölleri bir geniş nüfus içinde tutulur. Seçim, bu birleşik nüfustan, bir sonraki ebeveyn nüfusu olarak, μ ve λ 'nin her ikisinde bulunmuş en iyi çözümleri seçerek yapılır. Bu yöntem, nüfusun en iyi bireylerinin uygunluğu, bir tekdüzelik fonksiyonu olacak şekilde, o ana kadar bulunmuş en iyi çözümleri korur.

Ebeveyn ve döl nüfusu arasındaki ilişki, özellikle $(\mu + \lambda)$ strateji durumunda, seçimi deterministik yapar (van den Bergh, 2001).

2.3.2 Dağıtarak Arama ve Yeniden Yol İlişkilendirme

Dağıtarak arama ve onun genelleştirilmiş biçimi olan yeniden yol ilişkilendirme, öklit (dağıtarak arama) veya komşuluk uzayında (yeniden yol ilişkilendirme) çözümlerin birleştirilmesi için bütünleyici prensipler sağlaması açısından EC algoritmalarından farklıdır. Ayrıca yasak arama yöntemindeki, uyarlanmış hafıza ve birleştirilmiş hafıza-arastırma mekanizmaları gibi bazı fikirleri içerir. Şekil 2.10' da dağıtarak arama ve yeniden yol ilişkilendirme algoritmalarının yapısı gösterilmiştir.

Başlangıç aşaması:

BaşlangıçÇözümüOluştur()

Repeat

Çeşitlendirme()

Güçlendirme()

ReferansKümesiGüncelleme()

Until referans kümesi n kardinalitesine sahip

Dağıtarak arama ve Yeniden yol ilişkilendirme aşaması:

BaşlangıçÇözümüOluştur()

Repeat

AltKümeOluştur()

ÇözümBirleştirme()

Güçlendirme()

ReferansKümesiGüncelleme()

Until bitiş koşullarına ulaşılma

Şekil 2.10 Dağıtarak Arama ve Yeniden Yol İlişkilendirme Algoritmaları (Blum ve Roli, 2003)

Dağıtarak arama ve yeniden yol ilişkilendirme, gözönüne alınan problem için uygun çözümlerle ilişkili seçilmiş bir referans çözüm kümesinden bir çözüm kümesi oluşturan bir arama stratejisidir. Bu arama, o anki referans çözüm kümesinin altküme kombinasyonları ile yapılır. Sonuç çözümler, deneme sonuçları (trial solutions) olarak adlandırılır. Bu deneme sonuçları uygun olmayan çözümler olabilir ve genellikle bunları uygun çözüme dönüştürecek bir onarıcı süreç ile değiştirilirler. Daha sonra, genellikle yerel arama gibi bir güçlendirici mekanizma bu deneme çözüm kümesini güçlendirmek için kullanılır. Bu güçlendirilmiş çözümler, dağıtılmış çözüm (dispersed solution) kümesini oluşturur. Bir sonraki iterasyon için

kullanılacak yeni referans çözüm kümesi, o anki referans çözüm kümesinden ve yeni oluşturulmuş bu dağıtılmış çözüm kümesinden seçilir (Blum ve Roli, 2003).

2.3.3 Dağılım Tahmini Algoritmaları

Son on yılda, birçok araştırmacı EC algoritmalarında kullanılan çaprazlama operatöründeki sorunları çözmeye çalışmaktadır. Bunun için önerilen algoritmalarından biri Mühlenbein ve Paaß (1996) tarafından önerilen dağılım tahmini algoritmasıdır. Olasılık teorisinin teorik altyapısına sahip bu algoritma, arama gelişimleri ile evrimleştirilen nüfuslara dayanmaktadır. EDA, arama uzayında bir dağılımı tahmin etmek için umut verici çözümlerin olasılık modellemesini kullanır ve daha sonra bu model, tahmin edilen dağılıma göre bir sonraki nesli üretmek için kullanılır. Her iterasyondan sonra, bu dağılım yeniden tahmin edilir. Şekil 2.11’ de EDA’ nın yapısı görülmektedir.

```
P ← BaşlangıçNüfusu()
While bitiş koşullarına ulaşılmamış do
    Pseç ← Seç( P )
    p(x) = p(x|Pseç) ← OlasılıkDağılımıTahmini()
    P ← OlasılıkDağılımıÖrnekleme()
EndWhile
```

Şekil 2.11 Dağılım Tahmini Algoritması (Blum ve Roli, 2003)

Bu yöntemin amacı, arama uzayında örnekleme yapıldığında, yüksek olasılıkla yüksek kalitede çözümler oluşturan gerçel değerli bir olasılık vektörü yaratmaktır. Yüksek kalitede çözümler üretmek için bu olasılık vektörünün değerini değiştirme amacı şöyle gerçekleştirilir: birkaç çözüm vektörü, olasılık vektörüne göre üretilir. Daha sonra, olasılık vektörü yüksek kaliteli oluşturulmuş çözüm vektörüne göre değiştirilir. Bu olasılık vektörünün uzunluğu, öğrenme hızı parametresine göre değiştirilir. Daha sonra, olasılık vektörüne bir değişim operatörü uygulanır. Bundan sonra, çevrim tekrarlanır (Blum ve Roli, 2003).

2.3.4 Diferansiyel Gelişim

Diferansiyel gelişim algoritması ilk olarak sürekli uzay fonksiyonlarının (continuous space functions) en küçüklenmesi için önerilmiştir (Storn ve Price, 1997). Literatürde

çeşitli diferansiyel gelişim algoritması vardır. Burada *DE/rand/1/bin* yapısı tanımlanacaktır (Lampinen ve Zelinka, 2000; Lampinen, 1999).

Eniyilenecek f fonksiyonu aşağıdaki biçime sahiptir:

$$f(X) : R^D \rightarrow R \quad (2.4)$$

Eniyileme amacı, amaç fonksiyonu değerini en küçüklemeektir. Bu, aşağıdaki amaç fonksiyonu parametre değerlerinin eniyilenmesiyle yapılır:

$$X = (x_1, \dots, x_D) \quad X \in R^D \quad (2.5)$$

Burada X , D adet amaç fonksiyonu parametresinin oluşturduğu bir vektörü gösterir. Genellikle, amaç fonksiyonu parametreleri sırasıyla, $x^{(L)}$ ve $x^{(U)}$, alt ve üst sınır kısıtlarına sahiptir:

$$x_j^{(L)} \leq x_j \leq x_j^{(U)} \quad j = 1, \dots, D \quad (2.6)$$

Tüm evrimsel eniyileme algoritmaları gibi, DE algoritması da, tek bir çözümle değil aday çözümlerin nüfusu P_G üzerinde çalışır. Bu aday çözümler, nüfusun bireyleridir. DE, NP adet gerçek değerli vektör (X_{iG}) içeren sabit büyüklükteki bir P_G nüfusundan oluşur. Burada, i , nüfus indeksi ve G , nüfusa ait nesildir (generation).

$$P_G = X_{iG} \quad i = 1, \dots, NP, \quad G = 1, \dots, G_{\max} \quad (2.7)$$

Ek olarak, her vektör, D adet gerçek parametre içerir (bireylerin kromozomları):

$$X_{iG} = x_{j,i,G} \quad i = 1, \dots, NP, \quad j = 1, \dots, D \quad (2.8)$$

$P_{G=0}$ başlangıç çözümünü oluşturmanın doğal bir yolu, verilen kısıt sınırları içinden rasgele bir değer seçilmesidir:

$$P_0 = x_{j,i,0} = rand_j[0,1] \cdot (x_j^{(U)} - x_j^{(L)}) + x_j^{(L)} \quad i = 1, \dots, NP, \quad j = 1, \dots, D \quad (2.9)$$

burada, $rand_j[0,1]$, her j için, $[0.0,1.0]$ aralığındaki düzgün dağılan rasgele değeri gösterir.

Diferansiyel gelişimin kendi-referansiyel nüfusunu yeniden üretim yöntemi, diğer evrimsel algoritmalarından daha farklıdır. Birinci nesilden sonra, o anki P_G nüfusundaki vektörler, rasgele seçilir ve sonraki P_{G+1} nesli için aday vektörler yaratmak için birleştirilir. Aday nüfus veya deneme vektörleri $P'_{G+1} = U_{i,G+1} = u_{j,i,G+1}$ aşağıdaki gibi oluşturulur:

$$u_{j,i,G+1} = \begin{cases} v_{j,i,G+1} = x_{j,r3,G} + F \cdot (x_{j,r1,G} - x_{j,r2,G}) & \text{eger } rand_j[0,1] \leq CR \vee j = k \\ x_{j,i,G} & \text{diger} \end{cases} \quad (2.10)$$

burada,

$$i = 1, \dots, NP, \quad j = 1, \dots, D$$

$k \in \{1, \dots, D\}$; rasgele parametre indeksi, her i için bir kez seçilir,

$r_1, r_2, r_3 \in \{1, \dots, NP\}$; $r_1 \neq r_2 \neq r_3 \neq i$ dışında rasgele seçilir,

$CR \in [0,1]$, $F \in (0,1+]$.

F ve CR , diferansiyel gelişimin kontrol parametreleridir. NP gibi, her iki değer, tüm süreç boyunca sabit kalır. F , diferansiyel gelişimi dengeleyen $(0.0, 1.0+]$ aralığında gerçel-değerli bir faktördür. F için üst sınır, deneysel olarak belirlenir.

CR , $[0.0, 1.0]$ aralığında gerçel-değerli bir çaprazlama faktörüdür ve o anki $x_{j,i,G}$ vektörü yerine, değiştirilmiş $v_{j,i,G+1}$ vektörünün rasgele seçilme olasılığını kontrol eder. Genellikle, F ve CR , sürecin hızını ve sağlamlığını etkiler. Eniyi değerleri, amaç fonksiyonunun karakteristiğine ve NP nüfus büyüklüğüne bağlıdır. Çoğunlukla, F , CR ve NP için uygun değerler, farklı değerler kullanılarak yapılan birkaç testten sonra deneme yanılma ile bulunabilir.

Diferansiyel gelişimde seçim yöntemi, diğer evrimsel algoritmalarından farklıdır. Bir sonraki nesil için nüfus P_{G+1} , o anki P_G nüfusundan aşağıdaki gibi oluşturulur:

$$X_{i,G+1} = \begin{cases} U_{i,G+1} = u_{j,i,G+1} & \text{eger } f(U_{i,G+1}) \leq f(X_{i,G}) \\ X_{i,G} & \text{diger} \end{cases} \quad (2.11)$$

Böylece, geçici nüfusun her bireyi, o anki nüfustaki karşılığı ile kıyaslanır. Amaç fonksiyonunun en küçüklendiği farz edilirse, düşük amaç fonksiyonuna sahip vektör

bir sonraki nesilde olmaya hak kazanır. Böylece, sonraki neslin tüm bireyleri o anki nesildeki karşılıklarından daha iyidir. Diferansiyel gelişimin seçim yönteminde ilginç bir nokta, deneme vektörünün, o anki nüfusun tüm bireyleri ile değil tek bireyi ile karşılaştırılmasıdır.

Diferansiyel gelişim algoritması Şekil 2.12’ de verilmiştir (Becerra ve Coello, 2004).

Bir başlangıç nüfus büyüklüğü (NP) oluşturun.

Repeat

For nüfustaki her birey için **do**

$r_1, r_2, r_3 \in \{1, \dots, NP\}, r_1 \neq r_2 \neq r_3 \neq i$ ile üç adet rasgele tamsayı üret.
 $k \in \{1, \dots, D\}$; rasgele parametre indeksi üret.

For her j parametresi için **do**

Denklem(2.10)’ a göre aday nüfusu oluşturun.

EndFor

Eğer U_i daha iyiye, U_i ’ nin bireyleri, X_i ile yer değiştirir.

EndFor

Until bitiş koşullarına ulaşılma

Şekil 2.12 Diferansiyel Gelişim Algoritması (Becerra ve Coello, 2004)

2.3.5 Memetik Algoritmalar

Mascoto (1989) ve Norman (1999) içinde yerel aramanın önemli bir yer tuttuğu genetik algoritmaları açıklamak için memetik algoritmayı tanımlamışlardır.

Memetik algoritmada, nüfusa eklenmeden önce her bireyin yerel eniyiye tırmanmasını sağlamak için bir yerel eniyileme uygulanır. O anki t nesli için ebeveyn ve döl nüfusu, $P(t)$ ve $C(t)$ olarak tanımlanır. Şekil 2.13’ de memetik algoritmaların yapısı görülmektedir (Cheng ve Gen, 1997; Silva, 2003).

2.3.6 Küme Zekası Algoritmaları

“Küme zekası” ifadesi ilk olarak Beni, Hackwood ve Wang (1988-1992) tarafından hücreli robotik sistemler için kullanılmıştır. Burada basit ajanlar, yerel kurallara göre çevre ile iletişimde bulunurlar. Küme zekası, böcek kolonileri ve diğer hayvan topluluklarının sürü davranışlarından esinlenerek algoritmalar veya dağıtılmış problem çözme araçları tasarlama girişimidir (Bonabeau ve diğ., 1999; Kennedy ve diğ., 2001).

Literatürde, iki tip küme zekası algoritması bulunmaktadır (de Castro, 2002): karınca kolonileri eniyileme ve partikül küme eniyileme. Karınca kolonileri eniyileme algoritmalarında, karıncaların basit yiyecek arama süreçleri, birleşik eniyileme problemlerini çözmek için kullanılır. Partikül küme eniyileme algoritmalarında ise, bilgi işlemede insan topluluklarının yetenek benzetimi yapılır.

```
t ← 0;  
Başla P(t);  
Değerlendir P(t)  
While bitiş koşullarına ulaşılmamış do  
    C(t) oluşturmak için; Düzenle P(t)  
    YerelTırmanma C(t)  
    Değerlendir C(t)  
    P(t+1) ← Seç(P(t) ∪ C(t))  
    t ← t+1  
EndWhile
```

Şekil 2.13 Memetik Algoritmalar (Cheng ve Gen, 1997)

2.3.6.1 Karınca Kolonileri Eniyileme

Karınca kolonileri eniyileme, gezgin satıcı problemi, kuadratik atama problemi gibi zor çözümlü birleşik eniyileme problemlerinin çözümü için çok-ajanlı bir yaklaşım olarak ilk defa Dorigo ve diğ. (1991, 1992) tarafından önerilmiştir (Dorigo ve diğ., 1999). KKE metasezgiselinin temel mekanizması, yapay karınca kolonisi içerisindeki yapay karıncaların, eniyileme problemine iyi sonuçlar elde edebilmek için işbirliği yapmasıdır. İşbirliği için kullanılan yöntem ise yapay karıncaların birbirleri ile haberleşmesidir. Yapay karıncalar, doğal karıncaların yiyecek arama sürecinde kullandıkları ve en kısa yolun bulunmasını sağlayan temel mekanizmayı kullanırken doğal karıncaların sahip olmadığı bazı yeteneklerle de donatılmışlardır. Aslında bu donanım ve süreçteki farklılıklar genel anlamda karınca algoritması sınıfına giren fakat farklı isimlere sahip algoritmaların da ortaya çıkmasını sağlamıştır.

Yöntem, Bölüm 3' de ayrıntılı olarak açıklanacaktır.

2.3.6.2 Partikül Küme Eniyileme

Partikül küme eniyileme, Kennedy ve Eberhart (1995) tarafından önerilen nüfus-tabanlı eniyileme yöntemidir. Evrimsel algoritma ailesine ait algoritmalara benzeyen PSO, hata fonksiyonundan çıkarılan bir eğilim bilgisine ihtiyaç duymayan ve uygulanması kolay stokastik bir algoritmadır (van den Bergh, 2001). PSO, sosyal ve öğrenmeye dayalı davranış çalışmalarında kullanılmak üzere oluşturulmuş, ancak daha çok mühendislik ve bilgisayar biliminde bir problem çözüm tekniği olarak uygulanmaktadır. PSO algoritmalarının iki ana tipi vardır: ikili (binary) ve gerçek-değerli (real-valued) gösterimi dışında, algoritmanın her iki tipi oldukça benzerdir.

Partikül küme yaklaşımı, ikili veya gerçek değerli vektörlerle gösterilen bireylerin nüfusunu kullanır. Bu vektörlere partiküller (particles) adı verilir. s ise, küme büyüklüğüdür. Her partikül, bir eniyileme probleminde potansiyel bir çözümü gösterir. Bu bireylerin, belli bir komşuluk yapısı içinde diğer bireylerle iletişim yeteneğine sahip sosyal bireyler olduğu varsayılır.

Nüfustaki her birey için iki tip bilgi vardır. İlki kendi geçmiş deneyimleri ve ikincisi kendi etraflarındaki bireylerin oluşturdukları bilgi. Bu iki tip bilgi bireysel öğrenme ve kültürel değişim için birleştirilir.

Bireyler, geçmişteki başarılarından ve komşuları olan bireylerin başarılarından etkilene eğilimindedirler. Bireyler arasındaki bu iletişim düzeni, sosyometrik prensipler (sociometric principles) olarak adlandırılır. Bireyler birbirleri ile birkaç şekilde iletişim kurabilirler. En basit biçim, ikili iletişimdir. Burada bireyler en yakınlarındaki iki komşuları ile iletişim kurarlar. Herhangi k sayıda en yakın komşuluk kullanılabilir. Eğer en yakın komşuluk sayısı, nüfustaki birey sayısından küçükse, bu sosyometrik prensip *lbest* olarak adlandırılırken, diğer durumda, bu prensip *gbest* olarak adlandırılır. Kavramsal olarak, *gbest* sosyal iletişimi maksimum yaparak tüm bireyleri birbirine bağlar. *lbest* ise, bireylere yerel komşuluk sağlar.

İkili partikül küme algoritması, nitel ve nicel sosyal eniyileme algoritması olarak yorumlanırken, gerçek-değerli partikül küme algoritması, tamamen sayısal eniyileme algoritması olarak yorumlanır. Gerçek-değerli (sürekli) partikül küme algoritmasında, PSO, R^n 'de eniyiyi arar. Burada, n , arama uzayının boyutudur.

PSO' nun sürekli versiyonu, bireyleri, uzaydaki noktalar yani partiküller olarak farzeder ve zamana göre değişim, noktaların hareketi olarak kabul edilir. Bir

psikolojik sistem, bir bilgi işleme süreci olarak görülür ve arama uzayındaki bir partikülün her koordinatı, bir psikolojik ölçüt ile ilişkilidir. Unutma ve öğrenme, belli bir koordinat değerinde sırasıyla artış veya azalış olarak görülür (de Castro, 2002).

Bir i partikülünün o anki pozisyonu x_i , hızı v_i ve en iyi pozisyonu y_i ile verilsin. Hız, iterasyonlar boyunca arama uzayında partikülü hareket ettirmek için partikülün pozisyon koordinatlarına eklenen sayıların bir vektörüdür (t , zaman indeksi) (van den Bergh, 2001; de Castro, 2002):

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2.12)$$

i partikülünün en iyi pozisyonu, partikülün en yüksek uygunluk değerini oluşturan gittiği en iyi pozisyonudur. f , en küçüklenme amacına sahip amaç fonksiyonunu gösterebilir. Buna göre i partikülünün en iyi pozisyon şöyle güncellenir (Bergh, 2001):

$$y_i(t+1) = \begin{cases} y_i(t) & \text{eger } f(x_i(t+1)) \geq f(y_i(t)) \\ x_i(t+1) & \text{eger } f(x_i(t+1)) < f(y_i(t)) \end{cases} \quad (2.13)$$

gbest modelinde kullanıldığı gibi, o ana kadar herhangi bir partikülün en iyi pozisyonu ise:

$$\hat{y}(t) \in \{y_0(t), y_1(t), \dots, y_s(t) | f(\hat{y}(t))\} = \min\{f(y_0(t)), f(y_1(t)), \dots, f(y_s(t))\} \quad (2.14)$$

Algoritma, iki bağımsız rasgele sıra üretir, $r_1 \sim U(0,1)$ ve $r_2 \sim U(0,1)$. Bu sıralamalar algoritmanın stokastik yapısını etkilemek için kullanılır. r_1 ve r_2 değerleri, $0 < c_1, c_2 \leq 2$ sabitleri ile ölçümlendirilir. Bu sabitler, ivme katsayıları olarak adlandırılır ve bunlar bir partikülün bir iterasyonda alabildiği adımların maksimum ölçüsünü etkilerler.

$v_{i,j}$, i . partikülle ilgili hız vektörünün j . boyutunu gösterdiğinde hız güncellemesi şöyle yapılır:

$$v_{i,j}(t+1) = v_{i,j}(t) + c_1 r_{1,j}(t) [y_{i,j}(t) - x_{i,j}(t)] + c_2 r_{2,j}(t) [\hat{y}_j(t) - x_{i,j}(t)] \quad (2.15)$$

Burada, c_1 , partikülün en iyi pozisyonunun yönünde adım büyüklüğünü ayarlarken, c_2 , global en iyi partikülün yönünde maksimum adım büyüklüğünü ayarlar. $v_{i,j}$

değeri, partikülün arama uzayından ayrılma olasılığını azaltmak için $[-v_{\max}, v_{\max}]$ aralığı ile sınırlandırılır. Arama uzayı, $[-x_{\max}, x_{\max}]$ sınırları ile tanımlanırsa, $v_{\max}$ değeri, $v_{\max} = k \times x_{\max}$ olacak şekilde ayarlanır. Burada, $0.1 \leq k \leq 1.0$ dır. Partikül küme eniyileme algoritmasının yapısı Şekil 2.14' de görülmektedir (de Castro, 2002).

Rasgele partikül nüfusu oluştur.

Repeat

Her partikül için;

Değerlendirme ve Güncelleme: Partikülü değerlendir. Eğer o ana kadarki en iyi partikülden daha iyi ise, bu partikülü en iyi partikül yap.

Komşuluk değerlendirme: Bu partikül tüm komşuluklar arasında en iyi ise, bu partikülü global en iyi partikül yap.

Hız Belirleme: Denklem(2.15)' e göre hızı belirle.

Partikül Güncelleme: Denklem(2.12)' e göre partikül güncelle.

Until durma kriterine ulaşma

Şekil 2.14 Partikül Küme Eniyileme Algoritması (de Castro, 2002)

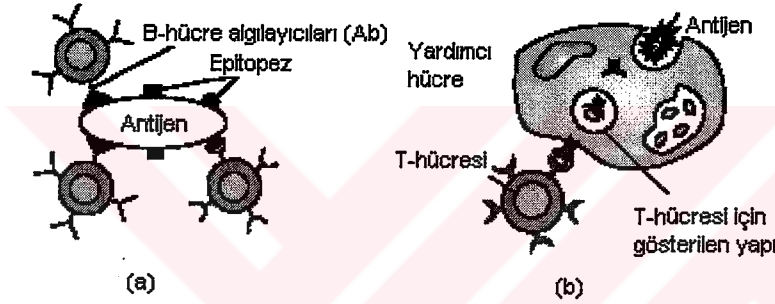
2.3.7 Yapay Bağışıklık Sistemleri

Omurgalı bağışıklık sistemi (Immune System-IS), en karmaşık vücut sistemlerinden biridir ve karmaşıklığı bazen beynin karmaşıklığı ile karşılaştırılır. Biyoloji ve moleküler genetik alanlarındaki gelişmelerle, bağışıklık sistemi davranışlarının anlaşılması hızla artmaktadır. Yapay sinir ağlarının (Artificial Neural Network-ANN) oluşturulmasında sinir sisteminden etkilenildiği gibi; bağışıklık sistemi, yapay bağışıklık sistemlerinin oluşturulmasına yol açmıştır (de Castro ve Timmis, 2002). Böylece, bağışıklık sistemleri, bağışıklık sisteminden esinlenilmiş ve problem çözmede kullanılan uyarlanmış bir sistem olarak tanımlanabilir (de Castro, 2002).

Bağışıklık sistemi:

Bağışıklık sisteminin ana görevi, organizmaları kendi vücutlarındaki kötü çalışan hücrelerin (yani, kanser ve tümör hücreleri) ve hastalığa yol açan elemanların (yani, virüs ve bakteri) araştırılmasını incelemektir. Bağışıklık sistemi tarafından tanınan her eleman antijen (antigen-Ag) olarak adlandırılır. Vücudumuza ait olan ve fonksiyonlara zarar vermeyen hücreler, self (veya self antijenler) olarak

adlandırılırken, hastalıklara yol açan elemanlar, nonself (veya nonself antijenler) olarak adlandırılır. Bağışıklık sistemi, bu iki hücre arasındaki farkı ayırt edebilir ve bu sürece ayırt etme (discrimination) adı verilir. Bağışıklık sisteminin en belli başlı karakteri, ayırt etmeyi sağlayan bağışıklık hücreleri üzerindeki algılayıcı molekülleridir (reseptor molecules). Bağışıklık hücreleri iki ana grupta toplanabilir: B-hücreleri ve T-hücreleri. Bu iki tip hücre oldukça benzerdir, ancak, antijenleri tanıma şekilleri ve fonksiyonel rolleri açısından farklıdırlar. B-hücreleri (antikor (antibodies-Ab)), ortamdaki serbest antijenleri tanıyabilirken, T-hücreleri, antijenlerin diğer yardımcı hücreler tarafından tanıtılmasına ihtiyaç duyar (Şekil2.15) (de Castro, 2002; de Castro ve Timmis, 2002).



Bağışıklık Algoritmaları:

Bağışıklık algoritmaları, teorik bağışıklık çalışma prensibinden ve bağışıklık sisteminde gerçekleşen birkaç süreçten esinlenir. Bağışıklık algoritmaları nüfus-tabanlı ve ağ-tabanlı bağışıklık algoritmaları şeklinde sınıflandırılabilir. Nüfus-tabanlı algoritmalarda, sistem elemanları birbirleri ile bağlı değildir. Yani, direk olarak sadece ortamla etkileşim kurarlar. Birbirleri ile olan etkileşim ise sadece dolaylı olarak gerçekleşir. Burada iki nüfus-tabanlı algoritmadan bahsedilecektir: negatif ve klonal seçim algoritmaları. Ağ-tabanlı bağışıklık algoritmalarında, sistemdeki bazı (veya tüm) elemanlar, birbirlerine bağlıdır. Bu yaklaşımda, çevreyle ve sistemdeki diğer elemanlarla olmak üzere sistemle iki düzeyde etkileşim olur (de Castro ve Timmis, 2002).

Negatif Seçim:

İnsan bağışıklık sistemi, göğüs kafesinin arkasında bulunan timüs adındaki bir organda bulunur ve T-hücrelerinin olgunlaşmasından sorumludur. T-hücreleri oluştuktan sonra, olgunlaşacakları yer olan timüs içine göç ederler. Timüs, nonsel antijenleri timük çevreden uzak tutan bir kan bariyer tarafından korunur. Böylece, timüs içinde bulunan birçok eleman nonsel olanlar yerine self olanlara örnek teşkil ederler. Sonuç olarak, timüs içinde bulunan ve self antijenleri algılama yeteneğindeki algılayıcıları içeren T-hücreleri, negatif seçim adı verilen bir süreç boyunca T-hücrelerinin dağarcığından (repertoire) silinirler. Timüsten ayrılan tüm T-hücreleri, vücutta dolaşırlar ve self olanlara karşı toleranslı olarak adlandırılırlar. Yani, self olanlara karşı tepki vermezler.

Bilgi işleme perspektifi açısından, negatif seçim tanınacak desenlerin (self) tamamlayıcı kümesi (nonsel) hakkında bilgi depolayarak desen tanımayı gerçekleştirir. Literatürde, bilgisayar ve ağ saldırısı bulma, zaman serileri tahmini, şekil denetimi ve kesimlemesi, donanım hatası tolerans etme gibi anormallik bulma problemleri üzerine odaklanan uygulamalarla birlikte negatif seçim algoritmaları önerilmiştir.

İlk olarak, **P** self-kümesi (self-set) olarak adlandırılan korunacak desenlerin kümesi belirlenir. Negatif seçim algoritmasına göre, self-kümesine ait olmayan tüm elemanları, yani nonsel elemanları belirlemekle sorumlu olacak **M** dedektörler (detectors) kümesi oluşturulur (de Castro ve Timmis, 2002). Negatif seçim algoritması Şekil 2.16' da verilmiştir (de Castro ve Timmis, 2002; de Castro, 2002).

Adım1. Başlangıç: Rasgele **C** aday dedektörler kümesi oluştur.

Adım2. Sansürleme (censoring):

While Verilen bir büyüklükteki bir **M** kümesi henüz oluşturulmadığında **do**

Adım2.1. Benzerlik değerlendirme (affinity evaluation): **P** kümesindeki her self ve **C** kümesindeki bir aday dedektör arasında benzerliği belirle.

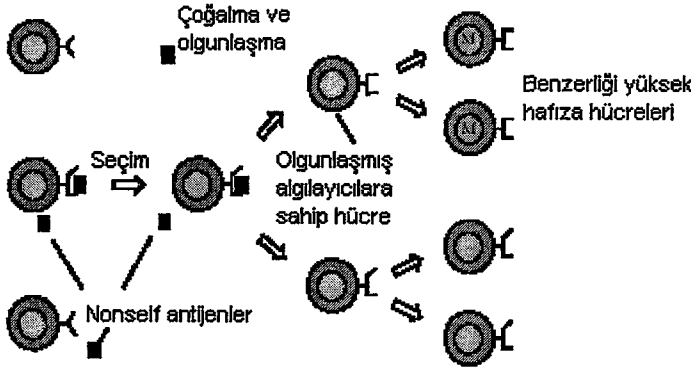
Adım2.2. Seçim (selection): Eğer aday dedektör selfdeki herhangi bir elemanı tanırsa, bu aday elemanı at. Aksi halde bu aday dedektörü **M** kümesine koy.

Adım3. İzleme (monitoring): **M** kümesi oluşturulduktan sonra, herhangi bir değişim için yeni bir self kümesini (**P***) izle. **M** kümesindeki herhangi bir eleman **P*** kümesindeki bir elemanla eşleşirse, bir nonsel elemanı tanınmıştır.

Şekil 2.16 Negatif Seçim Algoritması (de Castro, 2002)

Klonal Seçim:

Negatif seçimin rolünü tamamlayıcı klonal seçim, bir nonself antijenik desen, bir B-hücre tarafından tanındığında nasıl bir bağışıklık tepkisinin verileceğini açıklamak için kullanılan bir teoridir. Şekil 2.17’ de klonal seçim, çoğalma (expansion, proliferation) ve benzerlik olgunlaştırma (affinity maturation) süreçleri açıklanmıştır.



Şekil 2.17 Klonal Seçim, Çoğalma, Benzerlik Olgunlaştırma ve Hafıza Hücrelerinin Korunması (de Castro ve Timmis, 2002)

Bir B-hücre algılayıcısı, belli bir benzerliğe sahip bir nonself antijeni tanır, çoğalmayı seçer ve yüksek miktarda antikor üretir. Antikorlar, nonself antijenlerinin saldırıları ile mücadele etmek için B-hücre yüzeyinden ayrılan B-hücre algılayıcılarının çözülebilir biçimindedirler. Antikorlar, diğer bağışıklık hücreleri ile kendilerinin nihai eliminasyonlarına yol göstermek için antijenlerle birleşirler. Bağışıklık hücrelerindeki çoğalma, eşeysiz yani mitoz üretilir. Hücreler kendi kendilerine bölünürler. Yeniden üreme süresince, B-hücre yavruları, aşırı bir olgunlaşmaya maruz kalırlar. Böylece, seçici antijen ile yüksek benzerlik gösteren antijenik algılayıcılara sahip B-hücreleri oluşur. Hücrelerin ürettiği antikorlardaki farklılaşmaya ek olarak, yüksek antijenik benzerliğe sahip aktif B-hücreleri, uzun ömürlü hafıza hücreleri olmayı seçerler. Bu hafıza hücreleri, aynı veya benzer bir antijenik desene karşı gelecekteki karşılaşmalarında üstündürler. (de Castro ve Timmis, 2002). Klonal seçim algoritması Şekil 2.18’ de verilmiştir (de Castro ve Timmis, 2002; de Castro, 2002).

Tanımlanacak P desenler kümesi için:

Adım1. *Başlangıç*: Rasgele bağışıklık hücre nüfusu (**M**) oluşturun.

Adım2. **For** P’ deki her antijen **do**:

Adım2.1. *Seçim (selection)*: **M** nüfusundan, P’ deki antijenle yüksek benzerliğe sahip olan elemanları seç.

Adım2.2. *Yenidenüreme (reproduction)* ve *genetik değişim (genetic variation)*: **M** nüfusunun kopyalarını üret: **M** nüfusundaki hücre ne kadar iyi antijeni tanır, o kadar çok kopyası üretilir. Benzerlikleri oranında **M** nüfusun her hücresi farklıdır. Benzerlik ne kadar azsa, olgunlaşma oranı da kadar az olur.

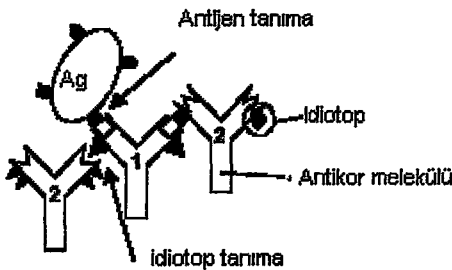
Adım2.3. *Benzerlik değerlendirme (affinity evaluation)*: **M** nüfusun olgunlaşmış her hücresi antijenle olan benzerliğini değerlendir.

Adım3. *Döngü* : Belli bir durma kriteri (minimum desen tanıma veya sınıflama hatası gibi) gerçekleşene dek Adım 2’ ye git.

Şekil 2.18 Klonal Seçim Algoritması (de Castro, 2002)

Bağışıklık Ağı:

Bağışıklık ağ teorisi, bağışıklık sisteminin, dış uyarı olmasa bile dinamik bir davranışa sahip olduğunu ileri sürmektedir. Bağışıklık hücreleri ve moleküllerinin birbirlerini tanıma yeteneğinde olduğunu ve bunun sisteme, yabancı uyarıya bağlı olmayan bir özel-davranış (eigen-behaviour) sağladığını farz eder. Birkaç bağışıklık uzmanı, bu teoriyi çürütmektedir. Bununla birlikte, hesaplama açısından konu ile ilgilidir ve bilgisayar sistemleri için güçlü bir model oluşturmaktadır. Bağışıklık ağ teorisine göre, bağışıklık hücrelerinin yüzeyinde bulunan algılayıcı moleküller, idiotoplar (idiotopes) adı verilen işaretleyicileri gösterirler. Bu idiotoplar, nonspecific antijenleri tanıyan algılayıcıların aynı bölüm çevresinde bulunurlar. Şekil 2.19’ da basit bir bağışıklık ağ teorisi örneği verilmektedir.



Şekil 2.19 Bağışıklık Ağ Teorisi (de Castro ve Timmis, 2002)

Ağ teorisini açıklamak için, B-hücresi üzerindeki bir Ab_1 algılayıcısının (antikor), Ag antijenini tanıdığı farz edilir. Aynı Ab_1 algılayıcısı, başka bir B-hücresinin Ab_2 algılayıcısı üzerindeki i_2 idiotopunu tanıdığı farz edilsin. i_2 idiotopunun Ab_2 'nin bir parçası olduğu gerçeğini takip ederek, Ab_1 , Ag ve Ab_2 'nin her ikisini de tanıma yeteneğindedir. Böylece, Ab_2 'nin Ag 'nin iç görüntüsü olduğu söylenir, tam olarak, i_2 , Ag 'nin iç görüntüsüdür. Bir hücre algılayıcısı üzerindeki idiotopların diğer hücre algılayıcıları tarafından tanınması, bağlı hücre ve molekül kümelerinin artışına yol açar. Bu durumda ağın, bir benzerlik ağı ve sinir sistemindeki sık ağ yapısından farklı olduğu bilinmelidir. Ağ tanıma olaylarının bir sonucu olarak, bir hücre algılayıcısının diğer bir hücre algılayıcısı tarafından tanınması, ağda baskı oluştururken, bir hücre algılayıcısının bir antijeni tanıması ağda hareketlenme ve hücre çoğalması ile sonuçlanır (de Castro ve Timmis, 2002). Şekil 2.20' de bağışıklık ağ algoritması verilmektedir (de Castro ve Timmis, 2002; de Castro, 2002).

Tanımlanacak P desenler kümesi için:

Adım1. *Başlangıç*: Rasgele bir bağışıklık hücre ağı (M) oluşturun.

Adım2. **For** P' deki her antijen **do**:

Adım2.1. *Antijenik tanıma (antigenic recognition)*: M' deki ağ hücreleri ile P' deki antijenlerle eşleştir.

Adım2.2. *Ağ etkileşimi (network interactions)*: M' deki ağ hücreleri ile P' deki ağ hücrelerini eşleştir.

Adım2.3. *Metadinamikler (metadynamics)*: Ağ için yeni hücreler üret ve yararsız olanları elimine et.

Adım2.4. *Uyarılma düzeyi (stimulation level)*: Bir önceki adımın sonuçlarını hesaba katarak, her M' deki ağ hücresinin uyarılma düzeyini değerlendir.

Adım2.5. *Ağ dinamiği (network dynamics)*: Ağ yapısını güncelle ve bireysel hücrelerin uyarılma düzeyine göre parametreleri serbest bırak.

Adım3. *Döngü*: Belli bir durma kriteri gerçekleşene dek Adım 2'ye git.

Şekil 2.20 Bağışıklık Ağ Algoritması (de Castro, 2002)

2.4 Metasezgisellerin Melezleştirilmesi

Metasezgiselleri melezleştirmede kullanılan birkaç biçim vardır (Blum ve Roli, 2003):

- Metasezgiseller arasında bileşen değişimi: En popüler melezleştirme yöntemlerinden biri nüfus-tabanlı yöntemlerinde yörünge yöntemlerinin kullanımıdır. Nüfus-tabanlı yöntemleri, arama uzayında umut verici alanların belirlenmesinde iyi iken, yörünge yöntemleri, arama uzayında umut verici alanların araştırılmasında iyidir. Böylece, nüfus-tabanlı yöntemlerinin gücü ile yörünge yöntemlerinin avantajlarını birleştirilerek oluşturulan metasezgisel melezleştirme oldukça başarılı olmaktadır.
- Ortak arama: Modeller, tüm alt-problemler, çözümler ve diğer arama uzayı özellikleri hakkında bilgi değişimi yapan farklı algoritmalar ile yapılmaktadır. Ortak arama algoritmaları, çeşitli iletişim seviyeleri ile arama algoritmalarının paralel uygulamasından oluşur. Algoritmalar farklı olabilir veya aynı algoritmanın farklı modelleri üzerinde veya farklı parametre kümesi ile çalıştırılan örnekleri olabilir. Bir ortak arama sisteminden oluşan algoritmalar, tümüyle yaklaşık, tam veya yaklaşık ve tam yaklaşımların bir karışımı olabilir.
- Metasezgisellerin ve sistematik yöntemlerin birleştirilmesi: Bu yaklaşım, özellikle gerçek-hayat problemlerine uygulandığında oldukça etkin algoritmalar üretmektedir. Metasezgisellerin ve sistematik yöntemlerin (kısıt programlama veya ağaç arama) birleştirilmesi için üç yaklaşım vardır:

Bir metasezgisel ve bir sistematik yöntemin sırasıyla uygulanması. Örneğin, metasezgisel, daha sonra sistematik arama tarafından sezgisel bilgi olarak kullanılan bazı sonuçlar üretir. Aksine, sistematik arama algoritması, daha sonra metasezgisel tarafından tamamlanan kısmi bir çözüm üretir.

Metasezgiseller, etkin şekilde komşuluk aramak için kısıt programlama ve/veya ağaç aramayı kullanır.

Bir algoritma sınıfından diğeri için kavram ve stratejiler tanımlanır. Örneğin, yasak arama için tanımlanan yasaklar listesi ve emme kriteri kavramları, bir ağaç arama algoritmasında açık düğümlerin listesini oluşturmak için kullanılır.

3. KARINCA KOLONİLERİ ENİYİLEME

KKE metasezgiselinin temel mekanizması, yapay karınca kolonisi içerisindeki yapay karıncaların, eniyileme problemine iyi sonuçlar elde edebilmek için işbirliği yapmasıdır. İşbirliği için kullanılan yöntem ise yapay karıncaların birbirleri ile haberleşmesidir. Ancak bu haberleşme direkt bir yöntem ile değil, izlerin kullanımı ile dolaylı bir yöntem kullanılarak sağlanır. Yapay karıncalar, doğal karıncaların yiyecek arama sürecinde kullandıkları ve en kısa yolun bulunmasını sağlayan temel mekanizmayı kullanırken doğal karıncaların sahip olmadığı bazı yeteneklerle de donatılmışlardır. Aslında bu donanım ve süreçteki farklılıklar genel anlamda karınca algoritması sınıfına giren fakat farklı isimlere sahip algoritmaların da ortaya çıkmasını sağlamıştır (Dorigo ve Di Caro, 1999).

Algoritma olarak ilk örnek, gezgin satıcı problemi olarak bilinen örnek bir uygulama için kullanılan karınca sistemidir. Bu başarısından hareketle, KKE metasezgiseli mevcut uygulamalar ve algoritmik değişiklikler için genel bir yapı olarak görülmektedir. KKE metasezgiselini kullanan algoritmalar, KKE algoritmaları olarak adlandırılır.

KKE algoritmalarının şu anki uygulamaları, iki önemli problem sınıfı olan statik ve dinamik birleşik eniyileme problemleri için yapılmaktadır. Bu problemlerin çözümü için kullanılan KKE algoritmaları birbirine oldukça benzerdir, ancak detayda önemli farklılıklar gösterir (Stützle ve Dorigo, 1999b, 2003).

3.1 KKE Metasezgiselinin Temel Özellikleri

Farklı birleşik eniyileme problemlerinde kullanılan karınca kolonileri eniyilemenin önemli özellikleri şöyle sıralanabilir (Dorigo ve diğ., 1996):

- Çok yönlüdür. Aynı problemin benzer versiyonlarına uygulanabilir.
- Güçlüdür. Kuadratik atama problemi, atölye çizelgeleme problemi (Job Shop Scheduling-JSS) gibi birleşik eniyileme problemlerine küçük değişimlerle uygulanabilir.

- Nüfus-tabanlı yaklaşım. Arama mekanizması olarak pozitif geri dönüşümü olan araştırma sağlar. Ayrıca, sistemi paralel uygulamalara uyarlar.
- Dağıtık hesaplama. Daha kolay yakınsamadan uzaklaşılmasını sağlar.
- Geliştirici hırslı sezgisel. Karıncaların erken aşamalarda uygun çözümler bulmaları sağlanır.

Aynı zamanda, sezgisel bilginin kullanımı, karıncalara daha umut verici çözüme doğru rehberlik edebilir. Daha da önemlisi, karınca araştırmaları, algoritmaların ileriki iterasyonlarında çözüm yapısını öğrenmeyi destekleyecek şekilde kullanılabilir.

3.2 Problem Gösterimi

(S, f, Ω) en küçüklenmesi problemini gözönüne alalım. Burada S , aday çözümler kümesi, f , her $s \in S$ aday çözümü için $f(s, t)$ amaç fonksiyonu değeri atayan amaç fonksiyonu ve Ω , kısıtlar kümesidir.

Amaç, global eniyi çözümü $s_{opt} \in S$ bulmaktır. Yani, Ω kısıtlar kümesini sağlayan minimum çözümü bulmaktır. Karıncalar tarafından araştırılacak (S, f, Ω) birleşik eniyileme problemi aşağıdaki gibi gösterilebilir:

- Bileşenlerin $\mathcal{C} = \{c_1, c_2, \dots, c_{N_c}\}$ bir sonlu kümesi verilir.

Problemin durumları, $\mathcal{C}'$ nin elemanları üzerinde $\chi = \langle c_{i_1}, c_{j_2}, \dots, c_{k_s}, \dots \rangle$ sıralamalarının terimleri ile tanımlanır. Tüm mümkün sıralamaların kümesi, X ile gösterilir. Bir χ sıralamasının uzunluğu, yani, sıralamadaki bileşenlerin sayısı, $|\chi|$ ile gösterilir.

- Sonlu Ω kısıtlar kümesi, $\tilde{X}$ uygun durumlar kümesi ile tanımlanır, $\tilde{X} \subseteq X$.
- Uygun çözümlerin bir S^* kümesi verilir, $S^* \subseteq \tilde{X}$ ve $S^* \subseteq S$.
- $f(s, t)$, her aday çözüm $s \in S$ için atanır.
- Bazı durumlarda, bir amaç yada amacın oranı, $J(\chi_{i,t})$, çözümlerden çok aşamalarla oluşturulabilir. Eğer, bir χ_i durumu için, χ_j çözüm bileşenlerine eklenerek oluşturulabiliyorsa, $J(\chi_{i,t}) \leq J(\chi_j, t)$ dir.

Bu gösterim verildiğinde, yapay karıncalar, yapının grafiği $G(\mathcal{C}, \mathcal{L})$ üzerinde hareket ederek çözümleri oluştururlar. $\mathcal{L}$ kümesi, $\mathcal{C}$ bileşenlerini birleştirir. Yani, $\mathcal{L}$ elemanları ilişkiler olarak adlandırılır. Ω problem kısıtları, yapay karıncalar tarafından takip edilen yön ile sağlanır (Stützle ve Dorigo, 2003).

3.3 Karıncaların Davranışları (Biyolojik Esaslar)

Biyolojik tabanlı diğer sezgisel yöntemlere (genetik algoritmalar, evrimsel stratejiler, yapay sinir ağları, yapay bağışıklık sistemleri gibi) benzer şekilde, karınca algoritmaları, gerçek karınca kolonilerinden esinlenilerek oluşturulmuştur. Karıncalar sosyal böcek türlerindendir. Başka bir deyişle koloniler halinde yaşarlar ve bireylerin davranışlarında öncelik koloninin hayatta kalmasına yöneliktir. Sosyal böceklerin davranış biçimleri bir çok bilim adamının dikkatini çekmiştir. Özellikle koloni içerisindeki bireylerin basit yapılarına karşın, koloninin üst düzeyde yapısal özelliklere sahip olması ilgi çekici olmuştur.

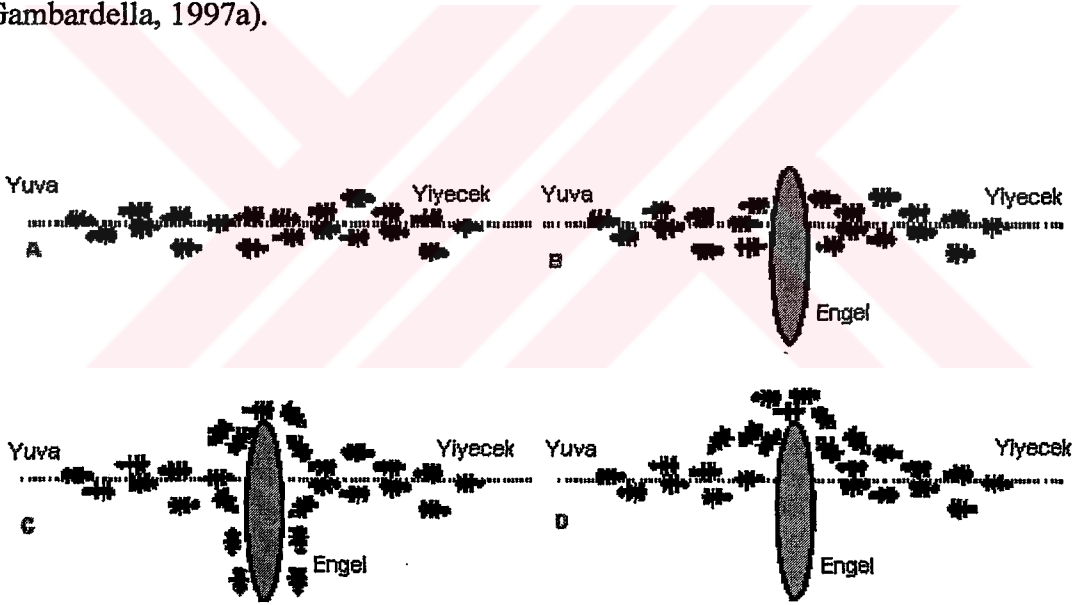
Karınca kolonilerinin önemli ve ilgi çekici olan davranış biçimi yiyecek arama sürecinde ortaya çıkmaktadır. Karıncalar yuvaları ile yiyecek arasındaki en kısa yolu görsel işaretleri kullanmadan bulabilmektedirler. Karıncalar yiyecek kaynağından yuvalarına gidip gelirken kullandıkları yol üzerine pheromone adlı bir madde bırakmaktadırlar. Karıncalar pheromone maddesinin kokusunu alabilirler ve ayırım noktalarında yollarını seçerken kokunun, başka bir deyişle iz miktarının yoğun olduğu tarafı daha yüksek bir olasılıkla seçme eğilimi gösterirler. Pheromone izi sayesinde karıncalar yiyecek kaynağına ve yuvalarına ulaşabilirler veya aynı yuvada yaşayan bir karıncanın bulduğu başka bir yiyecek kaynağına ulaşabilirler.

Yapılan deneylerde, karınca kolonisinin pheromone izini takip etme davranışı neticesinde en kısa yolun bulunabildiği görülmüştür. Yiyecek kaynağı ile yuva arasında bir çok yolun bulunduğu durumlarda karıncaların bireysel olarak bıraktığı izler, koloni tarafından en kısa yolu bulmak için kullanılabilir (Dorigo ve diğ., 1999).

Görme yetenekleri olmayan karıncalar yiyecek aramaya çıktıklarında öncelikle yuvalarının etrafında rasgele arama yapmaktadırlar. Karıncalardan biri yiyecek bulduğunda önce miktar ve kalite açısından yiyeceğin faydasını ölçer ve bir kısmını

yuvaya taşır. Karınca yiyeceğin faydasına bağlı oranda, yuvaya dönüşünde kullandığı yola pheromone izi bırakır. Bu iz diğer karıncalara aynı yiyeceğe ulaşmada rehberlik eder. Yiyeceğe ulaşan karınca miktarı arttıkça dönüş yoluna bırakılan iz miktarı da artar. Yuvaya yakın olan yiyeceklere daha çok karınca ulaşabileceğinden bu yiyecek kaynaklarının yolları üzerindeki iz miktarı da fazla olur.

Karıncalar Şekil 3.1' de gösterildiği gibi daha önce kullandıkları bir yol üzerinde yeni bir engel ile karşılaştıklarında engelin etrafından dolaşmayı denerler. Engel ile ilk karşılaşan karıncalar eşit olasılıkla engelin sağındaki veya solundaki yolu tercih eder. Ancak kısa olan yolu tercih eden karıncalar engeli daha çabuk geçerek daha önce kullandığı yola ulaşabileceğinden kısa yol üzerinde uzun yola göre daha fazla iz oluşur ve engelle yeni karşılaşan karıncalar da izler sayesinde yapılan bu olumlu geri besleme ile kısa yolu daha fazla olasılıkla tercih etmeye başlar. Sonuç olarak karıncalar yiyecek arama işlerinde eniyileme sağlamaktadırlar (Dorigo ve Gambardella, 1997a).

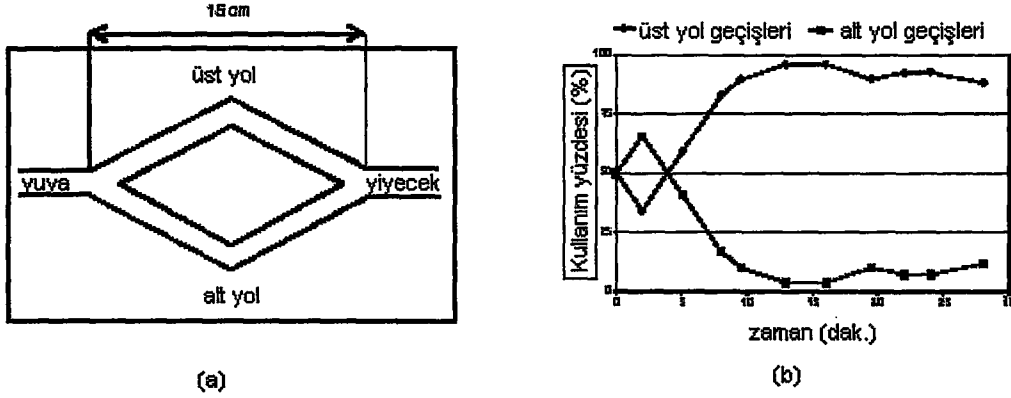


Şekil 3.1 Engelle Karşılaşan Karıncaların Seçimi ve En Kısa Yolu Bulmaları

- (A) Karıncaların izlediği yol. (B) Karıncaların engelle karşılaşması.
(C) Engelle karşı karıncaların seçimi. (D) Karıncaların kısa yolu bulmaları.

Karıncaların pheromone izini takip etme davranışlarının kontrollü koşullar altında izlendiği tek köprü deneyi sonuçları Şekil 3.2' de bulunmaktadır. Bu çalışmada yuva ve yiyecek kaynağı arasında eşit uzunlukta iki adet köprü bulunmaktadır. Karıncalar serbest bırakılarak hangi köprüyü kullandıkları yüzde olarak ölçülmüştür. Başlangıç

durumunda yollar eşit olasılıkla seçilmekte ancak karıncaların yolları eşit olasılıkla seçme davranışlarındaki dalgalanmalar ilerleyen zaman içerisinde yollardan birinin daha fazla seçimi ile sonuçlanmıştır (Dorigo ve diğ., 1999).



Şekil 3.2 Tek Köprü Deneyi

- (a) Deney düzeneği.
- (b) Karıncaların zaman içinde köprüleri kullanım yüzdesi.

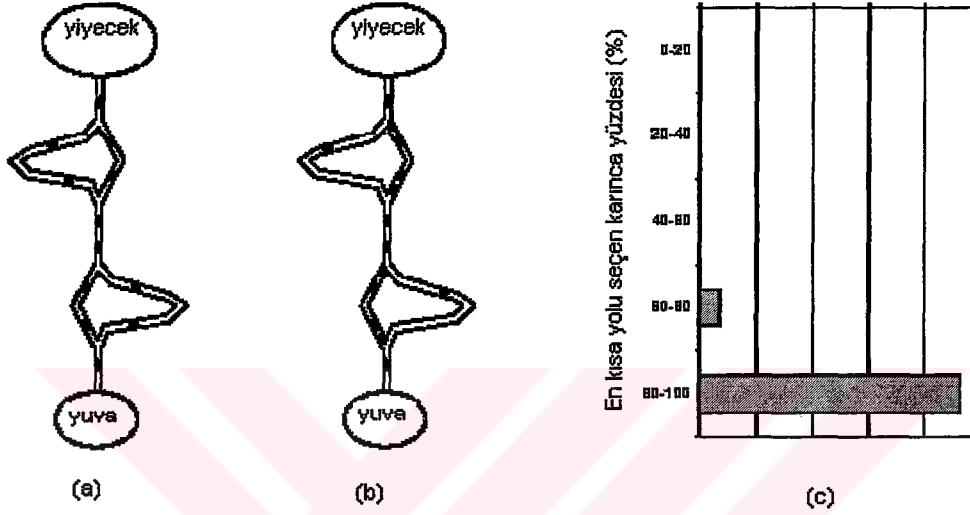
Bu fenomeni tanımlayan bir olasılık modeli Dorigo, Di Caro ve Gambardella (1999) tarafından açıklanmıştır. Bir yoldaki pheromone miktarı, o yolu geçmişte kullanmış karınca sayısı ile orantılıdır. Bu varsayım pheromone buharlaşmanın dikkate alınmadığını ifade etmektedir. Bu modelde, belirli bir zamanda bir yolu seçme olasılığı, o yoldaki toplam pheromone miktarına bağlıdır, yani; o ana kadar o yolu kullanmış karınca sayısının bir oranıdır. U_m ve L_m , m karınca köprüyü geçtikten sonra, alt ve üst yolu kullanmış karıncaların sayısıdır ($U_m + L_m = m$).

$(m+1)$. karıncanın üst yolu seçme olasılığı $P_U(m)$ aşağıdaki gibi tanımlanır:

$$P_U(m) = \frac{(U_m + k)^h}{(U_m + k)^h + (L_m + k)^h} \quad (3.1)$$

Karıncanın alt yolu seçme olasılığı $P_L(m)$ ise, $P_L(m) = 1 - P_U(m)$. Bu olasılık formu, iz-takip denemelerinden ortaya çıkmıştır. Parametre h ve k , kullanıcılara deneysel verileri model için uygun hale getirmelerini sağlar. Eğer $\psi < P_U$ ise, U_{m+1} şöyle güncellenir: $U_{m+1} = U_m + 1$, aksi halde, $U_{m+1} = U_m$ olur. Burada ψ , $[0,1]$ aralığında düzgün dağılan rasgele bir değişkendir.

Benzer bir çalışma, karıncaların pheromone izini takip etme davranışlarının farklı uzunluğa sahip iki köprü üzerinde izlendiği çift köprü deneyidir (Şekil 3.3). Bu çalışmada yuva ve yiyecek kaynağı arasında farklı uzunlukta iki adet köprü bulunmaktadır. Karıncalar serbest bırakıldığında çoğunlukla en kısa yol tercih etmişlerdir. Önceki durumdaki gibi, aynı pheromone bırakma mekanizması nedeniyle, en kısa yol tercih edilmiştir.



Şekil 3.3 Çift Köprü Deneyi

- (a) Karıncalar iki köprüde aramaya başlar
- (b) Karıncaların çoğu en kısa yolu tercih eder
- (c) En kısa yolu seçen karıncaların yüzde dağılımı

Bu en kısa yolu bulma davranışı, her karıncanın çok küçük katkılar sağladığı bir tür dağıtılmış eniyileme mekanizması (distributed optimization mechanism) ile açıklanabilir. Tek bir karınca, bir çözüm bulma yeteneğinde olmasına rağmen (yuva ile yiyecek alanı arasında), sadece karınca kolonisi adı verilen karınca grupları, en kısa yolu bulma davranışını göstermektedir. Bu davranış, karınca kolonisinin gelişmiş özelliğidir. Karıncalar bunu stigmergy olarak bilinen pheromone bırakılarak gerçekleştirilen dolaylı iletişim biçimini kullanarak mümkün kılmaktadırlar.

Stigmergy, diğer iletişim yollarından şu özellikleri ile farklıdır:

- Böceklerin iletişimi ile ortaya çıkan fiziksel bir bilgi olması,
- Oluşan bilginin yerel olması, yani sadece ortaya çıktığı yeri ziyaret eden böceklerin bu bilgiye ulaşabilmesi.

Karıncaları en kısa yolu bulma davranışına zorlayan diğer önemli bir özellik, otokataliz (Autocatalysis, positive feedback) mekanizması ve çözümün tam değerlendirmesi (implicit evaluation) arasındaki birleşimdir. Tam çözüm değerlendirme şöyle açıklanabilir: En kısa yollar (yapay karıncalar için en düşük maliyetli olan yollar) en uzun olanlardan daha çabuk tamamlanacak ve böylece, bu yollar daha çok pheromone takviyesi alacaklardır. Otokatalitik özelliği, tam çözüm değerlendirmesi ile birleştiğinde çok etkin olabilir. Yol ne kadar kısa olursa, karıncalar tarafından pheromone o kadar erken depolanır, en kısa yolu kullanan karınca o kadar çok olur. Tam olarak kullanıldığında, otokataliz, nüfus-tabanlı eniyileme algoritmaları için çok güçlü bir araç olmaktadır. Bu mekanizma en iyi bireylere ekstra bir önem verir ve böylece bunlar, arama sürecini yönetebilirler. Ancak, bu özellik kullanılırken, yakınsamadan (converge) (durağanlıktan (stagnation)) kaçınmak için, dikkatli olmak gerekmektedir. Durağanlık, iyi olmayan bireylerin nüfusu yönettiği ve arama sürecinde baskın oldukları tesadüfi bir durumdur. Bu durum arama uzayının daha fazla araştırılmasını önler (Dorigo ve diğ., 1999).

3.4 Gerçek ve Yapay Karıncaların Benzerlik ve Farklılıkları

Birçok KKE temel felsefesi, gerçek karıncalardan esinlenilerek oluşturulmuştur. Karıncaların bir takım özellikleri aynen kullanılarak ve bazı eklemelerle gerçek problemlerin çözümünde kullanılabilir hale getirilmiştir (Dorigo ve diğ., 1999; Cordón ve diğ., 2002):

Aynen alınan özellikler:

- Kolonideki bireyler arasında kurulan iletişim.
- Pheromone izi ve stigmergy: Pheromone, karıncalar arasında iletişim kanalıdır. Bu stigmergetic iletişim biçimi, ortak bilginin kullanımında önemli bir rol oynar.
- En kısa yolun araştırılması ve yerel hareketler: Karıncalar yuvaları ile yiyecek yeri arasında en kısa yolu bulmaya çalışırlar. Gerçek karıncalar atlamazlar. Yani, bitişik bölgelere yürürler. Yapay karıncalar da bitişik bölgelere adım-adım (step-by-step) hareket ederler.

- Stokastik geiş yn: Yapay karıncalar gerek karıncalar gibi bitişik blgeler boyunca hareket etmek iin bir olasılık karar politikası uygulayarak zm oluřtururlar.

Eklenen zellikler:

- Zamanın ayrıık olarak hesaplandığı bir ortamda yařarlar.
- Yapay karıncalar, isel bir durumdadırlar. Bu zel durum, karıncaların gemiřteki hareketlerinin hafızasını ierir.
- Yapay karıncalar, bulunan zmn kalitesinin bir fonksiyonu olacak miktardaki pheromone izini depolar.
- Yapay karıncaların pheromone bırakma zamanı probleme baėlıdır. rneėin, bir ok durumda karıncalar, bir zm oluřturduktan sonra pheromone izini gnceller.
- Tm sistemin etkinliėini arttırmak iin, KKE algoritmaları, ileriye bakma (lookahead), yerel eniyileme, geriye doėru izleme (backtracing) vb. ek zelliklerle glendirilebilir.

3.5 KKE Metasezgiseli

Karıncalar, $G(\mathcal{C}, \mathcal{L})$ zerinde hareket ederek zm oluřturan stokastik yapı sreleri olarak karakterize edilebilirler. Karıncalar, G zerinde keyfi olarak hareket etmeyip, Ω problem kısıtlarının bir fonksiyonu olan yapısal bir politika takip ederler. Genelde, karıncalar, uygun zmleri bulmaya alıřırlar ancak gerekirse uygun olmayan zmler de retebilirler. $c_i \in \mathcal{C}$ bileřenleri ve $l_{ij} \in \mathcal{L}$ iliřkileri karıncaların kendileri tarafından gncellenen ve tm karınca sistemi hakkında uzun dnemli hafızayı tutan bir pheromone izi τ ve problemin tanımını veya karıncalardan bařka bir kaynaktan saėlanan alıřma bilgisi hakkında ncl bilgiyi gsteren sezgisel deėer η ile tanımlanabilir. Bu deėerler, $G(\mathcal{C}, \mathcal{L})$ zerinde nasıl hareket edileceėi konusunda olasılık kararı vermek iin karıncalar tarafından sezgisel kural olarak kullanılır.

Kolonideki her k karıncası ařaėıdaki zelliklere sahiptir (Sttzle ve Dorigo, 2003):

- Minimum maliyetli s uygun çözümünü bulmak için $G(\mathcal{C}, \mathcal{L})$ üzerinde kullanılır. Yani, $\hat{f}_s = \min_s f(s, t)$ olacak şekildeki s çözümleri.
- Takip ettiği yollar hakkında bilgi depolamak için bir tabu_k hafızasına sahiptir. Bu hafıza, uygun çözüm oluşturmak, bulunan çözümü değerlendirmek ve pheromone depolamak amacıyla geriye doğru yolları tekrarlamak için kullanılabilir.
- Bir başlangıç konumu χ_s^k ve bir veya daha fazla bitiş konumlarına e^k atanabilirler. Genellikle, başlangıç konumu birim sıralama yani, tek bir bileşen veya boş bir sıralama şeklinde tanımlanabilir.
- $\chi_r = \langle \chi_{r-1}, i \rangle$ konumunda iken uygun N_i^k komşuluğu içindeki herhangi bir j düğümüne hareket etmeye çalışır. Yani, $\langle \chi_r, j \rangle \in \bar{X}$ konumuna. Bu mümkün değilse, karınca, uygun olmayan bir durum oluşturarak uygun olmayan IN_i^k komşuluğu içindeki herhangi bir j düğümüne hareket edebilir (yani, $\langle \chi_r, j \rangle \in X \setminus \bar{X}$ konumuna).
- Hareketi, bir olasılık karar kuralı uygulayarak seçerler. Bu olasılık karar kuralı, yerel olarak mevcut pheromone izlerinin ve sezgisel değerlerinin, geçmiş bilginin depolandığı karıncanın özel hafızasının ve problem kısıtlarının bir fonksiyonudur.
- k karıncasının yapısal süreci, e^k bitiş durumlarından en az biri sağlandığında durur.
- O anki çözüme bir bileşeni eklendiğinde, ona izin verilen veya ilişkiye göre pheromone izini günceller. Buna, çevrim içi adım-adım pheromone güncelleme (online step-by-step pheromone update) adı verilir.
- Bir çözüm oluşturulur oluşturulmaz, aynı yolu geriye doğru tekrarlayabilir ve kullanılan bileşenlerin ve ilişkilerin pheromone izlerini günceller. Buna, çevrim içi gecikmiş pheromone güncelleme (online delayed pheromone update) adı verilir.

Karınca faaliyetlerinin yanında, bir KKE algoritması, iki süreç içerir: pheromone izi buharlaştırma ve daemon faaliyetleri. Pheromone buharlaştırma, bileşenler üzerindeki pheromone izi yoğunluğunun zaman içinde azalması nedeniyle

gerçekleşir. Pheromone buharlaştırma, algoritmanın yarı-eniyi bölgeye doğru eğilimini önlemek için gereklidir. Daemon faaliyetleri, tek bir karınca tarafından gerçekleştirilemeyen merkezi faaliyetleri uygulamak için kullanılabilir. Örneğin, karıncaların oluşturduğu çözümlere yerel arama uygulama vb. faaliyetlerdir.

Karınca sistemi ile arama uzayı içerisinde eniyi sonucu arayan ajanlar arasında iletişim sağlanarak, arama esnasında yapılan iyi seçimlerden diğer ajanların da haberdar olması sağlanmış ve böylece çok ajanlı bir rasgele aramadan çok daha iyi sonuçlar elde edilmesi mümkün kılınmıştır. Birden fazla ajanın arama uzayında rasgele arama yapmasının ötesinde aralarındaki sinerjinin ortaya çıkması sağlanmıştır. Karınca sisteminin en önemli bileşenlerinden biri izlerin yönetimidir. Standart bir karınca sisteminde izler yeni çözümler oluşturmak için bir rehber olarak kullanılır. Bir çözüm elde edildiğinde önce tüm izler buharlaşarak zayıflatılır, sonra çözüm yolu üzerinde bulunan izler çözümün kalitesine bağlı oranda güçlendirilir. Karınca sisteminde kullanılan iz yapısı yerel bir eniyiye yakalanma ve salınma girme olasılığını azaltır. İz yapısı ile birlikte kullanılan klasik sezgisel yöntem ise global eniyiye arama esnasında yerel olarak da arama yapılmasını sağlar. Özellikle aramanın başlangıç safhasında ajanlara iyi çözümler üretmede yardımcı olur. KKE metasezgiseli Şekil 3.4’ de tanımlanmıştır (Dorigo ve diğ., 1999).

3.6 KKE Metasezgiseli Uygulamaları

KKE metasezgisel yönteminin kullanıldığı çalışmaların bir sınıflandırması Tablo 3.1’ de referans verilerek, Tablo 3.2’ de sadece yazar adı ve yılı belirtilerek verilmiştir. Bu algoritmaların büyük bir kısmı temel olarak ilk çalışma olan karınca sistemi algoritmasını kullanmaktadır. Daha sonraki çalışmalar genellikle uygulama alanına bağımlı olarak, özellikle izlerin güncellenmesi ve yapay karıncaların hareket ettirilmesindeki süreçlerde farklılıklar içermektedir. Diğer uygulamalarda karınca koloni sistemi, Ant-Q algoritması, max-min karınca sistemi, sıralamaya dayalı karınca sistemi gibi algoritmalar kullanılmıştır.

Tablo 3.3’ de bu algoritmalar kullanılarak yapılan uygulama çalışmaları özetlenmiştir.

Procedure KKE metasezgiseli

While bitiş koşullarına ulaşılmamış **do**

Çizelgeleme Faaliyetleri

KarıncaFaaliyetleriYönetimi()

İzBuharlaştırma()

DaemonFaaliyetleri() {opsiyonel}

end Çizelgeleme Faaliyetleri

EndWhile

EndProcedure KKE metasezgiseli

Procedure KarıncaFaaliyetleri()

While mevcut kaynaklar **do**

YeniKarıncaOluşturma();

YeniKarıncaFaaliyeti();

EndWhile

EndProcedure

Procedure YeniKarıncaFaaliyeti() {karınca yaşam çevrimi}

KarıncaBaşlangıçKonumunaYerleştir();

tabu_k = KarıncaHafızaGüncelle();

While Aşama ≠ Hedef aşama **do**

A = YerelKarıncaYörüngeTablosunuOku();

P = YerdeğiştirmeOlasılığınıHesapla(A; tabu_k; problem kısıtları);

Sonraki aşama = KarıncaKararınıUygula(P; problem kısıtları);

SonrakiAşamayaGit(sonraki aşama);

If (adım-adım iz güncelle)

GidilenYollardakiİzleriDepola();

KarıncaYörüngeTablosunuGüncelle();

EndIf

tabu_k = AşamayıGüncelle();

EndWhile

If (ertelenen iz güncelle)

ÇözümüDeğerlendir();

GidilenYollardakiTümİzleriDepola();

KarıncaYörüngeTablosunuGüncelle();

EndIf

Yoket();

EndProcedure

Şekil 3.4 KKE Metasezgiseli

Tablo 3.1 KKE Algoritmaları Çalışmaları-1

Problem İsmi	Algoritma İsmi	Referanslar
Gezgin satıcı problemi	AS	Dorigo ve diğ., 1991a,1991b, 1996; Colorni ve diğ., 1991, 1992
	Ant-Q	Gambardella ve Dorigo, 1995
	ACS & ACS-3-opt	Gambardella ve Dorigo, 1996; Dorigo ve Gambardella, 1997a, 1997b
	<i>MMAS</i>	Stützle ve Hoos, 1996, 1997, 1998a, 2000
	AS _{rank}	Bullnheimer ve diğ., 1999a
	<i>MMAS</i>	Stützle ve Dorigo, 1999b
	BWAS	Cord' on ve diğ., 2000
	MAPLE	Guerrieri, 2000
	ACSGA-TSP	Pilat ve White, 2002
	MOTSP_ACO	Cardoso ve diğ., 2003
	DACS	Li ve Gong, 2003
	ACOMAC	Tsai ve diğ., 2003
Kuatratik atama problemi	AS-QAP	Maniezzo ve diğ., 1994
	HAS-QAP	Gambardella, Taillard ve Dorigo, 1999
	<i>MMAS</i> -QAP	Stützle, 1997; Stützle ve Dorigo, 1999a
	ANTS-QAP	Maniezzo, 1999
	AS-QAP ^b	Maniezzo ve Colorni, 1999
	FANT	Merz ve Freisleben, 1999
	ANTabu	Talbi, Roux ve diğ., 2001
Araç rotalama (vehicle routing)	AS-VRP	Bullnheimer ve diğ., 1999b, 1999c
	MACS-VRPTW	Gambardella, Taillard ve Agazzi, 1999.
	MOACS-VRPTW	Barán ve Schaerer, 2003

Tablo 3.1 KKE Algoritmaları Çalışmaları-1 (Devam)

Problem İsmi	Algoritma İsmi	Referanslar
Çizelgeleme problemleri	AS-JSP	Colomi ve diğ., 1994
	AS-FSP	Stützle, 1998a
	ACS-SMTTP	Bauer ve diğ., 1999
	ACS-SMTWTP	den Besten ve diğ., 2000
	ACO	Gagné ve diğ., 2000, 2001, 2002
	ACO-RCPS	Merkle ve diğ., 2000a
	SMTWTP, SMTWDP	Merkle ve Middendorf, 2000, 2001, 2003
	ACO 1-6	McMullen, 2001
	ACSA	Yu ve Song, 2001
	SACO	TKindt ve diğ., 2000, 2002
	FOPSSP-ACO	Blum, 2002
	ACO	Gravel ve diğ., 2002
	ACO	Gottlieb ve diğ., 2003
	ACBOA	Baykasoğlu ve diğ., 2003
	ANT	McMullen ve Tarasewich, 2003
	AS-JSS	Leguizamón ve Schütz, 2003
	ACS	Ying ve Liao, 2003, 2004
	Beam-ACO-OSS	Blum, 2004
	PACO	Rajendran ve Ziegler, 2004
Sequential ordering	HAS-SOP	Gambardella ve Dorigo, 1997, 2000
Layout	ACO	Solimanpur ve diğ., 2003a, 2003b
Shortest common supersequence	AS-SCS	Michels ve Middendorf, 1998

Tablo 3.2 KKE Algoritmaları Çalışmaları-2 ²

Problem İsmi	Algoritma İsmi	Yazarlar	Yıl
Connection-oriented network routing	ABC	Schoonderwoerd ve diğ.	1996
	ASGA	White ve diğ.	1998
	AntNet-FS	Di Caro ve Dorigo	1998
	ABC-smart ants	Bonabeau ve diğ.	1998
Connection-less network routing	AntNet & AntNet-FA	Di Caro ve Dorigo	1997
	Regular ants	Druschel ve Chen	1997
	CAF	Heusse ve diğ.	1998
	ABC-backward	Van der Put ve Rothkrantz	1998
Graph coloring	ANTCOL	Costa ve Hertz	1997
Frequency assignment	ANTS-FAP	Maniezzo ve Carbonaro	1998
Generalized assignment	MMAS-GAP	Ramalhinho Lourenço ve Serra	1998
Multiple knapsack	AS-MKP	Leguizam'on ve Michalewicz	1999
Optical networks routing	ACO-VWP	Navarro Varela ve Sinclair	1999
Redundancy allocation	ACO-RAP	Liang ve Smith	1999
Constraint satisfaction	Ant-P-solver	Solnon	2000

² Stützle ve Dorigo, 2003, Dorigo ve diğ., 1999, Dorigo ve Di Caro, 1999

Tablo 3.3 KKE Metasezgiseli Uygulamaları³

Yazarlar	Yıl	Algoritma İsmi	Problem Tipi
Gravel, Price ve Gagne	2001	ACO	Alüminyum dökümhanesinde atölye çizelgeleme
Bland	2001	ACO	Yapısal dizayn problemi
Jayaraman ve diğ.	2001	ACO	Bioreactor eniyileme
Doerner, Hartl ve Reimann	2001	ACO	Yükleme ve dağıtım problemleri
Doerner, Hartl ve Reimann	2001	ACO	Dolu kamyon yük taşıma problemleri
Jong ve Wiering	2001	Çoklu ACS	Park yeri tahsis problemi
Baboglu, Meling ve Montresor	2001	Anthill	Peer-to-peer (P2P) ağları
Cicirello	2001	AC2	Atölye rotalama
Ramos ve Almeida	2000	Kavrayıcı harita modeli	Görüntü bölümlenme-desen düzenleme
Tzafestas	2000	Boyacı Karıncalar	Dijital sanat
Monmarché, Venturini ve Slimane	2000	API	Nümerik eniyileme
Zhou ve Liu	1999	Akıllı Karınca Algoritması	Telekomünikasyon ağlarında dinamik rotalama
Bland	1999	AS (TS)	Yerleşim planı
Bland	1999	ACO	Space planlama
Stützle	1998a	ACO	Akış tipi üretim
Forsyth ve Wren	1997	AS	Araç çizelgeleme

³ Krishnaiyer ve Cheraghi, 2002

3.7 Mevcut KKE Algoritmaları

Bugüne kadar çok sayıda KKE algoritması geliştirilmiştir:

- Karınca sistemi (Ant System-AS)
- Karınca koloni sistemi (Ant Colony System-ACS)
- Ant-Q algoritması (Ant-Q)
- Max-Min karınca sistemi (Max-Min Ant System- $\mathcal{MMAS}$)
- Sıralamaya dayalı karınca sistemi (Rank based Ant System, AS_{rank})

3.7.1 İlk KKE Algoritması: Karınca Sistemi

KKE algoritmaları için ilk uygulama gezgin satıcı probleminin test problemi olarak kullanılmasıyla yapılmıştır (Dorigo ve diğ., 1991a, 1991b, 1996; Colormi ve diğ., 1991, 1992). İlk olarak karınca algoritmasının genel özellikleri ve karınca-yoğunluk (ant-density), karınca-miktar (ant-quantity) ve karınca-çevrim (ant-cycle) algoritmaları olarak adlandırılan üç farklı versiyonu tanımlanacaktır.

Her şehirden yalnızca bir defa geçmek şartıyla ve tüm şehirleri ziyaret etmek için olası olan kapalı turlardan minimum tur uzunluğunu bulmayı hedefleyen gezgin satıcı problemleri için karınca sistemi ile eniyi çözümü bulmak için her karınca;

- Karınca hareket edeceği şehri seçerken, uzaklık ve mevcut izleri içeren bir fonksiyon ile seçim olasılığını belirlemektedir.
- Tur tamamlandığında geçilen her (i,j) yoluna iz bırakılır.
- Karıncanın bir şehirden sadece bir defa geçmesini sağlamak için tur tamamlanana kadar geçilen şehirler bir yasaklar listesinde (tabu list-tabu_k) saklanır. Liste aynı zamanda tur tamamlandığında karıncanın bulduğu çözümü gösterir.

Toplam karınca sayısı m şu şekilde belirlenir:

$$m = \sum_{i=1}^n b_i(t) \quad (3.2)$$

N : tüm şehirlerin kümesi, $N = \{1, \dots, n\}$

$b_i(t)$: t zamanında i şehirde bulunan karınca sayısı ($i=1, \dots, n$)

Yollar üzerine bırakılan izler:

$$\tau_{ij}(t+1) = \rho \tau_{ij}(t) + \Delta \tau_{ij}(t, t+1) \quad (3.3)$$

$\tau_{ij}(t)$: t anında (i,j) yolunda bulunan iz miktarı, 0 zamanında, $\tau_{ij}(0)$, küçük bir pozitif c değeri alabilir.

ρ : $t+1$ zamanında i şehrinde j şehrine kadar izlerin buharlaşma katsayısıdır, aşırı iz birikimini önlemek için $0 \leq \rho < 1$ olmalıdır.

$\Delta \tau_{ij}^k$: (i,j) yoluna k karıncasının t ile $t+1$ zaman aralığında bıraktığı iz miktarı:

$$\Delta \tau_{ij}(t, t+1) = \sum_{k=1}^m \Delta \tau_{ij}^k(t, t+1) \quad (3.4)$$

Başlangıçta, karıncalar değişik şehirler üzerinde bulunur ve yollar üzerinde $\tau_{ij}(0)$ miktarında iz bulunmaktadır.

Bir k karıncasının n farklı şehre gitme kısıtını (n -şehir turu) sağlamak için, karıncaların t zamanına kadar şehirler hafızaya alınır (tabu_k) ve bir tur tamamlanmadan önce uğradığı şehre tekrar uğraması yasaklanır. Tur tamamlandığında tabu_k listesi boşaltılır ve karınca yolunu seçmekte tekrar özgür olur.

t zamanında k karıncasının i şehrinde j şehrine gitme olasılığı p , rasgele-oran kuralı (random-proportional rule) adı verilen kural ile belirlenir ve bu olasılık dahilinde seçim yapılır. Bu geçiş kuralı şöyledir:

$$p_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{j=1}^n [\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta} \quad j \notin \text{tabu}_k \quad (3.5)$$

$\eta_{ij} = 1/d_{ij}$: karıncanın görme özelliği olarak tanımlanan sezgisel bilgi,

d_{ij} : i ve j şehirleri arasındaki uzaklık

α ve β : iz ve görüş arasındaki ağırlık oranını kontrol eden parametreler

Böylece her karınca bir sonraki adımda gideceği şehri seçerken, aday şehirleri uzaklıklarına göre ve daha önceki turlarda tercih edilme sıklıklarına göre değerlendirerek bir olasılık belirlemektedir.

Karınca-Yoğunluk ve Karınca-Miktar Algoritmaları:

Şekil 3.5' de, karınca-yoğunluk ve miktar modelleri tanımlanmıştır.

Karınca-yoğunluk modelinde;

$$\Delta\tau_{ij}^k(t, t+1) = \begin{cases} \frac{Q_1}{d_{ij}} & \text{eger } k \text{ karıncası } (i,j) \text{ yolunu kullanırsa} \\ 0 & \text{diger} \end{cases} \quad (3.6)$$

Q_1 : i den j ye giden karıncanın, yola her birim uzunluk için bıraktığı iz miktarı

Karınca-miktar modelinde;

$$\Delta\tau_{ij}^k(t, t+1) = \begin{cases} Q_2 / d_{ij} & \text{eger } k \text{ karıncası } (i,j) \text{ yolunu kullanırsa} \\ 0 & \text{diger} \end{cases} \quad (3.7)$$

Q_2/d_{ij} : i den j ye giden karıncanın, yola her birim uzunluk için bırakılan iz miktarı

Karınca-Çevrim Algoritması:

Şekil 3.6' da, karınca-çevrim modeli tanımlanmıştır.

Bu durumda, $\Delta\tau_{ijk}$ her adımda hesaplanmaz ancak, bir tur (n adım) sonra hesaplanır:

$$\Delta\tau_{ij}^k(t, t+n) = \begin{cases} \frac{Q_3}{L^k} & \text{eger } k \text{ karıncası } (i,j) \text{ yolunu kullanırsa} \\ 0 & \text{diger} \end{cases} \quad (3.8)$$

Q_3 : sabit bir sayı

L^k : k karıncasının oluşturduğu turun uzunluğu

Her hareket sonrasında izleri güncellemek yerine tüm çevrim sonunda bu yapılmaktadır. Bu noktada izlerin güncelleştirilmesi şu formül ile sağlanmaktadır:

$$\tau_{ij}(t+n) = \rho\tau_{ij}(t) + \Delta\tau_{ij}(t, t+n) \quad (3.9)$$

$$\Delta\tau_{ij}(t, t+n) = \sum_{k=1}^m \Delta\tau_{ij}^k(t, t+n) \quad (3.10)$$

Adım1. Başla:

$t := 0,$

Her ij yolundaki iz miktarı için başlangıç $\tau_{ij}(0)$ değerlerini,

i düğümündeki $b_i(t)$ karıncalarını,

Her i ve j için $\Delta\tau_{ij}(t, t+1) = 0$ belirle,

Adım2. Repeat

Adım2.1 For $i:=1$ to n do

For $k:=1$ to $b_i(t)$ do

Denklem(3.5) kullanılarak p_{ij} olasılığı ile hareket
edilecek şehri seç ve hareket et,

k karıncasının tabu_k listesine seçilen şehri ekle,

$\Delta\tau_{ij}^k(t, t+1)$ değerlerini hesapla,

Adım2.2 $\tau_{ij}(t+1)$ ve $p_{ij}(t+1)$ değerlerini Denklem(3.3) ve (3.4) kullanılarak
hesapla,

Until tabu_k dolduğunda

Adım3. Bulunan en kısa yolu hafızaya al ve tüm tabu_k listelerini boşalt,

Adım4. If not (End-Test) {döngü sayısı}

then

$t := t + 1,$

Her i ve j için $\Delta\tau_{ij}(t, t+1) = 0,$

Adım2' ye git,

else

En kısa yolu göster ve dur.

Şekil 3.5 Karınca-Yoğunluk ve Karınca-Miktar Algoritmaları

Adım1. Başla:

$t:=0$, Her ij yolundaki iz miktarı için başlangıç $\tau_{ij}(0)$ değerlerini,

i düğümündeki $b_i(t)$ karıncalarını,

Her i ve j için $\Delta\tau_{ij}(t, t+n) = 0$ belirle,

Adım2. Repeat

Adım2.1 For $i:=1$ to n do

For $k:=1$ to $b_i(t)$ do

Denklem(3.5) kullanılarak p_{ij} olasılığı ile hareket edilecek
şehri seç ve hareket et,

k karıncasının tabu _{k} listesine seçilen şehri ekle,

Adım2.2 $\Delta\tau_{ij}^k(t, t+n)$ değerlerini hesapla,

Adım2.3 $\Delta\tau_{ij}(t, t+n) = \sum_{k=1}^m \Delta\tau_{ij}^k(t, t+n)$ değerlerini hesapla,

**Adım2.4 $\tau_{ij}(t+1)$ ve $p_{ij}(t+1)$ değerlerini Denklem(3.3) ve (3.4) kullanılarak
hesapla,**

Until tabu _{k} dolduğunda

Adım3. Bulunan en kısa yolu hafızaya al ve tüm tabu _{k} listelerini boşalt,

Adım4. If not (End-Test) {döngü sayısı}

then

$t:=t+n$, Her i ve j için $\Delta\tau_{ij}(t, t+n) = 0$,

Adım2' ye git,

else

En kısa yolu göster ve dur.

Şekil 3.6 Karınca-Çevrim Algoritması

Diğer genel amaçlı sezgisel yöntemlerle karşılaştırıldığında karınca sisteminin büyük ölçekli problemlerde daha zayıf olduğu gözlenmiştir. Bu nedenle karınca sisteminin güçlendirilmesi üzerine çalışmalar yapılmıştır. Bunlardan ilki, elitist stratejisi adı verilen özelliğin karınca sisteminde uygulanmasıdır (Dorigo ve diğ., 1991a, 1991b, 1996, 1999; Bullnheimer ve diğ., 1999a; Stützle ve Dorigo, 1999b, 2003). Bu strateji algoritmanın çalışmaya başladığı andan itibaren bulunan en iyi çözüme (T^{gb} , burada gb , global-en iyiyi temsil etmektedir) ek bir ağırlık verilmesinden oluşur. Böylece, Denklem(3.8) aşağıdaki duruma gelir:

$$\Delta\tau_{ij}^{gb}(t) = \begin{cases} e \cdot \frac{Q_3}{L^{gb}} & \text{eger } (i,j) \in T^{gb} \\ 0 & \text{diger} \end{cases} \quad (3.11)$$

Böylece, T^{gb} 'nin yollarına $e \cdot Q_3 / L^{gb}$ kadar takviye yapılır. Burada, L^{gb} , T^{gb} 'nin uzunluğu ve e , elitist karınca sayısıdır.

3.7.2 Karınca Koloni Sistemi

Karınca koloni sistemi, karınca sisteminin performansını arttırmak amacıyla oluşturulmuştur (Gambardella ve Dorigo, 1996; Dorigo ve Gambardella, 1997a, 1997b ; Stützle ve Dorigo, 1999b; Dorigo ve diğ., 1999).

ACS, karınca sistemden 4 temel açıdan farklıdır. Bunlar:

- ACS, AS' den daha girişken (aggressive) bir geçiş olasılığı kullanır.
- Pheromone, sadece global-en iyi çözüme ait yollar için bırakılır.
- Bir karınca i şehrinden j şehrine gitmek için (i,j) yolunu kullanırsa, bu yoldan belli miktarda pheromone azaltılır.
- Aday liste kullanımı sağlanmıştır.

ACS şu şekilde çalışır: m karıncaları, belirli bir başlangıç kuralına (örneğin, rasgele) göre seçilen n şehirlerine yerleştirilir. Her karınca geçiş kuralını uygulayarak bir tur oluşturur. Ayrıca karınca yerel güncelleme kuralını uygulayarak geçtiği şehirlerdeki pheromone miktarını değiştirir. Tüm karıncalar turlarını tamamlar tamamlamaz, global güncelleme kuralını uygulayarak tekrar pheromone miktarını değiştirir.

ACS Geçiş Kuralı:

Karınca koloni sistemi ile bir çizelge oluşturulurken, karıncalar, i şehrinde aşağıdaki geçiş olasılığı kuralını uygulayarak hareket edeceği bir sonraki j şehrini seçer:

$$j = \begin{cases} \arg \max_{u \in S_k(i)} \{ [\tau_{iu}(t)]^\alpha [\eta_{iu}]^\beta \} & \text{eger } q \leq q_0 \quad (\text{devamlılık}) \\ J & \text{diger} \quad (\text{arastirma}) \end{cases} \quad (3.12)$$

Burada, $S_k(i)$, i şehrindeki karınca tarafından ziyaret edilebilecek şehirlerin kümesini gösterir. q , $[0,1]$ aralığında düzgün dağılan rasgele bir sayı ve q_0 , araştırmaya karşılık devamlılığın göreceli önemini gösteren bir parametredir ($0 \leq q_0 \leq 1$). Ek olarak, J , i şehrindeki k karıncasının, rasgele-oran kuralına göre seçeceği j şehrinin olasılığını veren rasgele bir değişkendir.

$$p_{ij}^k(t) = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{u \in S_k(i)} [\tau_{iu}(t)]^\alpha [\eta_{iu}]^\beta} & \text{eger } j \in S_k(i) \\ 0 & \text{diger} \end{cases} \quad (3.13)$$

Denklem (3.12) ve (3.13) ile gerçekleştirilen ve sanki-rasgele-oran kuralı (pseudo-random-proportional rule) adı verilen geçiş kuralı, fazla miktarda pheromone bulunan en kısa yola sahip şehri seçme yönündedir (Gambardella ve Dorigo, 1996; Dorigo ve Gambardella, 1997b).

Pheromone iki farklı şekilde kullanılır: araştırma ve devamlılık. Araştırma, bileşim seçiminin, probleme olasılığa dayalı olarak çözüm oluşturmak için kullanıldığı stokastik bir süreçtir. Devamlık, pheromone izi değerleri ile kısmi amaç fonksiyonu değerinin karışımını en büyükleyen bileşenleri seçer. Karınca algoritmalarının performansını arttırmak için en önemli faktörlerden biri bu iki mekanizma arasında bir denge sağlamaktır (Gambardella ve diğ., 1999).

Yerel Güncelleme Kuralı:

ACS' de bir çözüm (yani, tur) oluşturulurken, karınca, aşağıdaki yerel güncelleme kuralını uygulayarak geçilen yollar üzerindeki pheromone düzeyini azaltır:

$$\tau_{ij} := (1 - \varphi) \cdot \tau_{ij} + \varphi \cdot \tau_0 \quad (3.14)$$

Burada τ_0 , başlangıç pheromone düzeyi ve $\varphi (0 < \varphi < 1)$, pheromone buharlaştırma parametresidir.

Yerel güncelleme kuralının etkisi, turu farklılaştırmak amacıyla, yolların arzu edilebilirliğini dinamik olarak değiştirmektir. Eğer karıncalar, farklı yollar araştırırsa, bunlardan birinin daha güçlü çözüm bulma olasılığı, önceki en iyi çözümün dar komşuluğundaki tüm araştırmalarından daha yüksektir. Bir karınca, bir çözüm oluşturduğunda, yerel güncelleme kuralı, geçtiği yollardaki pheromone miktarını azaltır ve daha az cazip hale getirir. Böylece, bir karıncanın oluşturduğu çözüm yolları, diğer karıncalar tarafından çözüm oluştururken, daha düşük olasılıkla seçilecektir. Sonuç olarak, karıncalar, henüz gidilmemiş yolları tercih etme yönünde ve genel bir çözümden odaklaşmayı önlemektedirler (Ying ve Liao, 2003, 2004).

Global Güncelleme Kuralı:

ACS' de global güncelleme kuralı, tüm karıncalar çizelgelerini tamamladıktan sonra yapılır. Aramayı doğru şekilde yapmak için, global güncelleme kuralı, en iyi çözümün yollarına büyük miktarda pheromone bırakılmasını sağlar. Böylece, sadece o anki iterasyonda en iyi çizelgeyi bulan karıncanın pheromone depolamasına izin verilir. Pheromone düzeyi, aşağıdaki gibi düzenlenir:

$$\tau_{ij}(t+1) := (1 - \rho) \cdot \tau_{ij}(t) + \rho \cdot \Delta \tau_{ij}^{gb}(t) \quad (3.15)$$

burada,

$$\Delta \tau_{ij}^{gb}(t) = \begin{cases} (L_{gb})^{-1} & \text{eger } (i, j) \in \text{en iyi çizelge} \\ 0 & \text{diger} \end{cases} \quad (3.16)$$

Yukarıdaki denklemde, $\rho (0 < \rho < 1)$, global güncellemenin pheromone buharlaştırma parametresi ve L_{gb} , global-en iyi çözümün uzunluğudur.

Global güncelleme kuralında, global-en iyi L_{gb} yerine iterasyon-en iyi L_{ib} de kullanılabilir (Gambardella ve Dorigo, 1996; Dorigo ve Gambardella, 1997a, 1997b; Ying ve Liao, 2003, 2004).

Aday Listesi:

ACS, aday listesi adı verilen ve ek yerel sezgisel bilgi sağlayan bir veri yapısı sağlamaktadır. Aday listesi, verilen bir şehirden tercih edilen gidilecek şehirlerin bir listesidir. Bir karınca, i şehrinde iken, i şehrinin tüm gidilmemiş komşuluklarını araştırmak yerine, aday listedeki şehirler arasında gidilecek şehri seçer. Aday listesinde gidilmemiş şehir kalmadıysa, diğer şehirler araştırılabilir. Bir şehrin aday listesi, uzaklığa göre artan şekilde sıralanmış cl şehirden oluşur (cl parametredir). Liste sırasıyla ve henüz gidilmiş şehirlere gidişi önlemek için karıncanın yasaklar listesine göre taranır (Dorigo ve Gambardella 1997b; Stützle ve Dorigo, 2003; Dorigo ve diğ., 1999).

3.7.3 Ant-Q Algoritması

Ant-Q algoritması, Gambardella and Dorigo (1995) tarafından önerilen ve KKE algoritması ile Reinforcement Learning (RL) tekniğinin özel bir türü olan Q-Learning algoritmasını birleştirir. ACS ve Ant-Q algoritmaları sadece bir açıdan farklılık göstermektedir. Ant-Q algoritmasında, pheromone izi güncelleme işlemi, bir sonraki aşamanın tahmini değerine göre yapılmaktadır (Dorigo ve diğ., 1999; Gambardella and Dorigo, 1995; Mariano Romero ve Morales Manzanares, 1999). Böylece ACS algoritması için kullanılan Denklem(3.14) şu şekilde değişmektedir:

$$\tau_{ij} := (1 - \phi) \cdot \tau_{ij} + \phi \cdot \gamma \cdot \max_{z \in S_k(j)} \tau_{jz} \quad (3.17)$$

Burada, γ discount faktörüdür.

3.7.4 Max-Min Karınca Sistemi

Max-Min karınca sistemi, Stützle ve Hoss (1996, 1997) tarafından oluşturulmuştur. $MMAS$, aşağıdaki özelliklerle AS algoritmasından farklılık gösterir (Stützle ve Hoss, 1996; 1997, 1998a, 1998b, 2000; Stützle, 1997, 1998a; Dorigo ve diğ., 1999; Stützle ve Dorigo, 1999b):

- Her iterasyondan sonra sadece en iyi karıncanın pheromone eklemesine izin verilir.
- Araştırmanın durağanlığından kaçmak için, pheromone iz değerleri $[\tau_{\min}, \tau_{\max}]$ aralığında sınırlandırılır. Yani, $\forall \tau_{ij} \quad \tau_{\min} \leq \tau_{ij} \leq \tau_{\max}$

- Algoritmanın başında, pheromone izleri, yüksek bir araştırmaya neden olan üst sınır iz limitine $\tau_{\max}$ eşit olacak şekilde başlatılır.

Pheromone İzlerinin Güncellenmesi:

MMAS algoritmasında, her iterasyon sonunda tüm karıncalar bir çözüm oluşturduktan sonra, sadece bir karıncanın iz güncellenmesine izin verilir. Bu karıncanın seçiminde iki olasılık vardır. İlki, algoritmanın her iterasyonu bittikten sonra, iterasyon-en iyi olan karıncanın iz düzeyini güncellenmesine izin verilir. Diğeri, en iyi çözüm (global-en iyi) iz güncelleştirme için seçilir. Güncelleme kuralı şu şekildedir:

$$\tau_{ij}(t+1) := \rho \cdot \tau_{ij}(t) + \Delta \tau_{ij}^{best} \quad (3.18)$$

Burada, $\Delta \tau_{ij}^{best} = 1/L_{best}^{best}$. Eğer izlerin güncellenmesi için iterasyon-en iyi karınca seçilirse, tur uzunluğu L_{best}^{ib} , global-en iyi karınca seçilirse, tur uzunluğu L_{best}^{gb} gözönüne alınır. İterasyon-en iyi turun seçilmesi, yeni turların daha fazla düzeyde araştırılmasını sağlarken, global-en iyi turun seçilmesi, araştırmada daha kısa sürede durağanlığa neden olabilir.

Maksimum ve Minimum İz Limitleri:

MMAS algoritmasında, yeni çözümlerin araştırılması ile araştırmanın sürdürülmesi arasında bir denge sağlayan en önemli araç iz değerlerine limitler $[\tau_{\min}, \tau_{\max}]$ oluşturulmasıdır. Böylece iz değerleri $\forall \tau_{ij} \quad \tau_{\min} \leq \tau_{ij} \leq \tau_{\max}$ olacak şekildedir. *MMAS* için sanki-rasgele-oran kuralı kullanıldığında, p olasılığı ne kadar yüksekse, iz limitleri o kadar sıkı olmalıdır, yani, $\tau_{\max} / \tau_{\min}$ oranı o kadar düşük olmalıdır. Eğer $\tau_{\max} / \tau_{\min}$ oranı çok yüksekse, hemen hemen her zaman maksimum τ_{ij} değerine sahip şehrin seçilmesine neden olur ki bu da kısa sürede araştırmanın durağanlaşmasıdır.

Başlangıç İzleri:

MMAS algoritmasında, başlangıç iz değerleri, tüm düğümler için mümkün maksimum iz olacak şekildedir. Bu şekilde, *MMAS* araştırma süreci şu şekilde

açıklanabilir: Algoritmanın her iterasyonundan sonra, iz miktarı, $\tau_{ij}(t+1) := \rho \cdot \tau_{ij}(t)$ kadar buharlaştırılarak azaltılır. Sadece en iyi karıncanın turu güncellemesine izin verildiği için, sadece en iyi tura ait yollardaki izlerin yoğunluğunun artırılmasına izin verilir. Böylece, hiç veya çok az takviye edilen yolların iz miktarı sürekli azalacaktır. Bu şekilde, *MMAS* geçmişte yapılan hatalardan kaçınmaya çalışır. Buradaki hata, kötü turlara yol açan yolların tercih edilmesidir. Kötü yollardaki iz yavaşça azalmaktadır ve sadece iyi yollar yüksek düzeyde bir iz miktarına sahip olabilir. Diğer bir avantaj, ρ parametresinin öğrenme hızı gibi yorumlanmasıdır. Yüksek ρ değeri, diğer yollardaki iz miktarı yavaş yavaş azaldığı için, iyi yolların daha yavaş öğrenilmesi gerçekleşmektedir. Yüksek ρ değeri için, izlerdeki fark daha düşük olduğu için, belirli yollara verilen takviye hatalarının daha kolay düzeltilmesini sağlar.

3.7.5 Sıralamaya Dayalı Karınca Sistemi

Karınca sistemi üzerinde diğer bir gelişme karınca sisteminin sıralamaya dayalı versiyonudur. *AS_{rank}*, Bullnheimer ve diğ. (1999a) tarafından geliştirilmiştir. *AS_{rank}* algoritmasında, karınca sisteminin elitist stratejisine benzer şekilde her zaman en iyi tur, pheromone izlerinin güncellemesi için kullanılır. Ek olarak, o anki iterasyonun en iyi karıncaların sayısı, pheromone eklenmesi için kullanılır. Bu amaçla, karıncalar, tur uzunluklarına göre ($L^1(t) \leq L^2(t) \leq \dots \leq L^m(t)$) sıralanır ve karıncanın depolayabileceği pheromone miktarı karıncanın r sırasına göre ağırlıklandırılır. Sadece her iterasyonda $(w-1)$ karıncanın pheromone depolamasına izin verilir. En güçlü geri beslemeyi veren global-en iyi çözüme, w ağırlığı verilir. O anki iterasyonun en iyi r karıncası, verilen $\max\{0, w-r\}$ ağırlığı ile pheromone güncellemeye katkıda bulunur. Böylece güncelleme kuralı şöyle ifade edilir (Bullnheimer ve diğ., 1999a; Stützle ve Dorigo, 1999b):

$$\tau_{ij}(t+1) := (1 - \rho) \cdot \tau_{ij}(t) + \sum_{r=1}^{w-1} (w-r) \cdot \Delta\tau_{ij}^r(t) + w \cdot \Delta\tau_{ij}^{gb}(t) \quad (3.19)$$

Burada, $\Delta\tau_{ij}^r(t) = 1/L^r(t)$ ve $\Delta\tau_{ij}^{gb}(t) = 1/L^{gb}$.

4. AKIŞ TİPİ ÇİZELGELEME PROBLEMİ

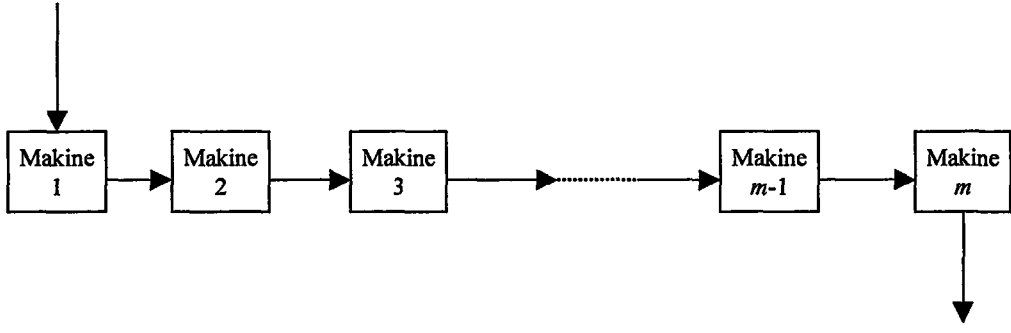
4.1 Basit Tanımlamalar

Akış tipi çizelgeleme problemleri, birçok endüstri alanında kullanıldığı için dikkate değer şekilde incelenmektedir. Akış tipi atölye, tek yönlü olan bir iş akışı ile tanımlanır. Yani bu sistemde, sisteme giren birimler ardışık olarak aynı sıradaki etkinlikler ile üretilirler. Bir yandan hammadde ve yarı ürün olarak alınan malzemeler, hattan işlemleri tamamlanmış veya ürün durumuna gelmiş olarak ayrılır. Akış tipi atölye, doğal bir makine sıralaması içerir: herhangi bir i işlemi j işleminden önce ise, i işlemini gerçekleştiren makinenin numarası j işlemini gerçekleştiren makinenin numarasından küçük olacak şekilde numaralandırmak mümkündür. Akış içindeki makineler, $1,2,...,m$ şeklinde numaralandırılırken, bunlara karşılık i işinin işlemleri $(i,1),(i,2),...,(i,m)$ şeklindedir. Şekil 4.1' de kuramsal bir akış tipi görülmektedir. Burada tüm işler her makinede tek işlemden geçmektedir. Şekil4.2' de çok daha genel bir akış tipi görülmektedir. Burada ise, işler m işlemden daha az işleme sahip olabilir. İşlerin işlemleri ise sıralı numaraları olan bitişik makinelerde işlenmeyebilir ve ilk ve son makineler sırasıyla 1 ve m makineleri olmayabilir. Akış tipi çizelgeleme problemleri aşağıdaki temel varsayımlara sahiptir (Baker, 1974):

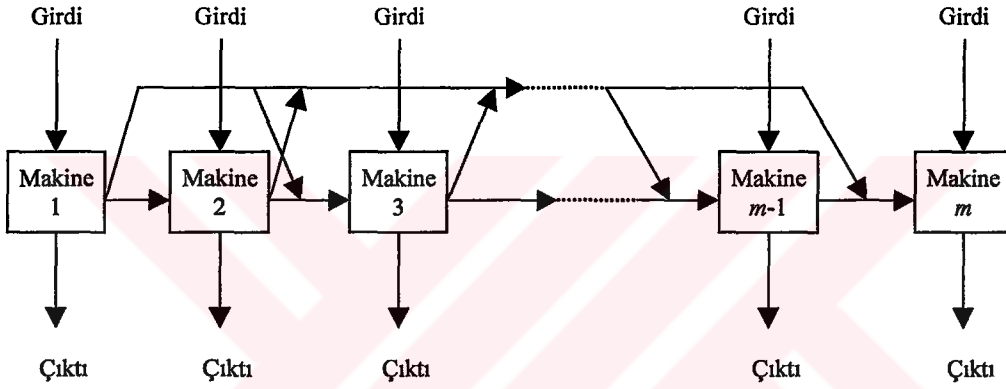
- n adet çok işli iş kümesi, işlenmek üzere sıfır anında hazırdır.
- İşlerin hazırlık süreleri sıraya bağlı değildir ve işlem sürelerine dahildir.
- İşlem süreleri önceden biliniyordur.
- m farklı makine sürekli çalışabilir.
- İşler kesilemez.

Akış tipi çizelgeleme problemi için literatürün büyük bir kısmı, her makinede işlerin aynı sırada işlendiği permütasyon akış tipi ile sınırlandırılmıştır. Permütasyon akış tipinde ilk makinedeki iş sıralaması sabitlenir ve kalan makinelerde de aynı sıralama kullanılır. Sonuç çizelgeye, permütasyon çizelge adı verilir (Blazewicz ve diğ., 1996).

Girdi (yeni işler)



Şekil 4.1 Kuramsal Akış Tipi Sistemi



Şekil 4.2 Genel Akış Tipi Sistemi

4.2 Amaçlar

Akış tipi çizelgeleme problemi, belirli bir amaca (kriter) göre n işlemin sıralamasını belirlemektir. En çok kullanılan amaç, işlerin tamamlanma sürelerini en küçükleyecek sıralamanın bulunmasıdır.

Şekil 4.3' de 4-işli 5-makineli bir akış tipi çizelgeleme problemi örneğinin Gantt Şeması verilmiştir.

İşlerin toplam tamamlanma zamanını (makespan) en küçükleyecek sıralamanın bulunması problemi ($n/m/P/C_{\max}$) aşağıdaki gibi tanımlanabilir:

Eğer j makinesindeki i işinin işlem süresi $t(i,j)$ ve permütasyon iş kümesi $\{\pi_1, \pi_2, \dots, \pi_n\}$ ise, tamamlanma zamanı $C(\pi_i, j)$:

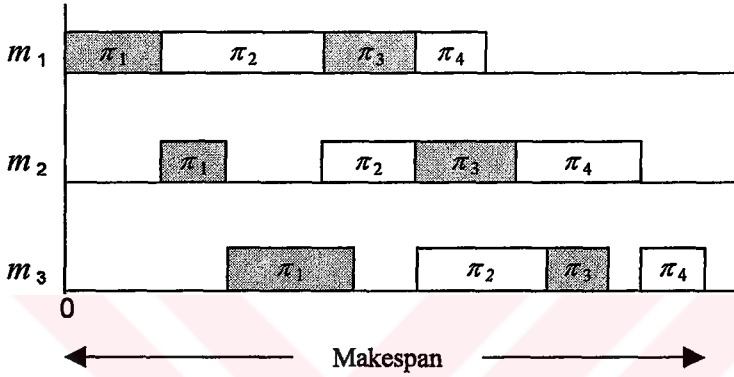
$$C(\pi_1, 1) = t(\pi_1, 1) \quad (4.1)$$

$$C(\pi_i, 1) = C(\pi_{i-1}, 1) + t(\pi_i, 1) \quad i=2, \dots, n \quad (4.2)$$

$$C(\pi_1, j) = C(\pi_1, j-1) + t(\pi_1, j) \quad j=2, \dots, m \quad (4.3)$$

$$C(\pi_i, j) = \max\{C(\pi_{i-1}, j), C(\pi_i, j-1)\} + t(\pi_i, j) \quad i=2, \dots, n; j=2, \dots, m \quad (4.4)$$

$$\text{Toplam tamamlanma zamanı: } C_{\max} = C(\pi_n, m) \quad (4.5)$$



Şekil 4.3 Akış Tipi Çizelgeleme Problemi için Gantt Şeması

Bugüne kadar akış tipi çizelgeleme problemi çalışmalarında kullanılan diğer kriterler: toplam akış zamanı (total flowtime, TF), ortalama akış zamanı (mean flowtime, $\bar{F}$), maksimum pozitif gecikme (maximum tardiness, $T_{\max}$), toplam pozitif gecikme (total tardiness, TT) ve toplam makine aylak kalma (idletime, IT) zamanıdır. $d(\pi_i)$, i işinin teslim tarihi olmak üzere bu amaçlar şöyle tanımlanabilir :

$$TF = \sum_{i=1}^n C(\pi_i, m) \quad (4.6)$$

$$\bar{F} = (1/n) \cdot \sum_{i=1}^n C(\pi_i, m) \quad (4.7)$$

$$T_{\max} = \max\{\max\{C(\pi_i, m) - d(\pi_i), 0\} | i = 1, \dots, n\} \quad (4.8)$$

$$TT = \sum_{i=1}^n \max\{C(\pi_i, m) - d(\pi_i), 0\} \quad (4.9)$$

$$IT = \left\{ C(\pi_1, j-1) + \sum_{i=2}^n \{\max\{C(\pi_i, j-1) - C(\pi_{i-1}, j), 0\}\} | j = 2, \dots, m \right\} \quad (4.10)$$

4.3 Matematiksel Formül

Akış tipi çizelgeleme problemini tanımlamak için temel bir model aşağıda gösterilmiştir. Kullanılan karar değişkenleri şunlardır (Blazewicz ve diğ., 1996):

$$z_{ij} = \begin{cases} 1 & \pi_i \text{ isi } j. \text{ pozisyona atanırsa} \\ 0 & \text{diger} \end{cases} \quad (4.11)$$

x_{jk} : iş permütasyonunda j pozisyonundaki iş başlamadan önce, P_k makinesindeki aylak zaman (bekleme zamanı)

y_{jk} : P_{k+1} makinesi serbest kalmak için beklerken, P_k makinesindeki iş bittikten sonra iş permütasyonunda j pozisyonundaki işin aylak zamanı

$C_{\max}$: işlerin toplam tamamlanma zamanı

Minimize $C_{\max}$

$$k.a. \quad \sum_{j=1}^n z_{ij} = 1, \quad i = 1, \dots, n \quad (4.12)$$

$$\sum_{i=1}^n z_{ij} = 1, \quad j = 1, \dots, n \quad (4.13)$$

$$\sum_{i=1}^n p_{ri} z_{i,j+1} + y_{j+1,r} + x_{j+1,r} = y_{j,r} + \sum_{i=1}^n p_{r+1,i} z_{ij} + x_{j+1,r+1}, \quad j = 1, \dots, n-1; r = 1, \dots, m-1 \quad (4.14)$$

$$\sum_{j=1}^n \sum_{i=1}^n p_{mi} z_{ij} + \sum_{j=1}^n x_{jm} = C_{\max}, \quad (4.15)$$

$$\sum_{r=1}^{k-1} \sum_{i=1}^n p_{ri} z_{i1} = x_{1k}, \quad k = 2, \dots, m \quad (4.16)$$

$$y_{1k} = 0, \quad k = 1, \dots, m-1 \quad (4.17)$$

Denklem(4.12) ve (4.13), işleri ve permütasyon pozisyonlarını karşılıklı atama yapar. Denklem(4.14), m -makinelili akış-tipindeki tüm bitişik makine çiftleri arasında Gantt Şeması hesabı yapar. Denklem(4.15), toplam tamamlanma zamanını belirler. Denklem(4.16), ilk işin gelişini beklerken ikinci ve takip eden makinelerin aylak kalma zamanını belirler. Denklem(4.17), permütasyondaki ilk işin ve hemen bir sonraki takip eden makineye geçişini sağlar.

4.4 Problemin Karmaşıklığı

Akış tipi çizelgeleme toplam tamamlanma zamanı en küçüklenmesi problemi, birleşik eniyileme problemi özelliğindedir ve NP-zor tipi problemler sınıfındadır. (Garey ve diğ., 1976; Gonzalez ve diğ., 1978). Sadece belli durumlar, etkin şekilde çözülebilir (Blazewicz ve diğ., 1996):

- İki-makinelili akış tipi durumu, kolaydır. Aynı şekilde, üç-makinelili durum ara makinenin işlem zamanı üzerine oldukça katı kısıtlar getirilerek polinomsal olarak çözülebilir.
- Johnson' ın iki-makinelili akış tipi çizelgeleme algoritması, durumu eğer ara makine darboğazda değilse, üç makinelili duruma da uygulanabilir.
- Grafik yöntemle iki-işli çizelgeleme yapılabilir.
- Johnson algoritması, işlerin kesilebildiği (preemptive) iki-makine çizelgeleme problemini ($F2|pmtn|C_{\max}$) çözer.
- Eğer, $\pi_i < \pi_j$ öncelik kısıtları tanımı, π_i işi, π_j işi o makine üzerinde işleme başlamadan önce her makinede işlemi tamamlanmayı gerektiriyorsa, ağaç veya seri-paralel öncelik kısıtları olan toplam tamamlanma zamanı en küçüklenmesi için iki-makinelili akış tipi çizelgeleme problemi polinomsal zamanda çözülebilir.

4.5 Akış Tipi Çizelgeleme Problemi Çözüm Algoritmaları

Akış tipi çizelgeleme problemleri ile ilgili ilk çalışma Johnson (1954) tarafından yapılmıştır. Johnson, n -işli ve 2-makinelili akış tipi çizelgeleme problemleri üzerinde toplam tamamlanma zamanını en küçüklemek için bir eniyileme algoritması tanımlamıştır. Ancak, çok sayıda iş ve makine içeren akış tipi çizelgeleme

problemleri birleşik eniyileme problemi özelliğindedir ve NP-zor tipi problemler sınıfındadır. Birleşik eniyileme problemlerinin büyük bir kısmı, son zamanlarda mevcut bilgisayarların CPU gücü hızla yükseldiği için, iterasyona dayanan yaklaşımlar yardımı ile çözülebilmektedir. Başka bir deyişle, bu tip problemler için yaklaşık çözümler elde edilmektedir.

4.5.1 Kesin Çözüm Algoritmaları

4.5.1.1 Johnson Algoritması

İki makineden geçecek n tane işin, aynı teknolojik sırayı izleyerek işlenmesi gerektiğini varsayalım. Bu tip seri iş akışı bulunan $(2 \times n)$ sıralama problemlerinde oluşan en iyi iş sırasının şu ölçütleri eniyilemesi istenir:

- Bitiş zamanlarının en küçüklenmesi
- İşlerin ortalama bekleme süresinin en küçüklenmesi
- Makinelerin ortalama boş kalma zamanının en küçüklenmesi

Bu istekleri gerçekleştiren iş sırasının bulunması, Johnson algoritması ile gerçekleştirilir.

İki-makineli toplam tamamlanma zamanı en küçüklenmesini amaçlayan akış tipi problemi için bu yöntem Şekil 4.4' de verilmiştir (Baker, 1974; Johnson, 1954).

t_{ij} : i . işin j . makinedeki işlem süresi olmak üzere;

Adım1: İlk makinede ikinci makineden daha kısa işlem süresine sahip işlerin kümesi U oluşturulur. $U = \{j | t_{j1} < t_{j2}\}$. İkinci makinede daha kısa işlem süresine sahip işlerin kümesi V oluşturulur. $V = \{j | t_{j2} \leq t_{j1}\}$.

Adım2: İlk makineye en kısa işlem süresi kuralına göre (Shortest Processing Time-SPT) düzenlenen U kümesindeki işleri çizelgele. İkinci makineye en uzun işlem süresi kuralına göre (Longest Processing Time-LPT) düzenlenen V kümesindeki işleri çizelgele.

Adım3: Optimal sıralama, U kümesi işlerini takip eden V kümesi işleri toplamıdır.

Şekil 4.4 Johnson Algoritması

4.5.1.2 Revize Johnson Algoritması

$m = 3$ için toplam tamamlanma zamanı en küçüklenmesini sağlayan iş sırasını saptamak üzere geliştirilmiş yalın bir teknik yoktur. Salt bazı özel durumlar için problemin en iyi çözümünü bulmak olanaklıdır. Belirli koşullar sağlandığı takdirde Johnson yönteminin geliştirilmiş şekli ile en kısa zamanda işlerin bitirilmesini sağlayacak iş sırasının bulunması kolaydır. Bu koşullar sağlanmadığı takdirde, analitik yöntemlerle işlerin en kısa zamanda bitirilmesini sağlayacak iş sırasının bulunması zor ve zaman alıcı olabilir. Bunlardan biri Johnson algoritmasının revize edilmiş durumudur. Burada, ikinci makine (yada üçüncü makine), ilk makineden daha kısa işlem süresine sahip olduğu durumlar için kesin çözümler mümkündür. Yani; üç-makinelili toplam tamamlanma zamanı en küçüklenmesini amaçlayan akış tipi problemi için; eğer $\min_j \{t_{j1}\} \geq \max_j \{t_{j2}\}$ yada $\min_j \{t_{j3}\} \geq \max_j \{t_{j2}\}$ ise problem, Johnson algoritması uygulanarak iki makineli durum gibi çözülebilir. Kısaca yöntemin uygulanabilmesi için ikinci makinenin, tümüyle birinci veya üçüncü makinenin yönetimi altında olması gerekir. İki koşuldan biri sağlandığı zaman iki yapay makine tanımlanır. Bu makinelerdeki işlem süreleri şöyledir:

$$T_{i1} = t_{i1} + t_{i2} \quad (4.18)$$

$$T_{i2} = t_{i2} + t_{i3} \quad (4.19)$$

Bu tanımlamadan sonra bu iki yapay makine kullanılarak Johnson Yöntemi veya revize Johnson algoritmasıyla sıralama yapılır (Morton ve Pentico, 1993).

4.5.1.3 Dal-Sınır Algoritması (Ignall-Schrage Algoritması)

m -makinelili toplam tamamlanma zamanı en küçüklenmesini amaçlayan akış tipi çizelgeleme problemi için Ignall ve Schrage (1965) tarafından dal-sınır algoritması geliştirilmiştir. Problem bir ağaç yapısı ile tanımlanır ve her düğüm noktası bir kısmi çözüm demektir. Ağaçtaki ilk düğüm, hiçbir işin çizelgelenmediği başlangıç durumunu gösterir. Bu düğümden çıkan n adet dal çizelgede yerleştirilecek ilk n işi gösterir (Ignall ve Schrage, 1965; Baker, 1974).

Burada $m = 3$ için sınır değerleri gösterilecektir. σ , sıralamanın başından sonuna doğru kısmi sıralamayı ve σ' , kısmi sıralamada bulunmayan işleri gösterir.

σ sıralaması için;

q_1 : σ kümesindeki işler arasında 1. makinede en geç tamamlanan iş.

q_2 : σ kümesindeki işler arasında 2. makinede en geç tamamlanan iş.

q_3 : σ kümesindeki işler arasında 3. makinede en geç tamamlanan iş.

Birinci makinede gerekli işlem süresi şöyle belirlenir:

$$\sum_{j \in \sigma'} t_{j1} \quad (4.20)$$

Bundan başka, birinci makinede, en son olan belli bir son i işi (bilinmeyen) olsun. Bu tamamlandıktan sonra, i işi, en az $(t_{i2} + t_{i3})$ kadar 2 ve 3 makinesinde işlem görmelidir. Bu son iş:

- Bir işlemin tamamlanması ile bir sonraki makinede işleme başlaması arasında gecikme yoktur.
- Son i işi işlemleri arasında gecikme yoktur.
- σ' kümesindeki işlerin arasında, i , minimum $(t_{i2} + t_{i3})$ toplama sahiptir.

İlk makinede toplam tamamlanma zamanı için alt-sınır:

$$b_1 = q_1 + \sum_{j \in \sigma'} t_{j1} + \min_{j \in \sigma'} \{t_{j2} + t_{j3}\} \quad (4.21)$$

İkinci makinede toplam tamamlanma zamanı için alt-sınır:

$$b_2 = q_2 + \sum_{j \in \sigma'} t_{j2} + \min_{j \in \sigma'} \{t_{j3}\} \quad (4.22)$$

Üçüncü makinede toplam tamamlanma zamanı için alt-sınır:

$$b_3 = q_3 + \sum_{j \in \sigma'} t_{j3} \quad (4.23)$$

Bu hesaplamayı yaptığımızda, Ignall-Schrage tarafından tercih edilen alt-sınır:

$$B = \max\{b_1, b_2, b_3\} \quad (4.24)$$

4.5.2 Yaklaşık Çözüm Algoritmaları

4.5.2.1 Palmer Sezgiseli

m -makinelili toplam tamamlanma zamanı en küçüklenmesini amaçlayan akış tipi problemi için, Palmer (1965) bir eğim indeksi (slope index) s tanımlamıştır:

$$s_j = -(m-1)t_{j1} - (m-3)t_{j2} - (m-5)t_{j3} + \dots + (m-3)t_{j,m-1} + (m-1)t_{jm} \quad (4.25)$$

Daha sonra aşağıdaki iş permütasyon çizelgesi oluşturulur:

$$s_{[1]} \geq s_{[2]} \geq \dots \geq s_{[n]} \quad (4.26)$$

4.5.2.2 CDS Sezgiseli

CDS sezgiseli, Campbell ve diğ. (1970) tarafından, $(3 \times n)$ sıralama problemlerinde olduğu gibi iş sürelerinde toplamlar elde ederek Johnson yönteminin kullanımını sağlamak üzere önerilmiştir. Bu yaklaşımda, m makinelili bir sistemde $(m-1)$ adet grupta 2 kümeli iş süreleri elde edilmekte ve bu kümelerin her birine Johnson yöntemi uygulanmaktadır. Bu çözümlemede n tane işin makineden aynı sırada geçtikleri varsayılmaktadır.

$$\text{İlk } i \text{ makinesindeki işlerin toplamı } T_{j1} = \sum_{k=1,i} t_{jk} \quad (4.27)$$

$$\text{Son } i \text{ makinesindeki işlerin toplamı } T_{j2} = \sum_{k=1,i} t_{j,m-k+1} \quad (4.28)$$

Kümelerin her birine Johnson yöntemi uygulanarak elde edilen toplam tamamlanma zamanı M_i biriktirilir ve $M = \min\{M_i\}$ olacak şekildeki sıralama seçilir. CDS algoritmasının en büyük yararı, en iyi (veya en iyiye yakın) iş sırasının belirlenmesinde, olası tüm sıraların araştırılmasının önlenmesidir (Morton ve Pentico, 1993).

4.5.2.3 Gupta Sezgiseli

m -makinelili toplam tamamlanma zamanı en küçüklenmesini amaçlayan akış tipi problemi için Gupta (1972) tarafından iyi çizelgeler oluşturacak şekilde, Johnson algoritmasına dayanan bir geçişli iş sıralama kuralı önermiştir (Baker, 1974).

Bir i ($i = 1, 2, \dots, n$) işi için iş indeksi s_i şöyle tanımlanır:

$$s_i = \frac{e_i}{\min_{1 \leq k \leq m-1} \{t_{ik} + t_{i,k+1}\}} \quad (4.29)$$

burada,

$$e_i = \begin{cases} 1 & \text{eger } t_{i1} < t_{im} \\ -1 & \text{eger } t_{i1} \geq t_{im} \end{cases} \quad (4.30)$$

Daha sonra aşağıdaki iş sıralaması oluşturulur:

$$s_{[1]} \geq s_{[2]} \geq \dots \geq s_{[n]} \quad (4.31)$$

4.5.2.4 NEH Sezgiseli

NEH sezgiseli, Nawaz ve diğ. (1983) tarafından, n -iş m -makinelikli toplam tamamlanma zamanı en küçüklenmesini amaçlayan akış tipi çizelgeleme problemine çözüm sağlamak üzere önerilmiştir. Önerilen algoritma, tüm makineler üzerinde daha fazla işlem süresine sahip bir işe, daha az işlem süresine sahip bir işten daha yüksek bir öncelik verilmesi varsayımına dayanır. Buna göre, n -iş arasından en yüksek işlem süresine sahip iki iş seçilir. Bu iki iş için en iyi kısmi sıralama belirlenir. Bu iki işin birbirlerine göre pozisyonları sabit tutulur. Daha sonra, üçüncü en yüksek işlem süresine sahip iş seçilir. Bu iş bir önceki adımda bulunan kısmi sıralamanın başına, ortasına ve sonuna yerleştirilerek oluşturulan üç kısmi sıralama değerlendirilir. En iyi kısmi sıralama, bu üç işin birbirlerine göre pozisyonlarını sabit tutar. Süreç tüm işler çizelgeleme de devam ettirilir.

4.5.3 Diğer Algoritmalar

Akış tipi çizelgeleme problemi ile ilgili çalışmalar halen devam etmektedir. Bu çalışmalarda, çoğunlukla yeni önerilen sezgisel ve metasezgisel yaklaşımlar kullanıldığı gözlenmektedir. Tablo 4.1' de literatürde bugüne kadar toplam tamamlanma zamanı amacı için tek amaçlı akış tipi çizelgeleme problemleri ile ilgili yapılmış çalışmaların bir özeti görülmektedir. Tablo 4.2' de toplam tamamlanma zamanı amacından farklı amaçlar için yapılmış çalışmaların bir özeti görülmektedir.

Ayrıca Ek A' da bu çalışmaların ayrıntılı özeti verilmiştir.

Tablo 4.1 Toplam Tamamlanma Zamanı Amacı için Tek-amaçlı Akış Tipi Çizelgeleme Çalışmaları

Ek Kısıt	Çözüm Yaklaşımı	Referans
$n \times m$ permütasyon akış tipi çizelgeleme problemi		
Yok	Kesin çözümler (dalsınır, eliminizasyon kuralı)	Johnson, 1954; Ignall ve Schrage, 1965, McMahon ve Burton, 1967; Ashour, 1970; Szwarc, 1973; Baker, 1975.
Yok	Sezgisel	Palmer, 1965; Smith ve Dubek, 1967; Gupta, 1969; Dannenbring, 1977; Nawaz ve diğ., 1983; Hundal ve Rajgopal, 1988; Widmer ve Hertz, 1989; Werner, 1993; Lai, 1996; Loirenço, 1996; Nagano ve Moccellini, 2002.
Yok	Yasak arama	Taillard, 1990; Nowichi ve Smutnicki, 1996.
Yok	Tavlama benzetimi	Ogbu ve Smith, 1991; Ishibuchi, 1995; Zegordi ve diğ., 1995
Yok	Genetik algoritma	Stöppler ve Bierwirth, 1992; Mulken, 1994; Chen ve diğ., 1995; Reeves, 1995; Yamada ve Reeves, 1997, Cotta ve Troya, 1998; Reeves ve Yamada, 1998; Chan ve Hu, 2001, 2002; Iyer ve Saxena, 2004.
Yok	Karınca kolonileri eniyileme	Stüttzle, 1998a; Ying ve Liao, 2004; Rajendran ve Ziegler, 2004.
Yok	Sinirsel ağlar	Lee ve Shaw, 2000.
Yok	Melez yaklaşım	Murata, Ishibuchi ve Tanaka, 1996; Wang ve Zheng, 2003.
Yok	Ardaşık yerel arama	Stüttzle, 1998b.
Aşamalar arası stoklama olan problemler	Sezgisel	Cao ve Bedworth, 1992.
Ara stoklama bulunmayan problemler	Sezgisel	Thornton ve Hunsucker, 2004.
Kullanılabilirlik kısıtı bulunan problemler	Sezgisel, dinamik programlama, genetik algoritma, yasak arama	Lee, 1997; Cheng ve Wang, 2000; Aggoune, 2001.
Lineer zamanlı işlem bulunan problemler	Genetik algoritma	Janiak ve Portmann, 1998.
Zaman boşluğu bulunan çok işlemli problemler	Sezgisel	Riezebos ve diğ., 1995; Riezebos ve Gaalman, 1998.

Tablo 4.1 Toplam Tamamlanma Zamanı Amacı için Tek-amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Ek Kısıt	Çözüm Yaklaşımı	Referans
$n \times m$ permütasyon akış tipi çizelgeleme problemi		
Bloklanan akış türü problemi	Genetik algoritma	Caraffa ve diğ., 2001.
Her aşamada çok işlemci	Dal-sınır	Brah ve Hunsucker, 1991.
İş atlamasız (no job passing) ve iş beklemesiz (no job waiting)	Sezgisel	King ve Spachis, 1980.
Kaynak kısıtlı akış türü problemi	Genetik algoritma	Leu ve Hwang, 2002.
Sürekli akış türü problemi	Genetik algoritma, yasak arama, tavlama benzetimi	Cleveland ve Smith, 1989; Fink ve Voß, 2003.
Bekleme olmayan melez akış tipi problemleri	Sezgisel	Zhixin ve diğ., 2002.
Sadece melez akış türü problemleri	Dal-sınır, genetik algoritma, sezgisel, yasak arama, tavlama benzetimi	Portmann ve diğ., 1998; Oğuz ve diğ., 2001; Oğuz ve diğ., 2004.
Süreç içi beklemeye izin verilmeyen (no-wait) problemler	Sezgisel	Wisner, 1972; Gangadharan ve Rajendran, 1993; Rajendran, 1994; Aldowaisan ve Allahverdi, 2003.
Bekleme olmayan yığın akışlı problemler	Sezgisel, genetik algoritma	Kumar ve diğ., 2000.
$n \times m$ permütasyon olmayan akış tipi çizelgeleme problemi		
Yok	Sezgisel	Koulamas, 1998.

Tablo 4.2 Farklı Amaçlar için Tek-amaçlı Akış Tipi Çizelgeleme Çalışmaları

Amaç	Ek Kısıt	Çözüm Yaklaşımı	Referans
Tamamlanma zamanı teslim zamanından ortalama ağırlıklı sapması	Sadece yığın akışlı problemler	Melez genetik algoritma	Yoon ve Ventura, 2002.
Bitiş zamanı	Esnek akış tipi	Sezgisel	Sriskandarajah ve Sethi, 1989.
Erken ve gecikme maliyeti	Sektör çizelgeleme	Genetik algoritma	Cartwright ve Tuson, 1994
Kompleks maliyet fonksiyonu	Yok	Sezgisel	Gelders ve Samdandam, 1978.
Toplam akış zamanı	Yok	Sezgisel	Rajendran ve Chaudhuri, 1991; Rajendran, 1993; Ho, 1995; Woo ve Yim, 1998; Tang ve Liu, 2002; Framinan ve diğ., 2004
Tamamlanma zamanları toplamı	Yok	Dal-sınır, sezgisel	Ahmadi ve Bargchi, 1990; Wang ve diğ., 1997.
Toplam bitiş zamanı	Yok	Genetik algoritma, yerel arama	Yamada ve Reeves, 1998.
Toplam pozitif gecikme	Yok	Genetik algoritma	Kim, 1995; Onwubolu ve Mutingi, 1999; Yong ve Sannomiya, 2001.
Toplam ağırlıklı pozitif gecikme	Yok	Genetik algoritma	Neppalli ve diğ., 1994.
Sıralı (ordered) akış tipi çizelgeleme problemi			
Ortalama tamamlanma zamanı	Yok	Sezgisel	Panwalkar ve Khan, 1976.

4.6 İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Problemi

Akış tipi çizelgeleme problemlerinde toplam tamamlanma zamanı, toplam akış zamanı, ortalama akış zamanı, maksimum pozitif gecikme, toplam pozitif gecikme tekil olarak gözönüne alınan kriterler arasındadır. Ancak gerçek hayatta karşılaşılan birçok problem çok amaçlıdır. Bu durumda, çok amaçlı bir eniyileme probleminin tüm baskın çözümlerini bulunmaya çalışılır. n amacı olan çok amaçlı bir problem aşağıdaki gibi ifade edilebilir (Ishibuchi ve Murata, 1998):

Maximize $f_1(x), f_2(x), \dots, f_n(x)$

Burada x , belirlenen bir vektör, $f_1(\cdot), f_2(\cdot), \dots, f_n(\cdot)$, en büyüklenen n amaç fonksiyonudur. Eğer bir uygun çözüm, çok amaçlı eniyileme probleminin diğer uygun çözümlerinden baskınsa bu çözüm, baskın çözümdür (non-dominated solution). Aşağıdaki eşitsizlikler iki çözüm x ve y arasında oluşturulduğunda, x çözümünü y çözümünden baskındır.

$$\forall_i : f_i(x) \leq f_i(y) \quad \text{ve} \quad \exists_j : f_j(x) < f_j(y)$$

Çok amaçlı eniyileme algoritmalarında ana amaç, amaçların tek tek son çözümlerini bulmak değil, çok amaçlı eniyileme probleminin tüm baskın çözümlerini bulmaktır. Çok amaçlı eniyileme problemi için karar vericiden etkileşimli olarak bilgi alınmaksızın tek bir çözüm belirlemek zor olduğu için, genel yaklaşım karar vericiye baskın çözümler kümesini sunmaktır. Daha sonra baskın çözümlerden biri karar vericinin tercihinine göre belirlenebilir.

4.7 İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Problemi Çözüm Algoritmaları

Çok-amaçlı akış tipi çizelgeleme problemi içinde literatürdeki mevcut çok amaçlı eniyileme algoritmalarından faydalanmak mümkündür. Bunun yanında problemin yapısına özgü geliştirilmiş yaklaşımlarda kullanılabilir. Tablo 4.3' de literatürde bugüne kadar çok-amaçlı akış tipi çizelgeleme problemleri ile ilgili yapılmış çalışmaların bir özeti görülmektedir.

Ayrıca Ek B' de bu çalışmaların ayrıntılı özeti verilmiştir.

Tablo 4.3 İki-ve Çok-Amaçlı Akış Tipi Çizelgeleme Çalışmaları

Problem Tipi	Amaç	Çözüm Yaklaşımı	Referans
$n \times 2$, 2-amaçlı permütasyon	Toplam tamamlanma zamanı ve maksimum pozitif gecikme zamanı	Sezgisel	Daniels ve Chambers, 1990.
	Toplam tamamlanma zamanı ve ortalama akış zamanı	Dal-sınır, genetik algoritma, sezgisel	Nagar ve diğ., 1996; Sivrikaya-Serifoğlu ve Ulusoy, 1998; Yeh, 1999, 2002; Allahverdi, 2003.
	Toplam tamamlanma zamanı ve toplam akış zamanı	Genetik algoritma, yasak arama, karınca kolonileri, sezgisel	Rajendran, 1992; Neppalli ve diğ., 1996; Gupta ve diğ., 1999; T'Kindt ve diğ., 2000; Gupta ve diğ., 2001.
	Erken maliyeti ve gecikme maliyeti	Genetik algoritma	Chen ve Smith, 1999.
	Toplam tamamlanma zamanı ve toplam bitiş zamanı	Dal-sınır, karınca kolonileri, sezgisel	Sayın ve Karabatı, 1999; T'Kindt ve diğ., 2002; T'Kindt ve diğ., 2003.
	Toplam tamamlanma zamanı ve toplam akış zamanı/toplam ağırlıklı akış zamanı/toplam ağırlıklı gecikme	Tavlama benzetim, threshold accepting, çok-aşamalı arama, genetik algoritma	Gupta, Hennig, ve Werner, 2002.
$n \times 2$, çok-amaçlı permütasyon	Toplam tamamlanma zamanı, A işlemcisinin hazırlık zamanı ve işlerin değişim oranı	Sezgisel	Tamura ve diğ., 1999.
$n \times m$, 2-amaçlı permütasyon	Toplam tamamlanma zamanı ve toplam akış zamanı	Tamsayılı programlama	Wilson, 1989.
	Toplam tamamlanma zamanı, toplam akış zamanı maksimum aylak kalma ve toplam aylak kalma	Genetik algoritma	Murata, Ishibuchi ve Lee, 1996.
	Toplam tamamlanma zamanı ve ortalama pozitif gecikme zamanı (tardiness)	Genetik algoritma	Cavalieri ve Gaiardelli, 1998.
	Toplam tamamlanma zamanı ve maksimum gecikme zamanı (lateness)	Genetik algoritma, sezgisel	Tagami ve Kawabe, 1998, 1999.; Allahverdi ve Aldowaisan, 2004.

Tablo 4.3 İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Problem Tipi	Amaç	Çözüm Yaklaşımı	Referans
$n \times m$, 2-amaçlı permütasyon	Toplam tamamlanma zamanı ve maksimum pozitif gecikme zamanı	Sezgisel , tavlama benzetimi	Chakravarty ve Rajendran, 1999; Allahverdi, 2004.
	Toplam tamamlanma zamanı ve toplam pozitif gecikme zamanı	Genetik algoritma	Mabed ve diğ., 2000; Talbi, Rahoual ve diğ., 2001; Basseur ve diğ., 2002.
	Toplam tamamlanma zamanı ve ortalama akış zamanı	Sezgisel	Allahverdi, 2003.
	Erken ve gecikme cezaları	Sezgisel	Gupta, Kruger, Lauff, Werner ve Sotskov, 2002.
$n \times m$, 2-amaçlı permütasyon olmayan	Toplam tamamlanma zamanı ve toplam akış zamanı	Sezgisel	Pugazhendhi ve diğ., 2002.
$n \times m$, çok-amaçlı permütasyon	Toplam tamamlanma zamanı, toplam akış zamanı ve makine aylak zamanı	Sezgisel, genetik algoritma	Ho ve Chang, 1991; Rajendran, 1994; Rajendran, 1995; Sridhar ve Rajendran, 1996.
	Toplam tamamlanma zamanı, maksimum pozitif gecikme ve toplam akış zamanı	Genetik algoritma	Murata ve Ishibuchi, 1997; Ishibuchi ve Murata, 1998; Ishibuchi ve diğ., 2003.
	Toplam tamamlanma zamanı, toplam akış zamanı ve toplam pozitif gecikme	Genetik algoritma	Ishibashi ve diğ., 2000.
	Toplam tamamlanma zamanı, ortalama akış zamanı ve ortalama pozitif gecikme	Genetik algoritma	Zhao ve diğ., 2001.
	Toplam tamamlanma zamanı, toplam akış zamanı, toplam pozitif gecikme ve maksimum pozitif gecikme	Genetik algoritma	Chang ve diğ., 2002.

5. AKIŞ TİPİ ÇİZELGELEME PROBLEMİ İÇİN KKE YAKLAŞIMI

5.1 Problemin Tanımı ve Çözüm Yöntemi

Ele alınan permütasyon akış tipi çizelgeleme problemi, tüm makinelerde bir işin işlem sırasının aynı olduğu, m makine ($j = 1, 2, \dots, m$) üzerinde belli işlem sürelerine sahip n işin ($i = 1, 2, \dots, n$) çizelgelenmesinden oluşur. Problem, Bölüm 4.1' de açıklanan akış tipi çizelgeleme problemi temel varsayımlarına sahiptir. Akış tipi çizelgeleme problemi, Johnson (1954) tarafından toplam tamamlanma zamanını en küçükleyen iki-aşamalı çizelgeleme problemi önerildiğinden buyana yaklaşık 50 yıldır geniş bir araştırma alanına sahiptir. Bu konuda çok sayıda çalışma yapılmış ve çözümü için kesin ve yaklaşık çözümler önerilmiştir. Ancak bu çalışmaların büyük bir çoğunluğu tek bir kriter üzerine yoğunlaşmaktadır. Toplam tamamlanma zamanı kriterinden başka, akış tipi çizelgeleme de önemli bir amaç, toplam akış zamanı veya işlerin tamamlanma zamanı toplamıdır. Toplam tamamlanma zamanı en küçüklenmesi, üretim hızını en küçüklerken, akış zamanı en küçüklenmesi, kaynakların daha düzgün kullanımını, işlerin hızlı dönüşümünü ve süreç içi envanteri en küçükler. Hatta, akış zamanı en küçükleme için çizelgeleme, toplam tamamlanma zamanı en küçükleme çizelgelemenin oluşturduğundan daha fazla toplam çizelgeleme maliyeti en küçüklenmesi sağlayabilir. Akış tipi çizelgeleme problemleri literatürüne bakıldığında birden fazla kriteri dikkate alan çalışmaların sayısı çok azdır. Bu amaçla bu çalışmada birden fazla amacı gözönüne alacak akış tipi çizelgeleme problemi ele alınacaktır. Problem için dikkate alınacak amaçlar sırasıyla, toplam tamamlanma zamanı, toplam akış zamanı ve toplam makine aylak kalma zamanıdır.

Bu probleme çözüm oluşturmak için, ilk çalışma Ho ve Chang (1991) tarafından yapılmıştır. Bu çalışmada toplam tamamlanma zamanı kriteri ile birlikte toplam akış zamanı ve toplam makine aylak kalma zamanı da dikkate alınmıştır. Önerilen algoritmada, çizelgeler mevcut işlemler arası zaman boşluğu (time lag) en küçüklendirilerek güçlendirilir. Ho ve Chang sezgiseli (HC), CDS sezgiseli tarafından oluşturulan başlangıç çözümünü güçlendirmektedir. Önerilen algoritma,

üç kriter için, mevcut diğer sezgisellerden (Palmer, CDS, Gupta, Dannenbring ve Hundal) daha iyi sonuçlar vermiştir. Bu konuda daha sonraki çalışma Rajendran (1994a, 1995) tarafından yapılmıştır. Burada, yine toplam tamamlanma zamanı, toplam akış zamanı ve toplam makine aylak kalma zamanı dikkate alınarak, bu amaçları en küçüklendirecek yeni bir sezgisel yaklaşım önerilmiş ve algoritmanın sonuçları HC sezgiseli tarafından üretilen sonuçlarla karşılaştırılmıştır. Yapılan bu çok amaçlı çalışmalarda, algoritma karşılaştırılan sezgiselden daha etkindir. Son olarak bu konuda Sridhar ve Rajendran (1996) tarafından bir çalışma yapılmış ve aynı amaçlar için genetik algoritmaya dayanan bir sezgisel geliştirilmiştir. Algoritmanın performansı, HC sezgiseli ile karşılaştırılmıştır. Önerilen algoritmanın performansı, HC sezgiselinden daha iyi sonuçlar vermiştir.

Bu problemin çözümü için son zamanlarda özellikle toplam tamamlanma zamanı/toplam akış zamanı kriterli permütasyon akış tipi üretim için çok iyi sonuçlar veren karınca kolonileri yaklaşımından faydalanılacaktır. Bu yöntemin seçilmesinin bir nedeni karınca kolonileri ile yapılan çok amaçlı problem sayısının çok az olması ve akış tipi problem için ise herhangi bir çalışmanın olmamasıdır. Bu amaçla bu yöntemin çok amaçlı problemler üzerindeki performansı saptanmaya çalışılacaktır. Ayrıca, akış tipi çizelgeleme problemi için KKE yaklaşımı uygulanarak daha önce yapılan çalışmalardan daha iyi bir çözüm elde edilip edilemeyeceği araştırılacaktır.

5.2 Literatür Araştırması

Son yıllarda, literatürde birleşik eniyileme problemlerini çözmek için KKE algoritmalarının kullanılması yönünde çalışmalar yapılmaktadır. Akış tipi çizelgeleme problemlerini çözmek için de çok az sayıda da olsa KKE algoritması önerilmiştir. Bu konudaki ilk çalışma Stützle (1998a) tarafından yapılmıştır. Bu çalışmada, permütasyon akış tipi problemi üzerinde durulmuştur. Amaç, son işin tamamlanma zamanının $C_{\max}$ (makespan) en küçüklenmesidir. Çözüm algoritması olarak yerel aramanın yapıldığı max-min karınca sistemi kullanılmıştır. *MMAS* de bir işin pozisyonu çözüm özelliği olarak kullanılır. Bir sıralamanın yapısında, karıncaların başlangıç olarak bulunacağı bir kukla "0" işi tanımlanır. Karıncalar, ilk pozisyona bir işi, ikinci pozisyona bir başka işi atayarak ve tüm işlerin ataması tamamlanana dek bu çizelgelemeyi devam ettirirler. Karıncalar, belli bir pozisyona işleri seçmek için, en iyi mümkün kararı bir p_0 olasılığına göre verirler ve belli bir

sıralamada kullanılan işlerin izini takip etmek için, kısmi sıralamaları depolayarak, bir yasaklar listesi oluştururlar. Tam bir sıralama oluşturulduktan ve yerel arama ile iyileştirildikten sonra, izler güncellenir. *MMAS* algoritmasında, sadece bir karıncanın, izleri güncelleştirmesine izin verilir. İşlerin yerleştirildiği pozisyonlar, çok miktarda pheromone izi taşır ve işler, çizelgeleme aşamasında, yüksek pheromone izine sahip bu pozisyonlar üzerine yerleştirilir. Yeni sonuçların araştırılması ile mevcut bilginin devamlılığının sağlanması arasında bir denge sağlamak için, *MMAS* algoritmasında, pheromone izi için belli sınırlar uygulanır. Ayrıca her karınca bir çözüm oluşturduktan sonra, bir yerel arama yöntemi uygulanarak çözüm iyileştirilmektedir. Sonuçlar, önerilen yaklaşımın karşılaştırılan diğer mevcut sezgisellerden (NEH, NEH+yerel arama, tavlama benzetimi, katlı iniş (Multiple Descent-MD)) daha iyi olduğunu göstermektedir.

Akış tipi çizelgeleme problemlerinin KKE algoritmaları ile çözümü üzerine diğer bir çalışma ise Ying ve Liao (2004) tarafından yapılmıştır. Bu çalışmada, $n/m/P/C_{\max}$ problemi için, bir ACS algoritması önerilmiştir. Bu algoritmada ilk olarak, yapay karıncalar kümesi, bir başlangıç kuralına göre ilk düğümlere yerleştirilir. Her karınca bir geçiş kuralı kullanarak bir tur oluşturur. Daha sonra, karınca tur oluşturulurken, bir yerel güncelleme kuralı uygulayarak geçtiği düğümler üzerindeki pheromone miktarını günceller. Son olarak tüm karıncalar turlarını tamamlar tamamlamaz, düğümler üzerindeki pheromone miktarını bir global güncelleme kuralı ile tekrar değiştirir. Karıncalar, uygun çözüm oluştururken hareket etmek için seçeceği bir sonraki düğümü sanki-rasgele-oran geçiş kuralı ile belirler. Seçilen düğüm, yasaklar listesine eklenir. Sonuçta, yasaklar listesindeki düğüm permütasyonu, iş sıralamasını oluşturur. Her karınca bir çözüm oluşturduktan sonra, API yöntemini kullanarak çözümü iyileştirir. Önerilen algoritmanın bulduğu çözümlerin, genetik algoritmalar, tavlama benzetimi ve bir yerel komşuluk arama (Neighbourhood Search-NS) algoritmaları ile bulunan çözümlerden daha iyi olduğu görülmüştür.

Son zamanlarda permütasyon akış tipi çizelgeleme problemlerinin KKE algoritmaları ile çözümü üzerine diğer bir çalışma ise Rajendran ve Ziegler (2004) tarafından yapılmıştır. Bu çalışmada, $n/m/P/(C_{\max}/\sum C_i)$ problemi ele alınmıştır. Bu problemi çözmek için iki karınca kolonileri algoritması önerilmiştir. İlk algoritma (M-MMAS), Stützle (1998a) tarafından önerilen max-min karınca algoritması ile Merkle ve Middendorf (2000) tarafından geliştirilen toplama kuralı ve önerilen yeni iş-

indeksine dayanan bir yerel arama tekniği (job-index-based local search procedure) ile birleştirilerek elde edilmiştir. İkinci algoritma (PACO) ise, geliştirilmiş bu yeni yerel arama tekniğine dayanmaktadır. Önerilen algoritmalar, toplam tamamlanma zamanı en küçüklendirmesine göre test problemleri üzerinde denenmiş ve her iki algoritma sonucunun, *MMAS* algoritmasına göre daha iyi performansa sahip olduğu görülmüştür. Daha sonra, önerilen algoritmalar toplam akış zamanı gözönüne alınarak, daha önce bu problem için önerilen en iyi sezgiselin sonuçları ile karşılaştırılmıştır. Karşılaştırmalar, önerilen karınca koloni algoritmalarının daha iyi sonuçlar verdiğini göstermiştir.

KKE algoritmaları yardımıyla da iki-amaçlı akış tipi çizelgeleme problemi için çözüm bulunmaya çalışılmıştır. İki-amaçlı akış tipi problemi için T'kindt ve diğ. (2000, 2002) tarafından ile yapılan çalışmada, 2-makineli ve işlemlerin yarıda kesilemediği durum ele alınmıştır. Toplam tamamlanma zamanı ve toplam bitiş zamanı kriterleri gözönüne alınır. Bu $F2||Lex(C_{max}, \sum C_i)$ probleminin çözümü için, SACO olarak adlandırılan bir KKE algoritması önerilmiştir. Algoritmada, her karınca bir geliştirici süreç kullanarak uygun bir çözüm oluşturur. Bu süreç, karıncaların kullandığı ve uygun çözümün tutulduğu bir hafıza kullanır. Ayrıca her karınca bir çözüm oluşturduktan sonra, *k*-API yöntemi uygulanarak çözüm iyileştirilmektedir. Önerilen algoritma, literatürde bu problem için önerilmiş diğer iki sezgisel ile karşılaştırılmış ve SACO' nun mevcut algoritmalarından daha iyi sonuçlar verdiği gösterilmiştir.

Literatürde çok-amaçlı akış tipi çizelgeleme problemi için KKE yaklaşımı kullanılarak yapılmış herhangi bir çalışma ise henüz mevcut değildir.

5.3 Amaçlar

Ele alınan akış tipi çizelgeleme problemi için dikkate alınan toplam tamamlanma zamanı, toplam akış zamanı ve toplam makine aylak kalma zamanı amaçlarının tanımlamaları $C(\pi_i, j)$, permütasyon iş kümesi $\{\pi_1, \pi_2, \dots, \pi_n\}$ için *i* işinin *j* makinesindeki tamamlanma zamanı olmak üzere şöyle yapılabilir:

Toplam tamamlanma zamanı:

$$f_1 = C(\pi_n, m) \quad (5.1)$$

Toplam akış zamanı:

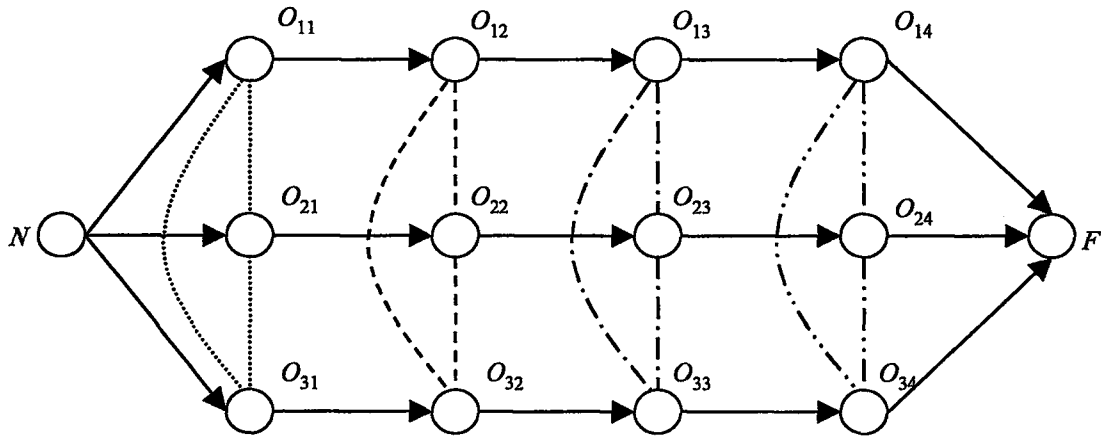
$$f_2 = \sum_{i=1}^n C(\pi_i, m) \quad (5.2)$$

Toplam makine aylak kalma zamanı:

$$f_3 = \left\{ C(\pi_1, j-1) + \sum_{i=2}^n \{ \max \{ C(\pi_i, j-1) - C(\pi_{i-1}, j), 0 \} \} \mid j = 2, \dots, m \right\} \quad (5.3)$$

5.4 Problemin Gösterimi

Çizelgeleme problemleri ayırıcı çizge modeli ile gösterilebilir. Bir n -iş, m -makineli akış problemi $G=(O,C,D)$ birleşik çizge ile tanımlanabilir. Burada, O , düğüm kümesi, C , direk birleştirici oklar kümesi ve D , direk olmayan ayırıcı oklar kümesidir. O düğümleri, tüm o_{ij} işlemleridir (j . makinedeki i . işlem). C , bir işin işlemleri arasındaki öncelik ilişkilerini ve D , farklı işlere ait işlemlerin makine kısıtlarını gösterir. Ayrıca, bir yuva N ve bir yiyecek kaynağı F vardır ve bunlar kukla düğümlerdir. Şekil 5.1’ de 3-iş, 4-makineli bir akış tipi problemi çizge modeli şeklinde gösterilmektedir (Colomi ve diğ., 1994; Ying ve Liao, 2003).



Açıklama: makine 1 için oklar -.-.-.-.- makine 3 için oklar
 - - - - - makine 2 için oklar -.-.-.-.- makine 4 için oklar

Şekil 5.1 Akış Tipi Çizge Modeli

5.5 Yapay Karıncaların Özellikleri

Problemin çözümü için kullanılan kolonideki her k karıncası ($k = 1, 2, \dots, k_{\max}$) aşağıdaki özelliklere sahiptir:

- Amaç fonksiyonuna uygun çözümleri araştırmak için çizge modelini kullanır.
- ana dek geçtiği yollar hakkında bilgi depolamak için bir hafızaya sahiptir. Bu hafıza uygun çözümleri oluşturmak, bulunan çizgelere değerlendirmek ve pheromone depolamak için kullanılır.
- Bir sonraki işi seçmek için bir geçiş kuralı uygular. Bu kural, pheromone izlerinin (τ) ve probleme-özel sezgisel bilginin (η) bir fonksiyonudur.
- Karıncalar geçtikleri yollarda kendilerine izin verildiği takdirde (uygun koşulları sağladıklarında) iz güncelleme işlevlerini yerine getirirler.
- Karıncaların yapısal süreci, durma kriterleri sağlandığında biter.

5.6 Otokatalitik Süreç

Önerilen algoritma için, karıncaların toplu olarak hareket etme özelliği, otokatalitik (autocatalytic) süreç biçiminde ortaya çıkar. Buna göre, bir izi ne kadar çok karınca takip ederse, o iz o kadar cazip hale gelir.

Akış tipi çizelgeleme problemi tıpkı TSP problemi gibi bir permütasyon problemidir. Bazı amaç fonksiyonları için en küçüklendirme yapılacak şekilde bir tamsayı permütasyonu bulunur. İki problemin farkı TSP' de sadece şehirlerin göreceli sıralaması önemli iken, yani; bir permütasyon $\pi_1 = (1\ 2\ \dots\ n)$, aynı amaç fonksiyonu değeri oluşturan permütasyon $\pi_2 = (n\ 1\ 2\ \dots\ n-1)$ ile aynı çözümü gösterir. Akış tipi probleminde ise, işlerin tam pozisyonu önemlidir, yani; permütasyon π_1 ve π_2 farklı toplam tamamlanma zamanına sahip farklı çözümlerdir (Stützle, 1998a).

Ele alınan akış tipi çizelgeleme problemi için uyarlanan KKE algoritmasında, işlerin tam pozisyonu önemlidir. Burada, τ_{ij} , i işinden sonra j işinin sıralanma arzusunu gösterir.

5.7 Yasaklar Listesi

Problem için uygun çözüm bulunurken, bir k karıncası seçeceği bir sonraki işi, geçiş kuralını uygulayarak hesaplar. Seçilen iş yasaklar listesine (constraint satisfaction) eklenir ve son iş seçilene dek süreç tekrarlanır. Bu listenin yasak arama algoritmalarında kullanılan yasaklar listesinden farkı, bu algorithmada seçilmemesi gereken işler bu listeye kaydedilirken, karınca algoritmalarında bu listeye çözümler kaydedilmektedir. Sonuçta, yasaklar listesi iş sıralamasını belirler.

5.8 Karınca Sayısı

Gerçekte tek bir karınca bir çözüm oluşturabilmesine rağmen, etkinlik konusunda, karınca kolonisi kullanımı tercih edilen bir seçenektir. Örneğin, coğrafi dağılım problemleri için, karıncaların bu problemlerin çözümüne oluşturduğu diferansiyel uzaklık etkisi, sadece karınca kolonisi kullanımı ile ortaya çıkabilir. Rotalama problemlerinde de karıncalar, paralel şekilde bir çok en kısa yol problemini çözebilirler ve bu tür problemlerin her biri için karınca kolonisi kullanılmalıdır.

Deneysel çalışmalar da göstermiştir ki, KKE algoritmalarının performansı, $k_{\max}$ karınca sayısı, bir $k_{\max} > 1$ değerine eşitlendiğinde daha iyi olmaktadır (Dorigo ve Di Caro, 1999; Stützle ve Dorigo, 2003).

5.9 Geçiş Olasılığı

Çizelgeleme oluşturma sürecinde, k karıncası tarafından i işinden sonraki j işinin seçimi bir geçiş kuralı ile yapılır. Burada kullanılan geçiş olasılığı kuralı, karınca koloni sistemi sanki-rasgele-oran seçim kuralıdır.

Sanki-rasgele-oran seçim kuralı: Bir çizelge oluşturulurken, i işinden sonraki j işinin seçimi aşağıdaki kural uygulanarak yapılır:

$$j = \begin{cases} \arg \max_{u \in S_k(i)} \{ [\tau(i, u)]^\alpha [\eta(i, u)]^\beta \} & \text{eger } q \leq q_0 \quad (\text{devamlılık}) \\ J & \text{diger} \quad (\text{arastirma}) \end{cases} \quad (5.4)$$

Burada, $\tau(i, u)$, (i, u) üzerindeki pheromone izi, $\eta(i, u) = 1/\delta(i, u)$, u işi ile i işi arasındaki uzaklığın $(\delta(i, u))$ tersidir. $\eta(i, u)$, orijinal karınca koloni sisteminde, i

ve u düğümleri arasındaki bir maliyet ölçüsüne (yani uzaklık) karşılık gelir. Burada kullanılacak uzaklık kavramı (sezgisel bilgi tanımı) daha sonraki kısımda açıklanacaktır. $S_k(i)$, i işinden sonra karınca tarafından seçilebilecek işlerin kümesini gösterir. α , pheromone izinin göreceli önemini belirleyen parametre ($\alpha > 0$); β , sezgisel bilginin göreceli önemini belirleyen parametre ($\beta > 0$); q , $[0,1]$ aralığında düzgün dağılan rasgele bir sayı ve q_0 , araştırmaya karşılık devamlılığın göreceli önemini gösteren bir parametredir ($0 \leq q_0 \leq 1$). Ayrıca, J , k karıncasının, aşağıdaki rasgele-oran kuralına göre i işinden sonra seçeceği j işinin olasılığını veren rasgele bir değişkendir:

$$p_k(i, j) = \begin{cases} \frac{[\tau(i, j)]^\alpha [\eta(i, j)]^\beta}{\sum_{u \in S_k(i)} [\tau(i, u)]^\alpha [\eta(i, u)]^\beta} & \text{eger } j \in S_k(i) \\ 0 & \text{diger} \end{cases} \quad (5.5)$$

i işinden sonra karınca sıralanacak j işini seçmek zorunda olduğu zaman, bir q rasgele sayısı üretir. Eğer $q \leq q_0$ ise, Denklem (5.4)' e göre en iyi iş seçilir, aksi halde Denklem (5.5)' e göre, en iyi iş seçilir.

5.10 Pheromone İzi Güncelleme Kuralları

Bir çok pheromone güncelleme kuralı vardır. Pheromone izi, global veya yerel olarak güncellenebilir. Ayrıca, hangi karıncaların güncelleme yapmak için seçilecekleri de önemlidir. Burada karınca koloni sistemi pheromone izi güncelleme kuralı kullanılacaktır.

Karınca Koloni Sistemi Pheromone İzi Güncelleme Kuralı:

Yerel Güncelleme Kuralı: Bir çizelge oluşturulurken, karınca, aşağıdaki yerel güncelleme kuralını uygulayarak geçilen işler arasındaki pheromone düzeyini azaltır:

$$\tau(i, j) := (1 - \rho l) \cdot \tau(i, j) + \rho l \cdot \tau_0 \quad (5.6)$$

Burada τ_0 , başlangıç pheromone düzeyi ve ρl ($0 < \rho l < 1$), yerel güncellemenin pheromone buharlaştırma parametresidir.

Global Güncelleme Kuralı: Global güncelleme kuralı, tüm karıncalar çizelgelerini tamamladıktan sonra yapılır. Global güncelleme kuralı, en iyi çizelgenin işleri arasına büyük miktarda pheromone bırakılmasını sağlar. Pheromone düzeyi, aşağıdaki gibi düzenlenir:

$$\tau(i, j) := (1 - \rho g) \cdot \tau(i, j) + \rho g \cdot \Delta \tau(i, j) \quad (5.7)$$

burada,

$$\Delta \tau(i, j) = \begin{cases} (L_b)^{-1} & \text{eger } (i, j) \in \text{en iyi çizelge} \\ 0 & \text{diger} \end{cases} \quad (5.8)$$

Yukarıdaki denklemde, ρg ($0 < \rho g < 1$), global güncellemenin pheromone buharlaştırma parametresi ve L_b , o anki iterasyon için en iyi çizelgenin amaç fonksiyonu değeridir.

5.11 Sezgisel Bilgi

Karıncaların olasılıksal çözüm yapısını yönetmek için sezgisel bilginin (greedy force) kullanımı önemlidir. Çünkü, probleme özel bilginin kullanılmasını mümkün kılar. Bu bilgi, öncül (statik durum) veya çalışma zamanlı (dinamik durum) olabilir. Statik problemlerde, sezgisel bilgi, başlangıçta bir kez hesaplanır ve daha sonra tüm algoritma boyunca sabit kalır. Statik sezgisel bilgini avantajı, kolay hesaplanması ve hesaplanma zamanından ciddi kazanç sağlanmasıdır. Dinamik durumda, sezgisel bilgi, o ana kadar oluşturulan kısmi çözümlere bağlıdır ve böylece karınca hareketlerinin her basamağında tekrar hesaplanır. Bu yüksek hesaplama maliyetine karşılık, tam (kesin) sezgisel bilgi değerinin hesaplanmasını sağlar (Stützle ve Dorigo, 2003). Burada kullanılan sezgisel bilgi Palmer eğim indeksi aşağıda tanımlanmıştır (Ying ve Liao, 2004):

İki iş arası uzunluk (lenght between two jobs): İşler arası uzaklığı tanımlamak için $n/m/P/C_{\max}$ akış tipi problemi için Palmer (1965) tarafından önerilen eğim indeksi s' den faydalanılacaktır. Bir i ($i=1,2,...,n$) işi için eğim indeksi s_i şöyle tanımlanır:

$$s_i = -(m-1)t_{i1} - (m-3)t_{i2} - (m-5)t_{i3} + ... + (m-3)t_{i,m-1} + (m-1)t_{im} \quad (5.9)$$

Daha sonra şu iş sıralaması oluşturulur: $s_{[1]} \geq s_{[2]} \geq \dots \geq s_{[n]}$. Palmer sezgiselindeki temel düşünce, süreçte işlem süreleri makineden makineye azalma eğilimine sahip işlere yüksek öncelik verilirken, artış gösteren işlere ise düşük öncelik vermektir. Bu düşünceyi takip ederek, eğim indeksi revize edilir ve i ($i=1,2,\dots,n \cup N$) işi ile u ($u=1,2,\dots,n$) işi arasındaki uzaklık şöyle tanımlanır:

$$\delta(i,u) = 1/\eta(i,u) \quad (5.10)$$

burada,

$$\eta(i,u) = -(m-1)t_{u,1} - (m-3)t_{u,2} - (m-5)t_{u,3} + \dots + (m-3)t_{u,m-1} + (m-1)t_{u,m} - \min_u \{\eta(i,u)\} + 1 \quad (i \neq u). \quad (5.11)$$

i ($i=1,2,\dots,n \cup N$) işi ile u ($u=1,2,\dots,n$) işi arasındaki uzaklık, u işinin eğim indeksine eşittir.

5.12 Aday Liste

KKE algoritmalarında ortaya çıkan zorluklardan biri, çizelge oluştururken büyük ölçülere sahip komşulukların olmasıdır. Büyük ölçülere sahip komşuluğu olan bir işi çizelgeleyen bir karınca, seçmek için çok fazla miktarda seçeneğe sahip olur. Bu ise, çözüm yapısının gittikçe yavaşlamasına ve birçok karıncanın aynı işi çizelgeleme olasılığının düşük olmasına neden olur. Aday liste kullanımı ile bu gibi problemler azaltılabilir (Stützle ve Dorigo, 2003).

Aday listesi, ek yerel sezgisel bilgi sağlayan bir veri yapısı sağlamaktadır. Aday listesi, verilen bir işten sonra tercih edilecek işlerin bir listesidir. Bir karınca, verilen bir işten sonra gelecek tüm çizelgelenmemiş işleri araştırmak yerine, aday listedeki işler arasından bir işi seçer. Aday listesinde çizelgelenmemiş iş kalmadıysa, diğer işler araştırılabilir. Bir işin aday listesi, örneğin işlem zamanına göre artan şekilde sıralanmış cl işten oluşur. Liste sırasıyla ve çizelgelenmiş işlere gidişi önlemek için karıncanın yasaklar listesine göre taranır.

5.13 Yerel Geliştirme Yöntemleri

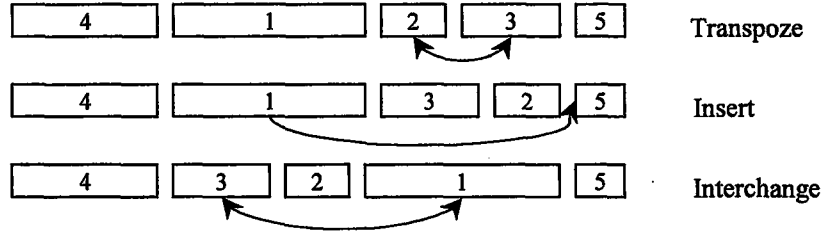
Geliştirici algoritmalar, yerel arama algoritmalarına göre, daha zayıf çözümler üretirler. Diğer yandan, rasgele üretilen başlangıç çözümleri ile yerel aramaları tekrar ettirmek, hala birçok problemin eniyi çözümü için ciddi bir boşluk oluşturmaktadır. Bu nedenle, kaliteli çözümler bulmak için melez yaklaşımlara ihtiyaç duyulmaktadır. En iyi kaliteli çözümler bazı çözüm biçimleri ile yerel aramanın birleştirilmesi ile bulunur. Çok sık kullanılan fikir, adaptif geliştirici sezgisel ile yerel aramanın birleştirilmesidir. Bu durumda, geliştirici sezgiseller, yerel arama sezgiselleri için iyi bir başlangıç çözümü sağlarlar. KKE algoritmaları da, böyle adaptif geliştirici sezgiselleridir. Bunlar, pheromone izleri ile yansıtılan geçmiş deneyimlerine göre yüksek kalitede çözümler üreten çözüm özelliklerini belirlemek ve bu şekilde arama uzayında ümit verici bölgelere doğru yerel aramaya yol göstermek için kullanılırlar. Ayrıca, iyi başlangıç çözümler oluşturdıkları için, yerel arama, yerel eniyiye ulaşmada daha az iterasyon gerektirir. Böylece, verilen bir zaman sınırında, rasgele üretilen başlangıç çözümlerden başlamaya göre daha çok yerel arama yapılabilir (Stützle, 1997).

Permütasyon gösterimi (n -işli bir problem, $1, \dots, n$ tamsayı permütasyonu ile gösterilir) için üç komşuluk türü vardır (den Besten, 2000):

- **Transpose:** Bitişik iki işin yer değiştirilmesidir. O anki permütasyon $\pi = (\pi_1, \dots, \pi_i, \pi_{i+1}, \dots, \pi_n)$, $\pi' = (\pi_1, \dots, \pi_{i+1}, \pi_i, \dots, \pi_n)$ şeklinde değişir.
- **Interchange:** İki işin karşılıklı yer değiştirmesidir. Interchange hareketinde $\pi = (\pi_1, \dots, \pi_i, \dots, \pi_j, \dots, \pi_n)$, $\pi' = (\pi_1, \dots, \pi_j, \dots, \pi_i, \dots, \pi_n)$ şeklinde değişir.
- **Insert:** (i, j) , pozisyon çifti ve $\pi = (\pi_1, \dots, \pi_{i-1}, \pi_i, \pi_{i+1}, \dots, \pi_j, \pi_{j+1}, \dots, \pi_n)$, o anki permütasyon olsun. Yeni permütasyon π' , i pozisyonundaki işin yerinden alınıp j pozisyona eklenmesi ile oluşturulur. Eğer $i < j$ ise, sağdan ekleme yapılır ve $\pi' = (\pi_1, \dots, \pi_{i-1}, \pi_{i+1}, \dots, \pi_j, \pi_i, \pi_{j+1}, \dots, \pi_n)$ oluşur. Eğer $i > j$ ise, soldan ekleme yapılır ve $\pi' = (\pi_1, \dots, \pi_{j-1}, \pi_i, \pi_j, \dots, \pi_{i-1}, \pi_{i+1}, \dots, \pi_n)$ oluşur.

Terminolojide tam bir fikir birliği yoktur. Transpose, bazen Adjacent Pairwise Interchange (API) olarak; interchange, genel pairwise interchange olarak; ve insert, shift olarak bilinir. Şekil 5.2' de bu komşulukların örnek şeklinde gösterimi

görülmektedir. Transpose, insert ve interchange için n işli bir sıralama üzerindeki komşuluk sayısı ise sırasıyla $(n-1)$, $(n-1)^2$ ve $n(n-1)/2$ olarak hesaplanır.



Şekil 5.2 Komşuluk Yapıları

5.14 Geliştirilen KKE Yaklaşımının Genel Yapısı

KKE algoritmalarında, çözüm oluşturacak karıncalara, pheromone izleri (τ) ve probleme-özel sezgisel bilgi (η) ile yol gösterilir. KKE algoritmaları, çözüm bileşenleri tanımlanarak birleşik eniyileme problemlerine uygulanabilir. Birleşik eniyileme problemlerinde, pheromoneler, çözüm bileşenlerine göre karınca-izlerinin yoğunluğunu belirler. Bu yoğunluk, her bir çözüm bileşeninin amaç fonksiyonuna etkisi veya katkısına göre belirlenir. Bireysel bir karınca, boş bir çözümle başlayarak, tam bir çözüm oluşturulana dek iteratif olarak çözüm bileşenleri ekleyerek tam bir çözüm oluşturur. Tam bir çözüm oluşturulduktan sonra, her karınca, her çözüm bileşeni üzerinde pheromone birikimi ile çözüm üzerine geri bilgi verir.

KKE yaklaşımının genel yapısı Şekil 5.3' de görülmektedir. İlk olarak, başlangıç aşamasında, başlangıç pheromone izleri, işler arasındaki uzaklıklar ve parametreler $q_0, \alpha, \beta, \rho_l, \rho_g$, karınca sayısı $k_{\max}$ ve maksimum iterasyon sayısı $t_{\max}$ belirlenir.

İkinci olarak, iteratif süreçte, karınca kolonisi, başlangıç işini seçer. Her karınca, tam bir çizelge oluşturana dek, bir sonraki işlemi seçmek için geçiş kuralını uygular. Bir çizelge oluşturulurken, hem sezgisel bilgi hem de pheromone miktarı seçilecek işi belirlemek için kullanılır. Sezgisel bilgi değeri ve pheromone miktarı ne kadar yüksekse olasılıkla o kadar yüksektir. Çizelge oluşturulurken, bir karınca, diğer karıncaların turunu değiştirmek ve yerel eniyiden kaçınmak için yerel güncelleme kuralını uygulayarak seçilen işler arasındaki pheromone miktarını azaltır. Tüm karıncalar çizelgelerini tamamlar tamamlamaz, yerel iyileştirme yapılır ve daha sonra o anki iterasyondaki en iyi çizelgenin işleri arasındaki pheromone miktarını arttırmak

ve diğer yolların pheromone miktarını azaltmak için global güncelleme kuralı uygulanır. Böylece tüm karıncalar, en iyi çizelge üzerinde odaklanacaklardır.

Üçüncü adım, durma kriteri olarak, maksimum iterasyon sayısına ulaşılan dek süreç tekrarlanır.

Adım1. Başlangıç: Başlangıç pheromone izleri, işler arasındaki uzaklıklar ve parametreleri belirle.

Adım2. İteratif süreç:

- a) Tüm karıncalar için başlangıç işi belirle.
- b) Her karınca için bir çizelge oluştur:
Tüm karıncalar çizelgelerini tamamlayana dek;
Bir sonraki işlemi seçmek için geçiş kuralını uygula.
Yerel güncelleme kuralını uygula.
- c) Yerel arama uygula.
- d) Global güncelleme kuralını uygula.

Adım3. Durma kriteri: Eğer maksimum iterasyon sayısına ulaşılmışsa, dur; aksi halde Adım2' ye git.

Şekil 5.3 KKE Yaklaşımının Genel Yapısı

Akış tipi çizelgelemede, tüm işler her makinede aynı sıralamaya sahip olduğu için, sadece ilk grup sıralamayı bulmak yeterlidir. Uygun çözüm bulunurken, karınca, seçeceği bir sonraki işi, geçiş kuralını uygulayarak hesaplar. Seçilen iş yasaklar listesine (tabu_k) eklenir ve son iş seçilene dek süreç tekrarlanır. Sonuçta, yasaklar listesi iş sıralamasını belirler.

Bunun yanında karınca sayısı, bir olarak seçilmiştir. Yani; her iterasyonda bir karınca, tek bir çözüm oluşturmaktadır. Algoritma Şekil 5.4' de görülmektedir.

Adım1. Başlangıç:

For her (i, u) işi **do**

başlangıç pheromone izleri $\tau(i, u) = \tau_0$,

işler arası uzaklıklar $\eta(i, u)$,

parametreler $k_{\max}, q_0, \alpha, \beta, \rho_l, \rho_g, t_{\max}$ belirle.

EndFor

T^* , en iyi sıralama ve L^* , bu sıralamanın amaç değeri olsun.

Adım2. İteratif süreç:

For $t = 1$ to $t_{\max}$ **do**

Adım2.1 Başlangıç işlerin belirlenmesi:

For $k = 1$ to $k_{\max}$ **do**

k karıncası için başlangıç işleri belirle,

Bu bilgiyi tabu_k ' ya kaydet.

EndFor

Adım2.2 Her karınca için bir çizelge oluşturulması:

For $k = 1$ to $k_{\max}$ **do**

Aday listedeki işler arasından Denklem(5.4) ve Denklem(5.5)' e göre sonraki işi belirle.

Bu bilgiyi tabu_k ' ya kaydet.

Denklem(5.6)' ya göre yerel güncelleme uygula.

EndFor

Adım2.3 Çözümlerin değerlendirilmesi:

For $k = 1$ to $k_{\max}$ **do**

$T_k(t)$ sıralaması için $L_k(t)$ amaç değerini hesapla.

$T_k(t)$ sıralaması için yerel arama uygula.

Eğer daha iyi bir sıralama elde edildiyse, T^* ve L^* ' yı güncelle.

EndFor

Adım2.4 Denklem(5.7)'ye göre global güncelleme uygulanması.
Tüm tabu_k ' ları boşalt.

EndFor

Adım3. T^* , en iyi sıralama ve L^* , bu sıralamanın amaç değerini göster.

Şekil 5.4 Akış Tipi Çizelgeleme Problemi için KKE Algoritması

6. DENEYSEL ÇALIŞMALAR

Bu bölümde öncelikle akış tipi çizelgeleme problemi için öngörülen KKE algoritmasının, uygun çözümler verdiğini göstermek üzere tek amaç için (toplam tamamlanma zamanı) test problemleri üzerindeki denemeleri verilmiştir. Daha sonra algoritmanın performansını geliştirmek için bazı mekanizmaları üzerinde deneysel çalışmalar yapılmıştır. Sonuçta elde edilen geliştirilmiş algoritmanın performansı literatürde mevcut en iyi algoritmaların performansları ile karşılaştırılmıştır. Son olarak algoritmanın performansı çok-amaçlı akış tipi çizelgeleme problemi için (toplam tamamlanma zamanı, toplam akış zamanı ve toplam makine aylak kalma zamanı amaçlarına göre) test problemleri üzerinde denenmiştir.

6.1 Tek-Amaçlı Durum için Denemeler ve Hesaplama Sonuçları

6.1.1 Test Problemleri

Çizelgeleme problemlerinde araştırmacıların karşılaştığı bir zorluk, diğer araştırmacıların geliştirdiği sezgiseller ile yapılan çalışmaların karşılaştırılmasıdır. Standart test problemlerine erişilebildiği takdirde, farklı algoritmaların performansı, aynı test problemleri üzerinde karşılaştırılabilir.

Bu çalışma kapsamında ilk olarak oluşturulan algoritmanın doğru ve kaliteli sonuçlar verip vermediğini test etmek üzere tek amaçlı (toplam tamamlanma zamanı) problem için algoritma çalıştırılmıştır. Bu testler için Taillard tarafından sunulan test problemleri kullanılmıştır. Test problemlerinin dosyaları Taillard' ın web sitesinden (URL:<http://www.idsia.ch/~eric/>) veya OR-library web sitesinden (URL:<http://mscmga.ms.ic.ac.uk/info.html>) elde edilebilir. Taillard, 5, 10 ve 20 makine ve 20, 50 ve 100 işten oluşan 90 test problemini rassal olarak şöyle oluşturmuştur: her i işi ($i = 1, \dots, n$) ve her j makinesi ($j = 1, \dots, m$) için işlem zamanı t_{ij} , uniform dağılımdan $[1, 99]$ geliştirilmiştir. Her problem büyüklüğü için 10 örnek oluşturulmuştur. Ayrıca her örnek için, başlangıç çekirdek (seed) değeri, eniyi toplam tamamlanma zamanı üst ve alt sınır değerlerini de vermiştir.

Algoritmanın ilk testlerinde ve performansını geliştirmek üzere yapılan denemeler Taillard' ın 20x5 (20-iş, 5-makine), 20x10 ve 20x20 problem hacmindeki 30 test problemi üzerinde yapılmıştır.

Geliştirilen algoritma Pascal ile kodlanmış ve PentiumIII 933 MHz PC üzerinde çalıştırılmıştır.

6.1.2 Başlangıç Çözümü ve Parametre Değerleri

Başlangıç iz yoğunluğu (τ_0) için, bu çalışmada, literatürde kabul gören başlangıç pheromone miktarı $\tau_0 = (n.OS)^{-1}$ kullanılmıştır. Burada n , problem büyüklüğü ve OS , kullanılan tek-amaçlı akış tipi problemi test problemleri için bilinen eniyi toplam tamamlanma zamanı üst sınır değeridir.

Geliştirilen karınca algoritmasında ilk çözüm için geçiş kuralı için sanki-rasgele-oran seçim kuralı (Denklem(5.4) ve (5.5)), pheromone güncelleme kuralı için karınca koloni sistemi pheromone güncelleme kuralı (Denklem(5.6) ve (5.7)), sezgisel bilgi için Palmer eğim indeksi (Denklem(5.11)) kullanılmıştır.

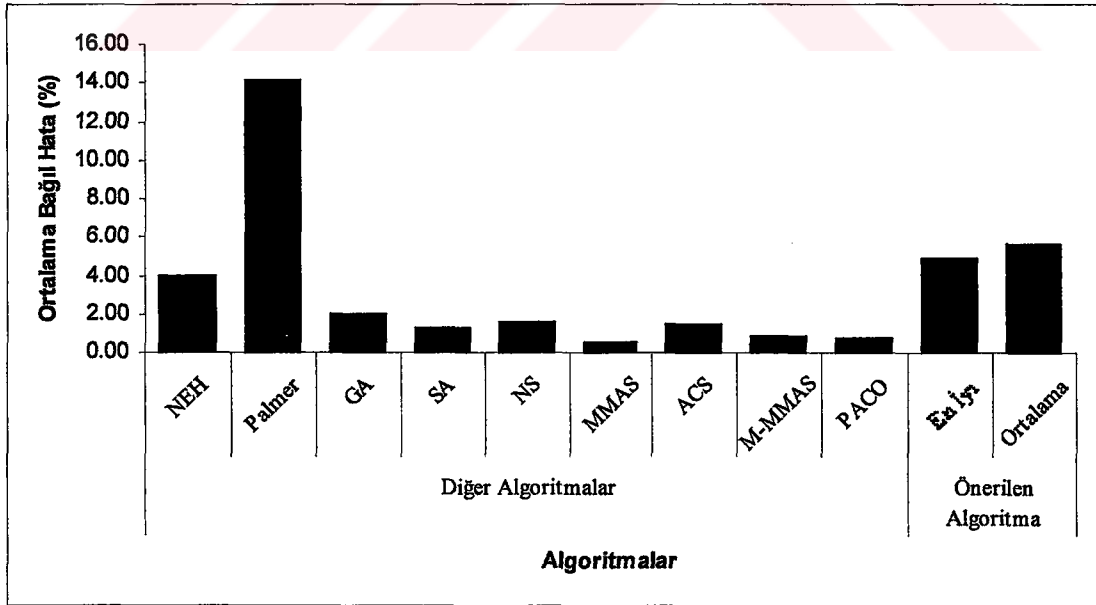
Algoritma Taillard' ın 20x5, 20x10 ve 20x20 problem hacmindeki 30 test problemi üzerinde test edilmiştir. Bunun için Dorigo ve Gambardella (1997) tarafından kullanılan ve önerilen başlangıç parametre değerleri $\alpha = 1$, $\beta = 2$, $\rho_l = 0.1$, $\rho_g = 0.1$, $q_0 = 0.9$, karınca sayısı $k_{max} = 20$ ve maksimum iterasyon sayısı $t_{max} = 5000$ olarak sabitlenmiştir. Her bir deneme 10 kez tekrarlanmış ve böylece 300 adet deneme yapılmıştır. Bu başlangıç testleri için algoritmada yerel arama ve aday liste özellikleri kullanılmamıştır. Deneme sonuçları Ek C' de verilmektedir. Her problem için denemelerin en iyi (Δ_{eniyi}), ortalama (Δ_{ort}), en kötü ($\Delta_{enkötü}$) ve standart sapma (Δ_{std}) değerlerine göre aritmetik ortalaması alınmıştır. Sonuçlar, Taillard' ın eniyi toplam tamamlanma zamanı değerinin üst sınır değerlerinden ortalama yüzde fark olarak Ek C' de verilmektedir.

Önerilen algoritmanın performans sonuçları toplam tamamlanma zamanı en küçüklmesi için literatürdeki mevcut diğer sezgiseller (NEH ve Palmer), metasezgiseller (GA, SA ve NS) ve karınca kolonileri algoritmaları (MMAS, ACS, M-MMAS ve PACO) ile karşılaştırılmıştır. Algoritmaların performans sonuçları özet olarak Tablo 6.1 ve Şekil 6.1' de görülmektedir.

Tablo 6.1 İlk Çözüm için Algoritma Performans Değerlendirmesi

n	m	Problem No	Diğer Algoritmalar									Önerilen Algoritma			
			NEH	Palmer	GA	SA	NS	MMAS	ACS	M_MMAS	PACO	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}
20	5	ta001-010	3.66	10.81	1.61	1.27	1.46	0.41	1.19	0.76	0.70	4.01	4.24	4.56	2.35
	10	ta011-020	4.60	15.27	2.29	1.71	2.02	0.59	1.70	0.89	0.84	6.12	7.22	8.02	8.88
	20	ta021-030	3.73	16.34	1.95	0.86	1.10	0.41	1.60	0.72	0.72	4.50	5.43	6.10	11.93
Ortalama=			4.00	14.14	1.95	1.28	1.53	0.47	1.50	0.79	0.76	4.88	5.63	6.23	7.72

Burada algoritmanın, bu problem için literatürde mevcut olan diğer sezgisel, metasezgisel ve karınca algoritmalarından daha kötü sonuçlar verdiği görülmüştür. Bu nedenle algoritma üzerinde bazı iyileştirme çalışmaları yapılarak daha iyi performans üreten bir algoritma haline dönüştürülmeye çalışılmıştır. Bu amaçla, ilk olarak probleme özel en iyi parametre değerlerini bulmak üzere bir parametre analizi yapılmıştır. Daha sonra ise, algoritmaya ait mekanizmaları üzerinde denemeler yapılarak bunların problem için en uygun yapıları bulunmuştur.



Şekil 6.1 İlk Çözüm için Algoritma Ortalama Performans Değerlendirmesi

6.1.3 Parametre Analizi

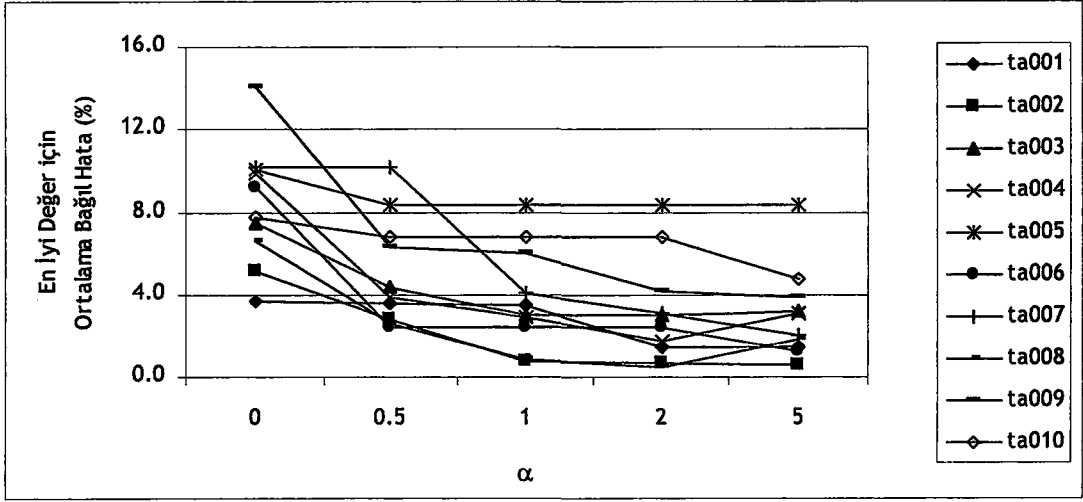
6.1.3.1 Geçiş Kuralı Parametre Analizi

Algoritmayı test etmek için daha önceki adımda literatürde tavsiye edilmiş parametre değerleri kullanılmıştı. Burada, daha iyi sonuçlar elde etmek için bu parametrelerin probleme özgü en uygun değerleri bulunmaya çalışılmıştır. Algoritmanın geçiş kuralında kullanılmış parametreler:

- α , pheromone izinin göreceli önemini belirleyen parametre ($\alpha \geq 0$),
- β , uzaklığın göreceli önemini belirleyen parametre ($\beta \geq 0$),
- ρl , yerel güncellemenin pheromone buharlaştırma parametresi ($0 \leq \rho l < 1$),
- ρg , global güncellemenin pheromone buharlaştırma parametresi ($0 \leq \rho g < 1$),
- q_0 , araştırmaya karşılık devamlılığın göreceli önemini gösteren bir parametre.

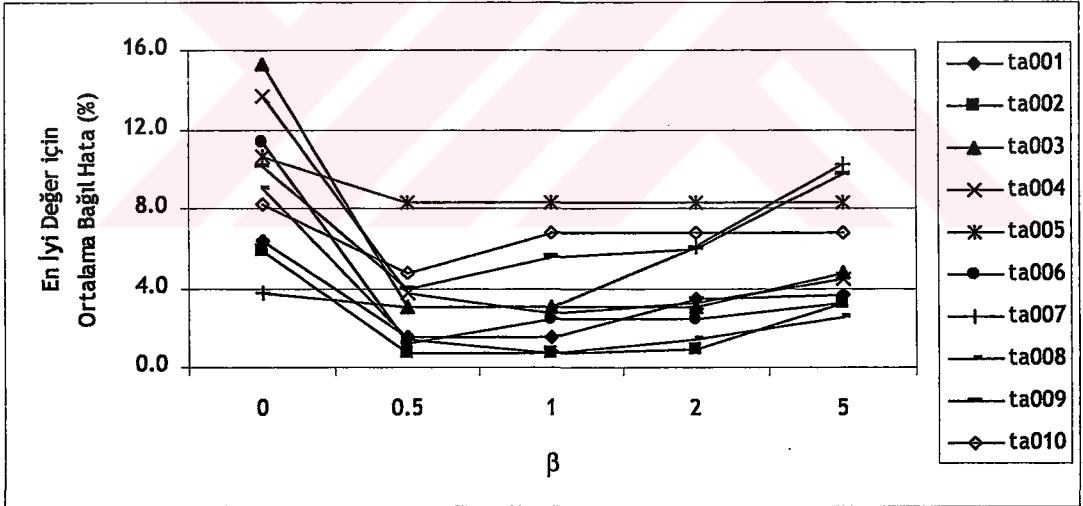
En uygun parametre değerlerini bulmak üzere çok sayıda deney yapılmıştır. Öncelikle başlangıç parametrelerinin alacağı değerler için bir eğilim tespit etmek amacı ile Taillard' ın 20x5 boyutundaki 10 örnek test problemi üzerinde denemeler yapılmış ve parametreler için test edilecek uygun değerleri tespit edilmiştir. Başlangıç parametre değerleri için $\alpha = 1$, $\beta = 2$, $\rho l = 0.1$, $\rho g = 0.1$ ve $q_0 = 0.9$ varsayılan değer olarak alınarak her seferinde bunlardan biri test edilmiştir. Ayrıca, karınca sayısı $k_{max} = 20$ ve maksimum iterasyon sayısı $t_{max} = 5000$ olarak sabitlenmiştir. Her bir deneme 5 kez tekrarlanmış ve böylece toplam 1250 adet deneme yapılmıştır. Bunun için kullanılan parametre değerleri şöyledir: $\alpha \in \{0, 0.5, 1, 2, 5\}$, $\beta \in \{0, 0.5, 1, 2, 5\}$, $\rho l \in \{0, 0.1, 0.2, 0.3, 0.5\}$, $\rho g \in \{0, 0.1, 0.2, 0.3, 0.5\}$ ve $q_0 \in \{0, 0.25, 0.5, 0.75, 0.9\}$. Bu testler için algoritmada yerel arama ve aday liste özellikleri kullanılmamıştır. Deneme sonuçları, en iyi ve ortalama değerlere göre denemelerin aritmetik ortalaması, Taillard' ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Ek D' de verilmektedir. Denemeler sonucu $\alpha, \beta, \rho l, \rho g, q_0$ parametrelerinin eğilimleri sırasıyla Şekil 6.2, Şekil 6.3, Şekil 6.4, Şekil 6.5 ve Şekil 6.6' da verilmektedir.

Sonuçlar, parametreler için alınan bu test değerlerinin uygun olduğunu ve denemeler için kullanılabileceğini göstermiştir.



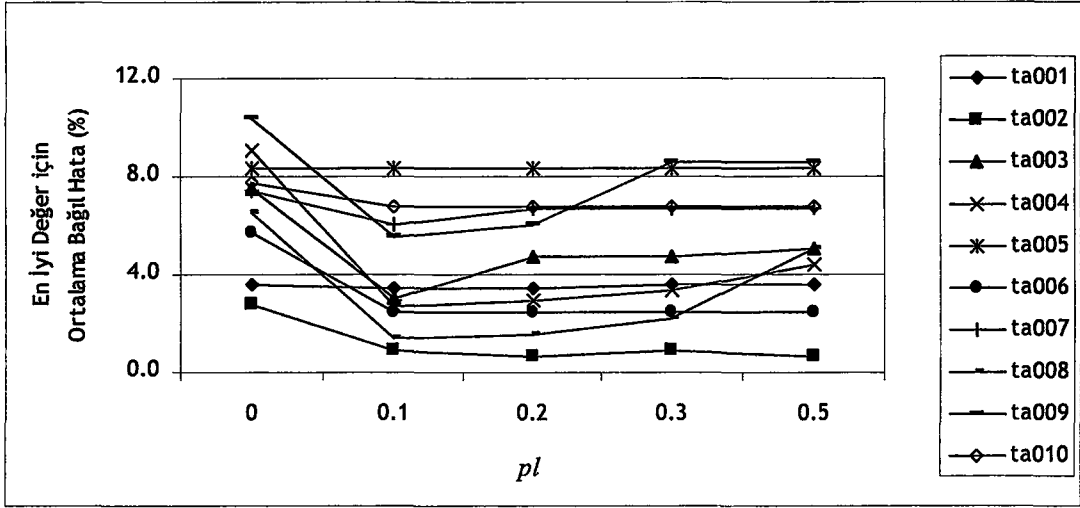
Şekil 6.2 α Parametresi Eğilim Sonuçları

Şekil 6.2' den α parametresinin kullanılmadığı durumda algoritmanın en kötü sonuçları verdiği görülmüştür. Ayrıca algoritma, bu parametrenin yüksek değerleri için daha kötü ve ortalama değerleri için ise en iyi sonuçları vermiştir.



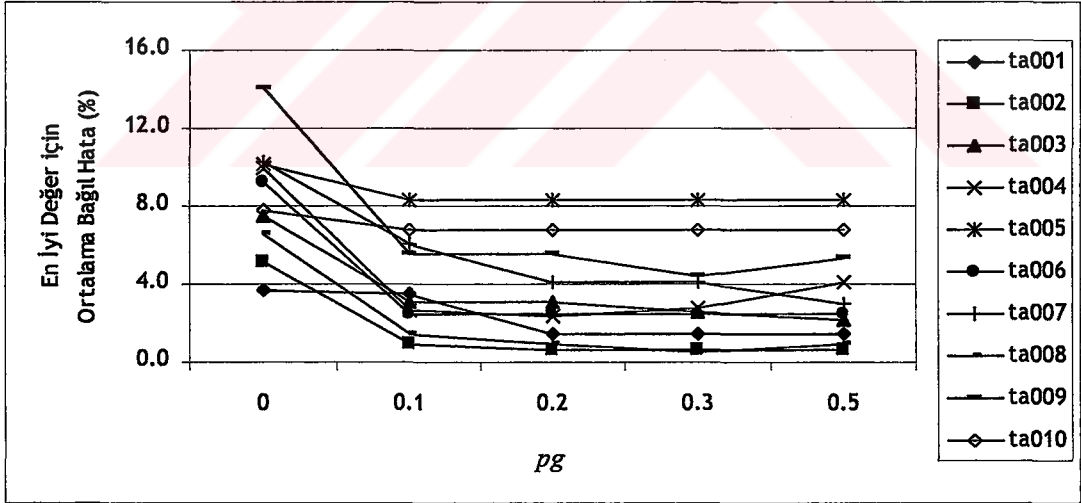
Şekil 6.3 β Parametresi Eğilim Sonuçları

Şekil 6.3' den β parametresinin kullanılmadığı durumda ve yüksek değerlerde algoritmanın en kötü sonuçları verdiği, ortalama değerler için ise en iyi sonuçları verdiği görülmüştür.



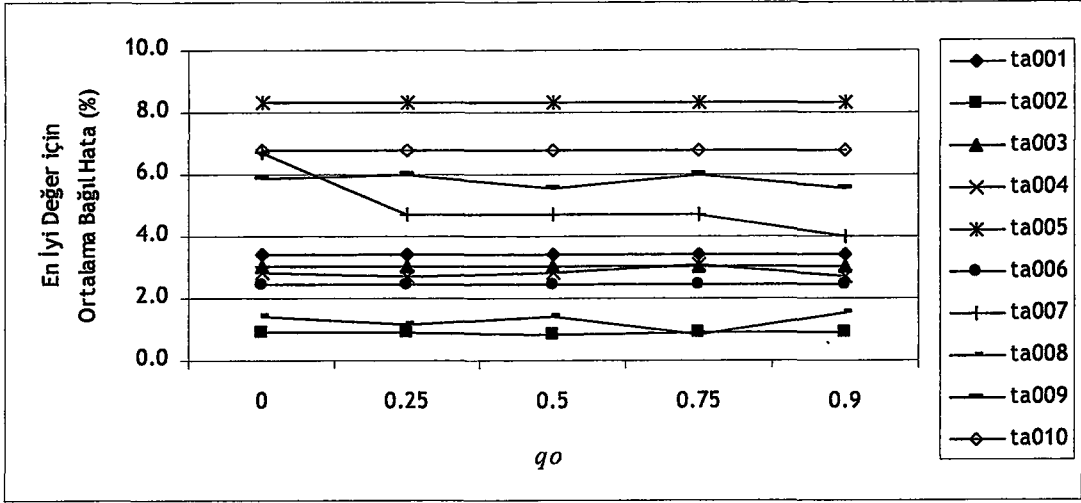
Şekil 6.4 pl Parametresi Eğilim Sonuçları

Şekil 6.4' den pl parametresinin kullanılmadığı durumda algoritmanın en kötü sonuçları verdiği görülmüştür. Ayrıca algoritma, bu parametrenin yüksek değerleri için kullanılmadığı durumdaki kadar kötü sonuçlar vermiştir. Ortalama değerler için ise en iyi sonuçları verdiği görülmüştür.



Şekil 6.5 pg Parametresi Eğilim Sonuçları

Şekil 6.5' den pg parametresinin kullanılmadığı durumda algoritmanın en kötü sonuçları verdiği, ortalama ve yüksek değerler için ise en iyi sonuçları verdiği görülmüştür.



Şekil 6.6 q_0 Parametresi Eğilim Sonuçları

Şekil 6.6' dan q_0 parametresi için test edilen değerler arasında performans açısından çok küçük fark olduğu görülmüştür.

Daha sonra denemeler parametrelerin eniyi değerlerini bulmak üzere Taillard' ın 20x10 boyutundaki 4 örnek test problemi üzerinde genişletilmiştir. Başlangıç parametre değerleri için $\alpha = 1$, $\beta = 2$, $\rho_l = 0.1$, $\rho_g = 0.1$ ve $q_0 = 0.9$ varsayılan değer olarak alınarak her seferinde bunlardan biri test edilmiştir. Ayrıca, $k_{\max} = 20$ ve $t_{\max} = 5000$ olarak sabitlenmiştir. Her bir deneme 10 kez tekrarlanmış ve böylece toplam 1000 adet deneme yapılmıştır. Bu testler için yerel arama ve aday liste özellikleri kullanılmamıştır. Deneme sonuçları, en iyi ve ortalama değerlere göre denemelerin aritmetik ortalaması, Taillard' ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Ek D' de verilmektedir.

Denemeler tamamlandıktan sonra parametrelere ait bulunan en küçük iki değer için ($\alpha = 1, 2$, $\beta = 0.5, 1$, $\rho_l = 0.1, 0.2$ ve $\rho_g = 0.1, 0.2$) iki seviyeli Taguchi analizi yapılmıştır. Burada q_0 parametresi için test edilen değerler arasında çok küçük fark olduğu ve parametre değeri büyüdükçe çözüm süresi kısaldığı için en yüksek değer ($q_0 = 0.9$) tercih edilmiştir. Ortaya çıkan sonuçlara göre parametreler ve seviyeleri Tablo 6.2' de görülebilir. Taguchi analizi için dört faktör ve iki seviye için uygun olan standart L8 ortogonal diziler tablosu Tablo 6.3' de görülmektedir. Bu tabloya bağlı olarak oluşturulan deney tasarımı Tablo 6.4' de görülmektedir.

Tablo 6.2 Taguchi Analizinde Kullanılan Faktör ve Seviyeleri

Faktör No	Faktör	Seviyeler	
		1	2
1	α	1	2
2	β	0.5	1
3	pl	0.1	0.2
4	pg	0.1	0.2

Tablo 6.3 Taguchi Analizi L8 Ortogonal Diziler

Deneme No	Sütun No						
	1	2	3	4	5	6	7
1	1	1	1	1	1	1	1
2	1	1	1	2	2	2	2
3	1	2	2	1	1	2	2
4	1	2	2	2	2	1	1
5	2	1	2	1	2	1	2
6	2	1	2	2	1	2	1
7	2	2	1	1	2	2	1
8	2	2	1	2	1	1	2

Tablo 6.4 L8 Tablosuna Bağlı Deney Tasarımı

Deneme No	Parametreler			
	α	β	pl	pg
1	1	0.5	0.1	0.1
2	1	0.5	0.1	0.2
3	1	1	0.2	0.1
4	1	1	0.2	0.2
5	2	0.5	0.2	0.1
6	2	0.5	0.2	0.2
7	2	1	0.1	0.1
8	2	1	0.1	0.2

Denemeler Taillard' ın 20x10 boyutundaki 10 örnek test problemi üzerinde yapılmıştır. Başlangıç parametre değerleri için $\alpha = 1$, $\beta = 1$, $\rho l = 0.1$, $\rho g = 0.1$ ve $q_0 = 0.9$ varsayılan değer olarak alınarak her seferinde bunlardan biri test edilmiştir. Ayrıca, $k_{\max} = 20$ ve $t_{\max} = 5000$ olarak sabitlenmiştir. Bu testler için yerel arama ve aday liste özellikleri kullanılmamıştır. Her bir deneme 10 kez tekrarlanmış, böylece toplam 800 adet daha deneme yapılmıştır. Deneme sonuçları Ek D' de verilmektedir. Sırasıyla en iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması, Taillard' ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Tablo 6.5, Tablo 6.6 ve Tablo 6.7' de verilmektedir.

Tablo 6.5 En İyi Değerlere Göre Deney Tasarımı Sonuçları

Deneme No	Problem No										Ortalama
	<i>ta011</i>	<i>ta012</i>	<i>ta013</i>	<i>ta014</i>	<i>ta015</i>	<i>ta016</i>	<i>ta017</i>	<i>ta018</i>	<i>ta019</i>	<i>ta020</i>	
1	2.47	5.36	2.74	5.66	5.21	2.79	4.99	3.25	6.91	4.21	4.36
2	3.54	6.15	6.02	4.65	5.71	2.58	4.65	7.09	6.78	3.46	5.06
3	4.05	7.47	2.87	6.54	5.92	3.51	6.47	3.25	8.85	5.28	5.42
4	3.10	6.51	2.94	6.10	7.54	2.58	5.66	2.86	7.22	5.09	4.96
5	3.29	5.73	2.87	5.01	7.12	2.93	3.17	2.15	6.28	4.90	4.34
6	3.16	5.06	2.74	4.21	7.19	2.22	3.10	2.54	5.15	4.78	4.01
7	3.10	6.27	3.94	4.65	5.71	2.15	5.39	2.60	6.40	4.53	4.47
8	4.30	6.63	6.02	4.58	7.26	3.15	4.18	6.89	6.47	4.27	5.37

En iyi değerler için yapılan çalışmalar da 6 numaralı deney en iyi, 3 numaralı deney en kötü sonucu vermiştir. Elde edilen gelişme böylece ortalama %26 oranında olmuştur.

Tablo 6.6 Ortalama Değerlere Göre Deney Tasarımı Sonuçları

Deneme No	Problem No										Ortalama
	ta011	ta012	ta013	ta014	ta015	ta016	ta017	ta018	ta019	ta020	
1	4.12	6.47	4.43	6.60	7.14	3.56	6.08	4.06	7.34	4.93	5.47
2	5.28	6.94	7.27	5.95	7.72	4.16	5.40	8.45	7.51	4.77	6.34
3	4.41	8.34	3.60	6.80	9.08	3.89	7.09	3.89	9.13	6.49	6.27
4	4.21	7.38	3.52	6.64	8.71	3.56	6.67	3.42	8.44	5.68	5.82
5	4.05	7.09	3.30	5.87	8.15	3.40	5.40	3.27	7.26	5.59	5.34
6	3.77	6.28	3.32	4.79	7.78	3.20	3.72	3.05	6.35	5.51	4.78
7	4.08	6.67	5.19	6.22	7.28	3.39	6.24	3.56	7.39	5.25	5.53
8	5.51	7.29	7.73	6.21	8.45	4.61	5.71	8.62	7.47	4.85	6.64

Ortalama değerler için yapılan çalışmalar da ise 6 numaralı deney en iyi, 8 numaralı deney en kötü sonucu vermiştir.

Tablo 6.7 En Kötü Değerlere Göre Deney Tasarımı Sonuçları

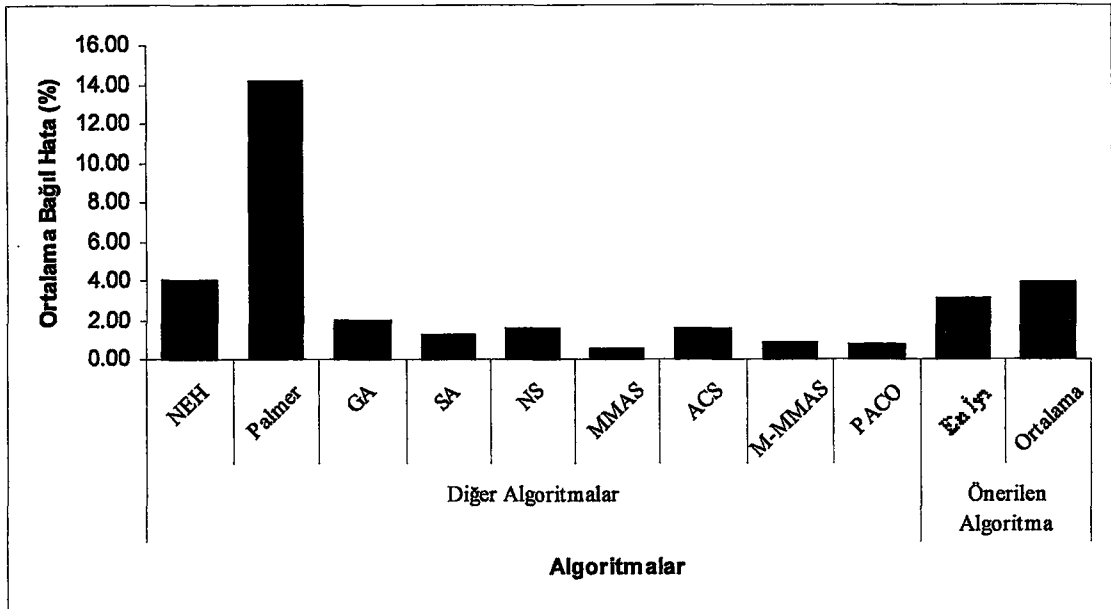
Deneme No	Problem No										Ortalama
	ta011	ta012	ta013	ta014	ta015	ta016	ta017	ta018	ta019	ta020	
1	4.99	7.23	5.95	7.99	8.17	4.37	6.74	4.68	7.78	5.66	6.36
2	6.26	7.53	8.82	7.55	10.22	5.58	6.60	9.10	8.22	5.28	7.52
3	4.74	8.92	4.14	6.97	10.29	4.37	7.35	4.29	9.42	7.04	6.75
4	4.74	8.14	3.88	7.04	9.87	3.94	7.28	4.49	8.85	7.04	6.53
5	4.30	7.66	3.94	6.39	9.23	3.87	6.20	4.10	8.54	6.22	6.04
6	4.17	7.35	3.74	5.52	8.74	4.08	4.38	3.58	7.09	6.10	5.48
7	4.80	7.29	8.22	7.70	8.53	4.29	6.94	4.36	7.85	5.91	6.59
8	7.84	8.44	9.89	7.77	10.78	6.30	6.74	10.27	8.41	5.66	8.21

En kötü değerler için yapılan çalışmalar da 6 numaralı deney en iyi, 8 numaralı deney en kötü sonucu vermiştir.

Özet olarak en iyi, ortalama ve en kötü değerlere göre deneme çalışmalarının tümü için 6 numaralı deney en iyi sonuçları vermiştir. Yani, en uygun parametre değerleri $\alpha = 2$, $\beta = 0.5$, $\rho_l = 0.2$, $\rho_g = 0.2$ ve $q_0 = 0.9$ belirlenmiştir. Bu değerler kullanılarak algoritma Taillard'ın 20x5, 20x10 ve 20x20 problem hacmindeki 30 test problemi üzerinde test edilmiştir. $k_{\max} = 20$ ve $t_{\max} = 5000$ olarak sabitlenmiştir. Her bir deneme 10 kez tekrarlanmış ve böylece 300 adet deneme yapılmıştır. Bu testler için algoritmada yerel arama ve aday liste özellikleri kullanılmamıştır. Deneme sonuçları ve en iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması, Taillard'ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Ek D' de verilmektedir. Algoritmaların performans sonuçları özet olarak Tablo 6.8 ve Şekil 6.7' de özetlenmiştir.

Tablo 6.8 Parametre Analizi Sonrası Algoritma Performans Değerlendirmesi

n	m	Problem No	Diğer Algoritmalar										Önerilen Algoritma			
			NEH	Palmer	GA	SA	NS	MMAS	ACS	M_MMAS	PACO	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}	
20	5	ta001-ta010	3.66	10.81	1.61	1.27	1.46	0.41	1.19	0.76	0.70	1.81	2.63	3.33	5.91	
	10	ta011-ta020	4.60	15.27	2.29	1.71	2.02	0.59	1.70	0.89	0.84	3.88	4.74	5.47	7.92	
	20	ta021-ta030	3.73	16.34	1.95	0.86	1.10	0.41	1.60	0.72	0.72	3.58	4.46	5.26	11.07	
Ortalama=			4.00	14.14	1.95	1.28	1.53	0.47	1.50	0.79	0.76	3.09	3.94	4.68	8.30	



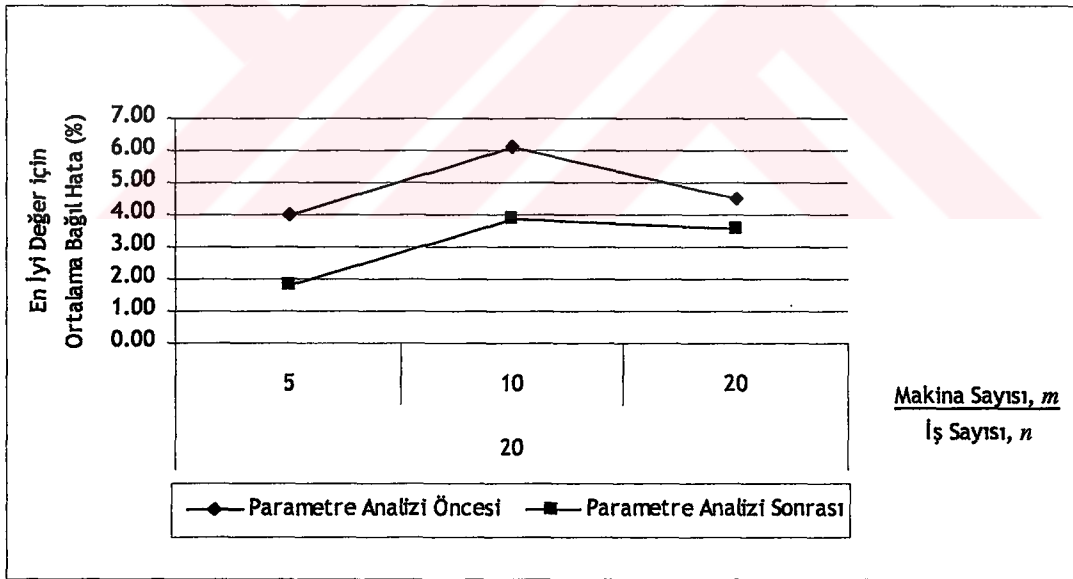
Şekil 6.7 Parametre Analizi Sonrası Algoritma Ortalama Performans Değerlendirmesi

Problem hacmi bazında analiz sonuçları ve sağlanan gelişme Tablo 6.9 ve Şekil 6.8’ de özetlenmiştir.

Algoritma için yapılan bu parametre analizi sonucunda algoritmanın performansı en iyi, ortalama ve en kötü değerlere göre sırasıyla %37, 30 ve 24 oranında arttığı görülmüştür.

Tablo 6.9 Problem Hacmi Bazında Parametre Analizi Öncesi ve Sonrası Ortalama Değerler

n	m	Problem No	Parametre Analizi Öncesi				Parametre Analizi Sonrası			
			Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}
20	5	ta001-ta010	4.01	4.24	4.56	2.35	1.81	2.63	3.33	5.91
	10	ta011-ta020	6.12	7.22	8.02	8.88	3.88	4.74	5.47	7.92
	20	ta021-ta030	4.50	5.43	6.10	11.93	3.58	4.46	5.23	11.07
Ortalama=			4.88	5.63	6.23	7.72	3.09	3.94	4.68	8.30



Şekil 6.8 Problem Hacmi Bazında Parametre Analizinin Etkisi

6.1.3.2 α ve β Parametrelerinin Çözümüne Etkisi

α ve β parametreleri bir sonraki işin seçiminde iz yoğunluğunun ve sezgisel bilginin karşılıklı etkisini belirler. Bu parametrelerin etkisini görmek üzere Taillard’ ın 20x10’boyutundaki 4 örnek test problemi üzerinde testler yapılmıştır. Başlangıç

parametre deęerleri $\rho l = 0.1$, $\rho g = 0.1$ ve $q_0 = 0.9$ olarak alınarak bu iki parametre test edilmiřtir. Ayrıca, $k_{\max} = 20$ ve $t_{\max} = 5000$ olarak sabitlenmiřtir. Bu testlerde yerel arama ve aday liste özellikleri kullanılmamıřtır. Her deneme 10 kez tekrarlamıř ve toplam 1000 adet deneme yapılmıřtır. Deneme sonuçları Ek D' de verilmektedir. En iyi ve ortalama deęerlere göre denemelerin aritmetik ortalaması, Taillard' ın eniyi toplam tamamlanma zamanı üst sınır deęerlerinden ortalama yüzde fark olarak Tablo 6.10, Tablo 6.11 ve Tablo 6.12' de verilmektedir.

Tablo 6.10 α ve β Parametrelerinin En İyi Deęere Göre Çözüm Etkisi

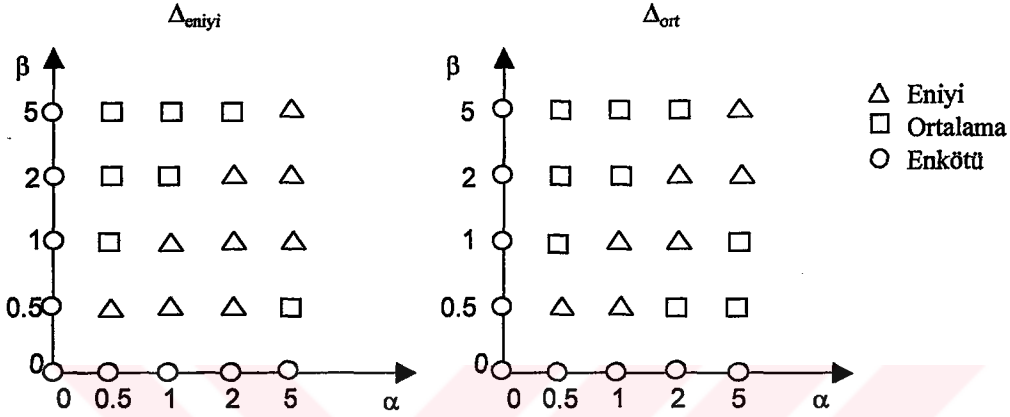
$\beta \backslash \alpha$	0	0,5	1	2	5	Ortalama
0	20.693	12.906	12.629	11.893	12.151	14.054
0,5	13.525	4.537	4.201	4.310	5.118	6.338
1	13.525	5.413	4.469	4.327	3.809	6.309
2	13.525	6.970	5.152	4.230	4.524	6.880
5	13.525	9.499	7.706	5.365	4.389	8.097
Ortalama	14.958	7.865	6.832	6.025	5.998	

Tablo 6.11 α ve β Parametrelerinin Ortalama Deęere Göre Çözüm Etkisi

$\beta \backslash \alpha$	0	0,5	1	2	5	Ortalama
0	20.693	15.282	15.173	15.553	15.021	16.344
0,5	13.525	5.154	5.451	6.516	6.919	7.513
1	13.525	6.514	5.244	5.516	6.297	7.419
2	13.525	7.768	6.247	5.151	5.810	7.700
5	13.525	9.983	8.189	6.869	5.144	8.742
Ortalama	14.958	8.940	8.061	7.921	7.838	

Her iki parametrenin kullanılmadıęı durumlarda ($\alpha = 0$ veya $\beta = 0$) algoritma en kötü performansı göstermiřtir. α parametresinin düşük deęerleri algoritmanın performansı üzerinde negatif bir etkiye sahipken, β parametresinin düşük deęerleri

algoritmanın performansı üzerinde pozitif bir etkiye sahiptir. Diğer yandan, α parametresinin yüksek değerleri algoritmanın performansı üzerinde pozitif bir etkiye sahipken, β parametresinin yüksek değerleri algoritmanın performansı üzerinde negatif bir etkiye sahiptir. Ayrıca parametrelerin her ikisinin kullanımı algoritmanın performansını arttırdığı görülmüştür. Ayrıca Şekil 6.9’ da tüm problemler için ortalama sonuçlar özetlenmiştir.



Şekil 6.9 α ve β Parametrelerinin Etkisi

6.1.3.3 pl ve pg Parametrelerinin Çözüm Etkisi

pl ve pg parametrelerinin çözüm performansı üzerindeki etkisini görmek üzere Taillard’ ın 20x10 boyutundaki 4 örnek test problemi üzerinde testler yapılmıştır. Başlangıç parametre değerleri $\alpha=1$, $\beta=1$ ve $q_0=0.9$ olarak alınarak bu iki parametre test edilmiştir. Ayrıca, $k_{\max}=20$ ve $t_{\max}=5000$ olarak sabitlenmiştir. Bu testler için yerel arama ve aday liste özellikleri kullanılmamıştır. Her bir deneme 10 kez tekrarlanmış, böylece toplam 1000 adet deneme yapılmıştır. Deneme sonuçları EkD’ de verilmektedir. Sırasıyla en iyi ve ortalama değerlere göre denemelerin aritmetik ortalaması, Taillard’ ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Tablo 6.12 ve Tablo 6.13’ de verilmektedir.

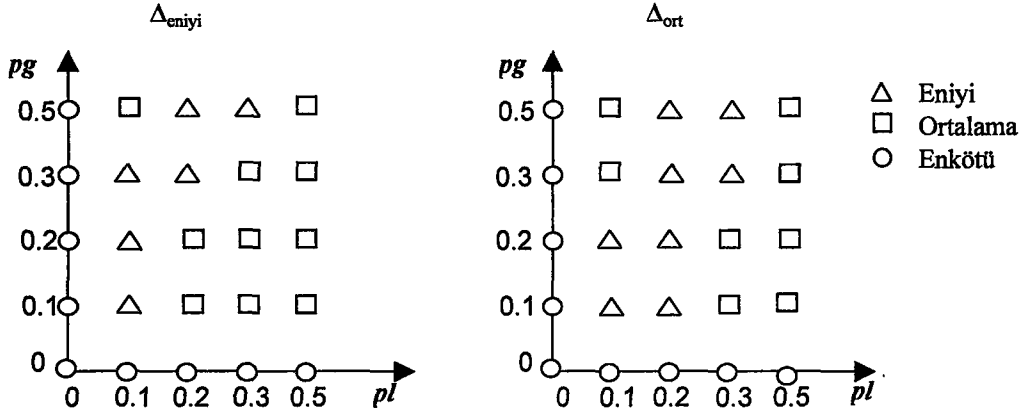
Tablo 6.12 pl ve pg Parametrelerinin En İyi Değere Göre Çözüm Etkisi

$pg \backslash pl$	0	0,1	0,2	0,3	0,5	Ortalama
0	13.525	13.525	13.525	13.525	13.525	13.525
0,1	11.078	4.516	5.041	5.383	6.366	6.477
0,2	11.327	4.025	4.999	5.165	5.872	6.278
0,3	11.123	4.798	4.693	5.168	5.858	6.328
0,5	11.744	5.305	4.656	4.833	5.633	6.434
Ortalama	11.759	6.434	6.583	6.815	7.451	

Tablo 6.13 pl ve pg Parametrelerinin Ortalama Değere Göre Çözüm Etkisi

$pg \backslash pl$	0	0,1	0,2	0,3	0,5	Ortalama
0	13.525	13.525	13.525	13.525	13.525	13.525
0,1	12.550	5.212	5.565	6.223	6.751	7.260
0,2	12.826	5.500	5.421	5.873	6.683	7.261
0,3	12.990	5.815	5.240	5.687	6.464	7.239
0,5	13.041	6.638	5.108	5.519	6.388	7.339
Ortalama	12.986	7.338	6.972	7.365	7.962	

Açıkça görülmektedir ki, pl parametresinin düşük değerleri algoritmanın performansı üzerinde pozitif bir etkiye sahiptir. Ayrıca parametrelerin her ikisinin kullanımı da algoritmanın performansını arttırdığı görülmüştür. Çıkarılan sonuçlar ayrıca Şekil 6.10' da özetlenmiştir.



Şekil 6.10 pl ve pg Parametrelerinin Etkisi

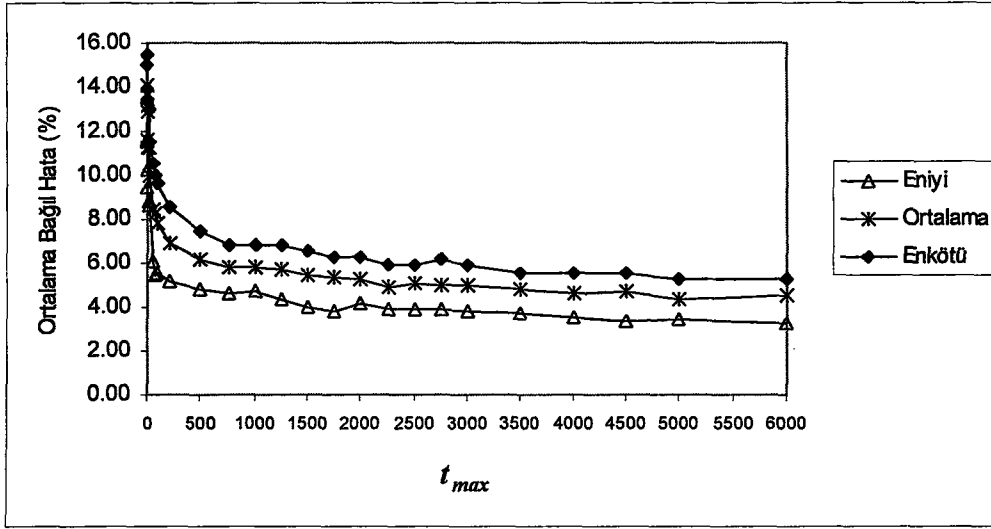
6.1.3.4 İterasyon Sayısı Parametresinin Çözümüne Etkisi

İterasyon sayısının algoritma üzerindeki önemini görmek amacıyla Taillard'ın 20x10 boyutundaki 4 örnek test problemi üzerinde testler yapılmıştır. Başlangıç parametre değerleri $\alpha = 2$, $\beta = 0.5$, $pl = 0.2$, $pg = 0.2$, $q_0 = 0.9$ ve $k_{max} = 20$ olarak sabitlenmiştir. Bu testler için yerel arama ve aday liste özellikleri kullanılmamıştır. Her bir deneme 10 kez tekrarlanmış, böylece toplam 1080 adet daha deneme yapılmıştır. Deneme sonuçları Ek D' de verilmektedir. En iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması, Taillard'ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Tablo 6.14 ve Şekil 6.11' de özetlenmiştir.

Sonuçlara göre, iterasyon sayısı arttıkça algoritmanın performansı da artmaktadır. Ancak, iterasyon sayısı artırıldıkça çözüm elde etmek için gerekli sürede artmaktadır. Testlerden de görüldüğü gibi algoritma, iterasyon sayısı yaklaşık $t_{max}=1000$ ' den sonra iyi değerler üretmeye başlamaktadır. Bu yüzden, çalışma süresini de kısaltmak amacıyla elde edilecek bu küçük iyileştirme göz ardı edilerek $t_{max}=1000$ kullanılabilir. Bunun yanında küçük problemler için iterasyon sayısının daha büyük değerlerde alınması çözüm sürelerini etkilememektedir. Bu nedenle küçük boyutlu problemler için yüksek iterasyon sayısı, daha büyük boyutlu problemler için ise daha düşük iterasyon sayısının tercih edilmesi mümkündür.

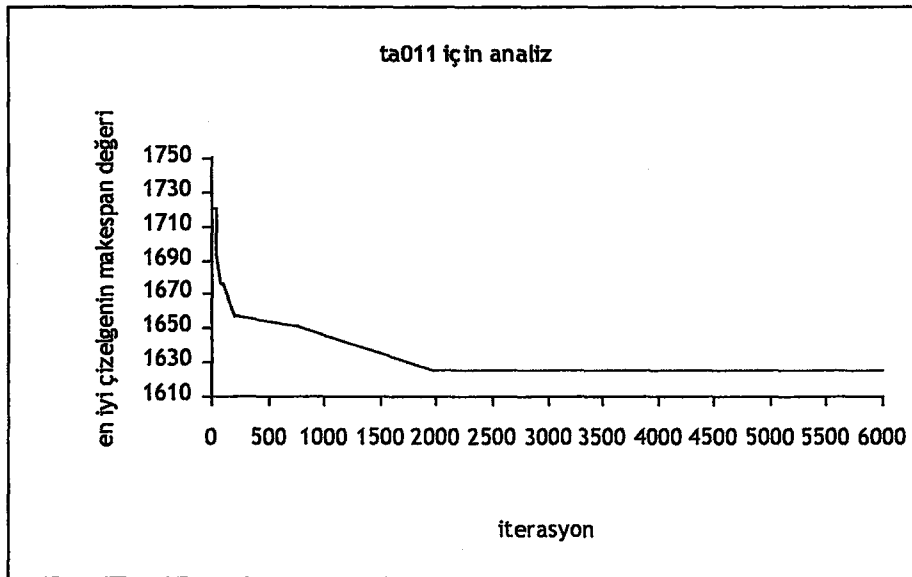
Tablo 6.14 İterasyon Sayısının Çözüm Etkisi

$t_{\max}$	Δ_{eniyi}				Δ_{ort}				$\Delta_{\text{enkötü}}$				Ortalama		
	Problem No				Problem No				Problem No						
	$ta011$	$ta012$	$ta013$	$ta014$	$ta011$	$ta012$	$ta013$	$ta014$	$ta011$	$ta012$	$ta013$	$ta014$	Δ_{eniyi}	Δ_{ort}	$\Delta_{\text{enkötü}}$
1	9.86	15.91	13.30	15.02	11.03	16.36	13.66	15.28	13.15	17.42	14.51	16.69	13.52	14.08	15.44
2	8.72	13.38	12.10	12.41	10.08	14.88	13.41	14.35	13.15	16.40	14.51	15.97	11.65	13.18	15.00
3	8.72	12.96	12.10	12.41	9.80	14.88	12.82	14.26	10.81	16.58	13.30	15.02	11.55	12.94	13.93
5	8.72	12.06	10.23	10.23	9.03	13.39	12.02	12.28	10.11	16.34	13.30	14.22	10.31	11.68	13.49
8	8.72	9.40	9.56	10.09	8.98	12.94	11.05	12.08	10.30	16.40	13.30	13.35	9.44	11.26	13.34
10	7.14	10.37	8.36	9.36	8.48	13.54	11.02	11.92	9.42	16.34	12.10	14.22	8.81	11.24	13.02
20	6.70	9.16	9.29	9.36	8.30	10.71	10.10	11.04	8.72	14.10	10.96	12.48	8.63	10.04	11.57
50	5.75	7.05	5.08	6.60	7.08	9.23	7.75	9.79	8.72	11.21	10.36	11.83	6.12	8.46	10.53
75	5.37	6.69	5.28	4.64	6.96	9.55	7.61	8.97	8.41	11.39	9.29	10.96	5.50	8.27	10.01
100	4.74	7.47	3.94	5.88	6.85	8.77	6.97	8.52	8.41	10.49	9.36	10.38	5.51	7.78	9.66
200	4.30	6.93	4.21	5.22	5.80	8.27	5.68	7.77	6.83	9.40	6.82	11.03	5.17	6.88	8.52
500	4.11	6.81	3.81	4.43	5.54	7.70	5.15	6.49	6.51	8.68	6.15	8.35	4.79	6.22	7.42
750	4.17	6.33	3.54	4.50	5.03	7.46	4.38	6.34	5.94	8.26	5.41	7.76	4.64	5.80	6.84
1000	3.73	7.35	2.74	5.15	4.64	7.91	4.39	6.18	5.75	8.50	6.08	7.04	4.74	5.78	6.84
1250	3.41	6.15	3.34	4.43	4.60	7.43	4.58	6.20	5.50	8.08	5.61	8.13	4.33	5.70	6.83
1500	3.10	6.33	3.21	3.41	4.77	7.31	4.18	5.70	5.56	8.56	4.88	7.11	4.01	5.49	6.53
1750	3.48	6.21	2.81	2.83	4.58	7.12	3.91	5.70	5.25	7.90	4.81	6.97	3.83	5.33	6.23
2000	4.05	5.49	3.48	3.63	4.58	7.08	4.06	5.36	4.99	7.84	4.95	7.26	4.16	5.27	6.26
2250	3.48	5.79	2.74	3.70	4.25	6.81	3.59	5.07	4.74	8.20	4.14	6.46	3.93	4.93	5.89
2500	3.60	5.85	2.74	3.41	4.14	7.02	3.64	5.44	4.87	7.78	4.01	6.89	3.90	5.06	5.89
2750	3.29	5.42	2.74	4.28	4.13	6.79	3.84	5.28	5.31	7.72	4.95	6.60	3.93	5.01	6.14
3000	2.78	5.91	3.34	3.41	3.99	7.00	3.80	5.24	4.80	7.90	4.48	6.53	3.86	5.01	5.93
3500	3.29	5.85	3.21	2.61	4.12	6.75	3.73	4.80	4.74	7.72	4.01	5.88	3.74	4.85	5.59
4000	2.72	5.00	2.74	3.77	3.94	6.16	3.62	4.70	4.42	7.17	4.95	5.59	3.56	4.61	5.53
4500	2.84	4.70	3.01	2.90	3.74	6.55	3.59	4.88	4.17	7.47	4.08	6.31	3.36	4.69	5.51
5000	3.10	4.58	2.74	3.41	3.51	6.05	3.38	4.67	4.17	7.35	3.88	5.81	3.46	4.40	5.30
6000	2.59	4.46	2.74	3.41	3.78	6.12	3.40	4.93	4.61	7.05	3.88	5.66	3.30	4.56	5.30

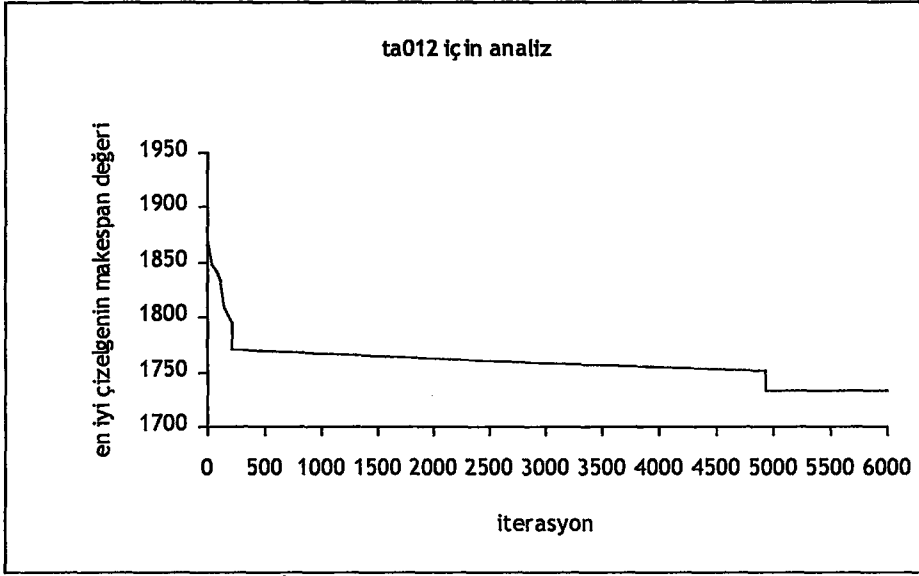


Şekil 6.11 İterasyon Sayısının Çözüme Etkisi

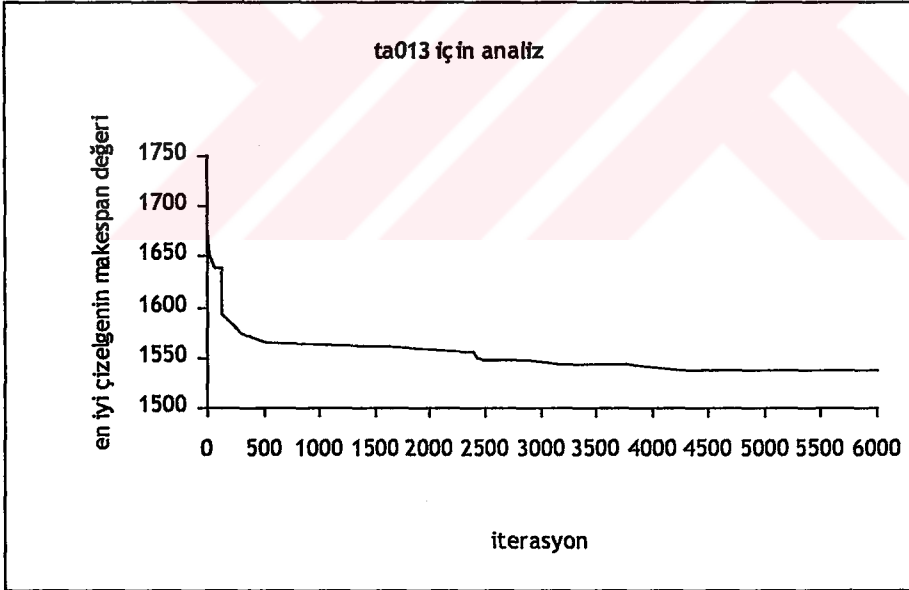
Test problemlerinde (ta011, ta012, ta013 ve ta014) en iyi elde edilen sonuçlar için iterasyon sayısının artışına göre performanstaki gelişmeleri sırasıyla Şekil 6.12, Şekil 6.13, Şekil 6.14 ve Şekil 6.15’ de verilmiştir. İlk problem ta011 için, iterasyon sayısı $t_{max}=2000$ ’ den sonra en iyi değerler üretilirken, ta012 için, iterasyon sayısı $t_{max}=5000$ ’ den sonra, ta013 için, iterasyon sayısı $t_{max}=4250$ ’ den sonra, ta014 için, iterasyon sayısı $t_{max}=1600$ ’ den sonra en iyi değerler üretilmiştir. Ancak tüm problemler için, iterasyon sayısı yaklaşık $t_{max}=500$ ’ den sonra algoritma uygun değerler üretmiştir.



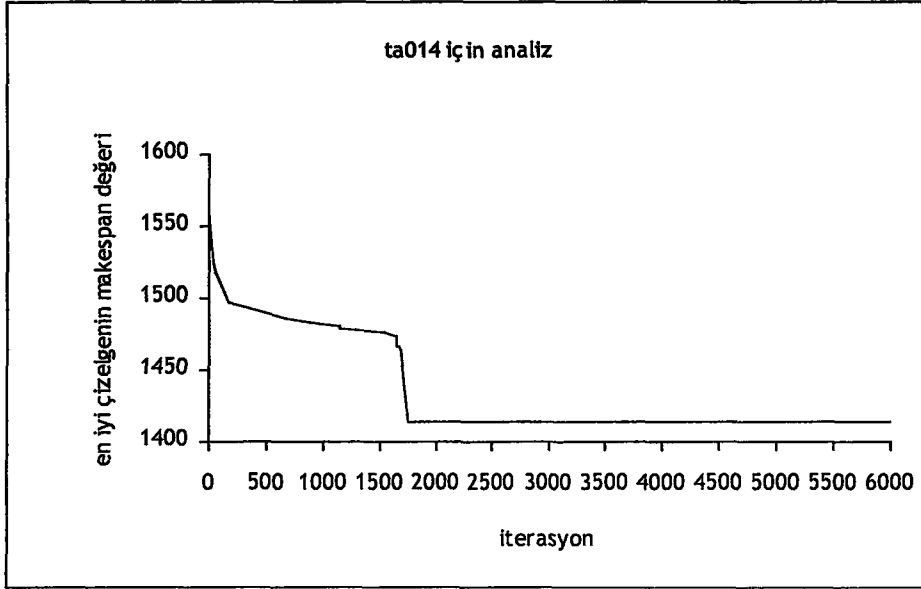
Şekil 6.12 ta011 için İterasyon Sayısı Analizi



Şekil 6.13 ta012 için İterasyon Sayısı Analizi



Şekil 6.14 ta013 için İterasyon Sayısı Analizi



Şekil 6.15 ta014 için İterasyon Sayısı Analizi

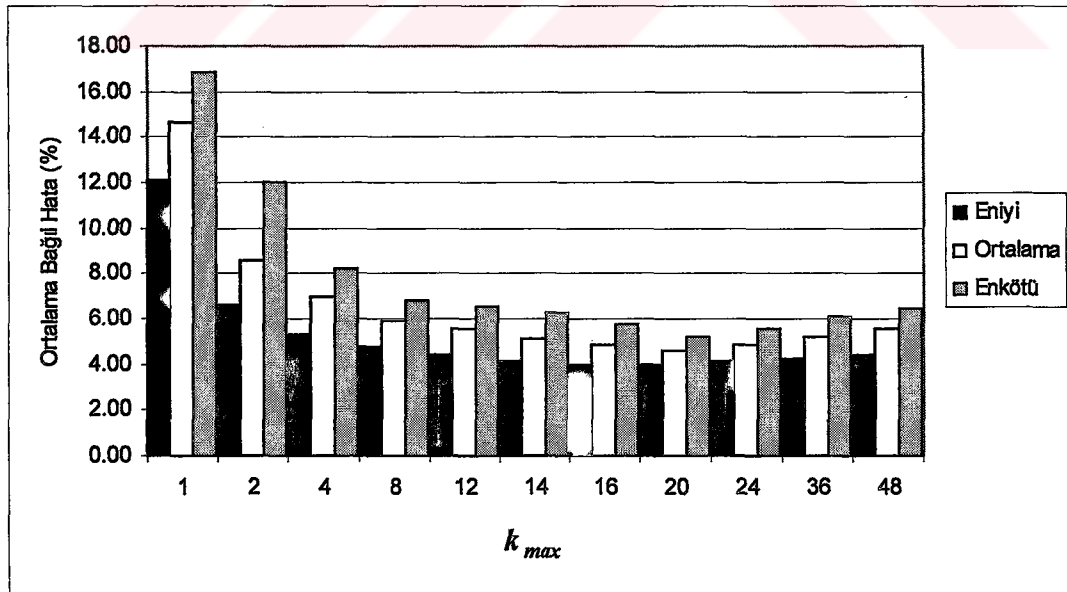
6.1.3.5 Karınca Sayısı Parametresinin Çözüm Etkisi

Karınca sayısının algoritma üzerindeki etkisini görmek amacıyla Taillard'ın 20x10 boyutundaki 4 örnek test problemi üzerinde testler yapılmıştır. Başlangıç parametre değerleri $\alpha = 2$, $\beta = 0.5$, $\rho_l = 0.2$, $\rho_g = 0.2$, $q_0 = 0.9$ ve $t_{\max} = 5000$ olarak sabitlenmiştir. Bu testler için yerel arama ve aday liste özellikleri kullanılmamıştır. Her bir deneme 10 kez tekrarlanmış, böylece toplam 440 adet deneme yapılmıştır. Deneme sonuçları Ek D' de verilmektedir. En iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması, Taillard'ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Tablo 6.15 ve Şekil 6.16' da özetlenmiştir.

Sonuçlara göre, en iyi karınca sayısı düşük olduğu zaman algoritma daha kötü sonuçlar vermiştir. Karınca sayısı $k_{\max} = 20$ değerine kadar karınca sayısı arttıkça algoritmanın performansı iyileşirken, bu değerden sonra performansta kötüleşme görülmüştür. Bundan sonra yapılacak çalışmalarda algoritma için en uygun karınca sayısı değeri olarak $k_{\max} = 20$ kullanılacaktır.

Tablo 6.15 Karınca Sayısının Çözümüne Etkisi

$k_{\max}$	Δ_{eniyi}				Δ_{ort}				$\Delta_{\text{enkötü}}$				Ortalama		
	Problem No				Problem No				Problem No						
	$ta011$	$ta012$	$ta013$	$ta014$	$ta011$	$ta012$	$ta013$	$ta014$	$ta011$	$ta012$	$ta013$	$ta014$	Δ_{eniyi}	Δ_{ort}	$\Delta_{\text{enkötü}}$
1	9.36	13.38	13.30	12.26	12.28	15.79	15.26	15.26	13.91	17.42	18.45	17.71	12.08	14.65	16.87
2	5.50	6.21	8.02	6.60	8.03	7.78	9.48	8.81	13.15	9.28	15.57	10.01	6.58	8.53	12.00
4	4.68	5.12	5.01	6.24	6.36	6.59	7.43	7.55	7.46	7.66	8.96	8.85	5.26	6.98	8.23
8	4.49	4.76	5.41	4.28	5.23	6.03	6.29	5.99	6.26	6.99	6.75	7.11	4.74	5.88	6.78
12	3.86	4.58	4.41	4.72	4.66	5.49	6.13	5.90	5.75	6.39	7.09	6.89	4.39	5.54	6.53
14	3.35	4.94	4.08	4.35	4.54	5.67	5.03	5.20	5.56	6.33	6.75	6.39	4.18	5.11	6.26
16	3.03	5.06	3.21	4.64	4.01	5.78	4.16	5.29	4.74	7.17	4.95	6.10	3.99	4.81	5.74
20	3.16	6.15	2.87	3.63	3.64	6.59	3.64	4.61	4.17	6.87	4.14	5.59	3.95	4.62	5.19
24	3.29	6.63	2.87	3.63	4.05	6.97	3.52	4.77	4.55	7.47	4.01	6.17	4.11	4.83	5.55
36	3.48	6.51	3.21	3.70	4.91	7.07	3.79	5.13	5.75	7.72	4.55	6.39	4.22	5.22	6.10
48	4.99	6.09	3.21	3.41	5.73	6.81	3.82	5.71	6.83	7.59	4.55	6.82	4.43	5.52	6.45



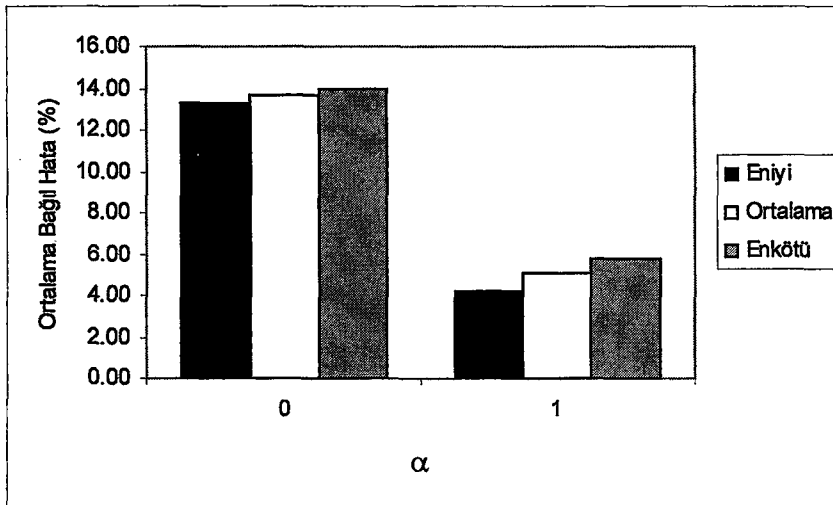
Şekil 6.16 Karınca Sayısının Çözümüne Etkisi

6.1.4 Karıncalar Arasındaki İşbirliğinin Çözümüne Etkisi

Karıncalar arasındaki iletişimin çözüm performansı üzerindeki etkisini görmek üzere denemeler yapılmıştır. Taillard' ın 20x10 boyutundaki 4 örnek test problemi üzerinde testler yapılmıştır. Başlangıç parametre değerleri $\alpha = 2$, $\beta = 0.5$, $\rho_l = 0.2$, $\rho_g = 0.2$, $q_0 = 0.9$, $k_{\max} = 20$ ve $t_{\max} = 5000$ olarak sabitlenmiştir. Bu testler için yerel arama ve aday liste özellikleri kullanılmamıştır. Her bir deneme 10 kez tekrarlanmış, böylece toplam 80 adet daha deneme yapılmıştır. Deneme sonuçları EkE' de verilmektedir. En iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması, Taillard' ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Tablo 6.16 ve Şekil 6.17' de özetlenmiştir.

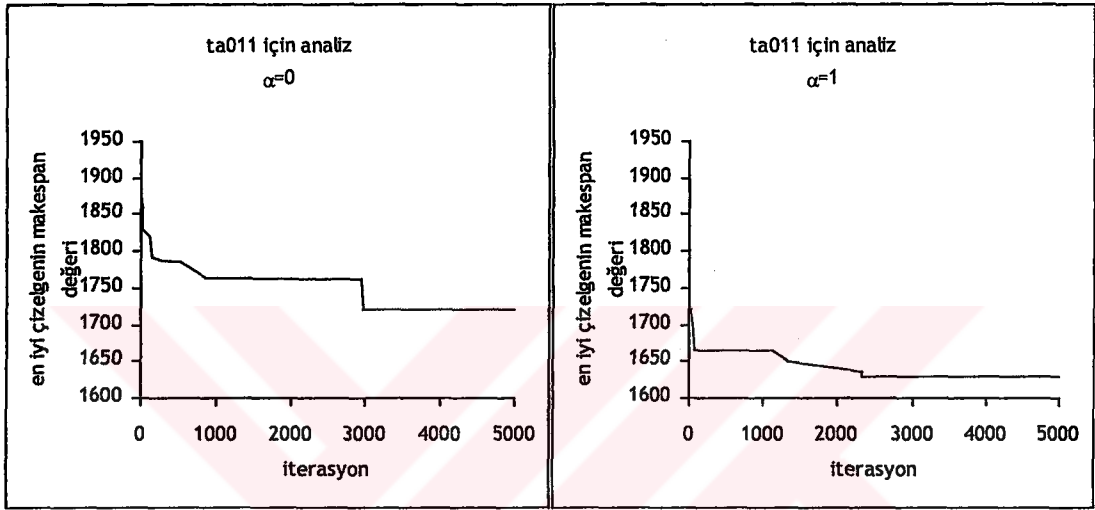
Tablo 6.16 Karıncalar Arası İşbirliğinin Çözümüne Etkisi

Problem No	Δ_{eniyi}		Δ_{ort}		$\Delta_{\text{enkötü}}$	
	α		α		α	
	0	1	0	1	0	1
ta011	8.79	2.91	10.28	3.96	11.63	4.74
ta012	15.91	6.39	15.91	7.08	15.91	7.53
ta013	13.30	2.74	13.30	3.30	13.30	3.94
ta014	15.11	4.87	15.11	6.01	15.11	6.97
Ortalama=	13.28	4.23	13.65	5.09	13.99	5.80

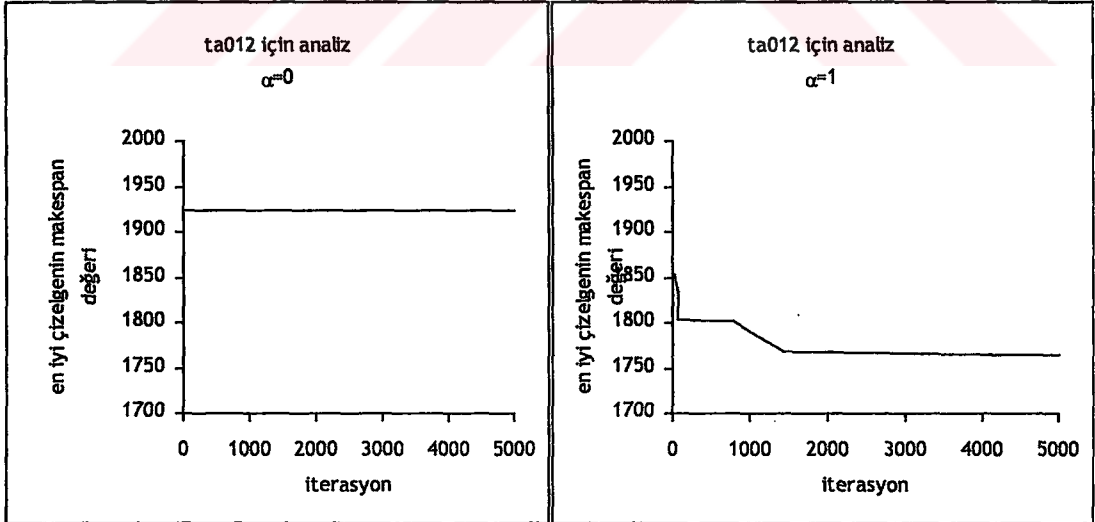


Şekil 6.17 Karıncalar Arası İşbirliğinin Çözümüne Etkisi

Sonuçlar, birçok karınca kullanımı bir sinerji etkisi yarattığını göstermişti. Burada işbirliği yapmayan ($\alpha = 0$) ve işbirliği yapan ($\alpha = 1$) karıncaların olduğu durumlar karşılaştırılmıştır. Sonuçlar, işbirliği olması halinde algoritmanın performansının ciddi şekilde arttığını göstermiştir. Test problemlerinde elde edilen en iyi sonuçlar için iterasyon sayısının artışına göre performanstaki gelişmeler sırasıyla Şekil 6.18, Şekil 6.19, Şekil 6.20 ve Şekil 6.21’ de verilmiştir.

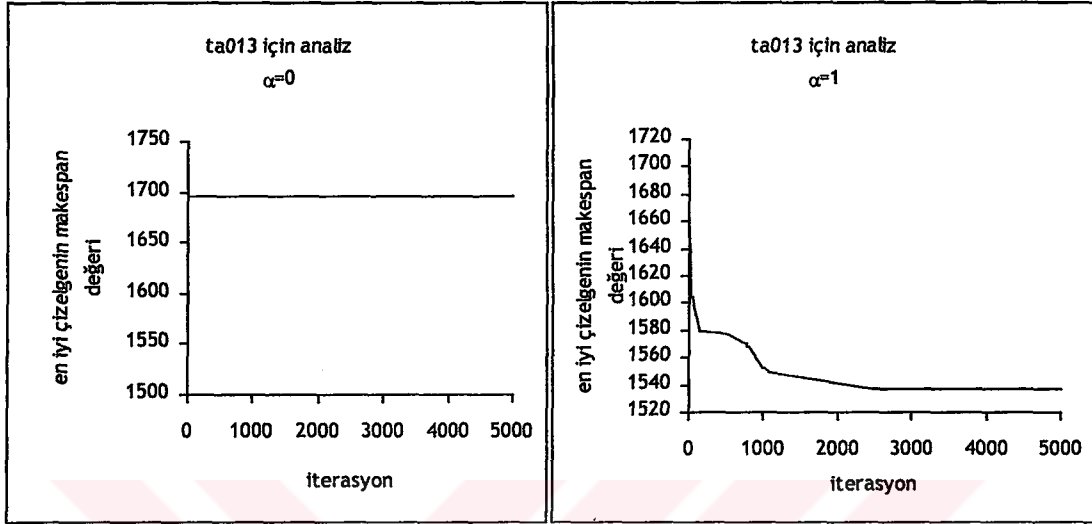


Şekil 6.18 ta011 için Karıncalar Arası İşbirliği

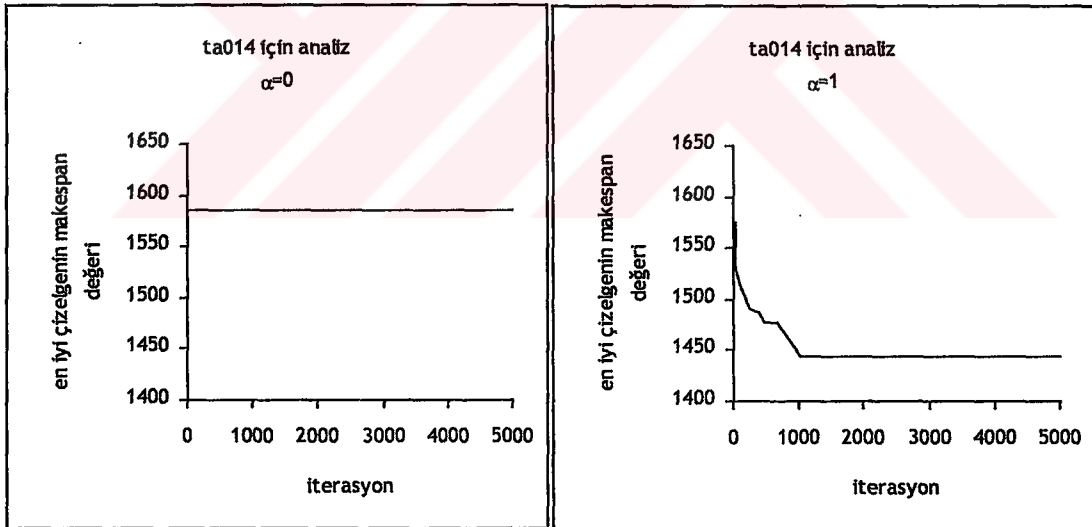


Şekil 6.19 ta012 için Karıncalar Arası İşbirliği

İlk problem ta011 için, karıncalar arasında iletişimin problemin en iyi değerinde %5, iyileşme sağlarken, ta012 için %8, ta013 için %9, ta014 için %14 iyileşme sağladığı görülmüştür. Ayrıca karınca kolonisi kullanımının problem çözümünü daha fazla çeşitlendirerek iyileştirdiği görülmüştür.



Şekil 6.20 ta013 için Karıncalar Arası İşbirliği



Şekil 6.21 ta014 için Karıncalar Arası İşbirliği

6.1.5 Geçiş Olasılığının Çözümüne Etkisi

Burada karınca algoritmaları için kullanılan aşağıdaki geçiş olasılıklarının çözüm performansına olan etkisi karşılaştırılacaktır:

Rasgele-oran Geçiş Kuralı (R-Prop): Bu seçim kuralı karınca sistemi algoritması tarafından kullanılan seçim kuralıdır. Bu kural daha önce Denklem(3.5) ile tanımlanmıştır.

Pseudo-rasgele-oran Geçiş Kuralı (PR-Prop): Bu seçim kuralı karınca koloni sistemi algoritması tarafından kullanılan seçim kuralıdır. Bu kural daha önce Denklem(5.4) ve (5.5) ile tanımlanmıştır.

Pheromone Toplama Kuralı (Psum): Bu seçim kuralı Merkle ve Middendorf (2000, 2003) tarafından tek makine çizelgeleme problemi için önerilen kuraldır. Bu kural, herhangi bir $l > j$ pozisyona iş ataması yaparken, daha sonra verilecek kararların etkisini içeren bir iz miktarı tanımlar. Bu etkiyi güvence altına alan basit bir yol, bir j pozisyonu için bir iş hakkında karar verirken, iz matrisindeki ilk satırdan j satırına kadar olan her işe ait iz değerlerinin toplanmasıdır. Buna göre bir çizelge oluşturulurken, j pozisyonu için karınca, q_0 olasılıkla aşağıdaki ifadeyi en büyüklenecek i işini seçer:

$$\sum_{q=1}^j \tau(i, q)]^\alpha [\eta(i, j)]^\beta \quad (6.1)$$

ve $1 - q_0$ olasılıkla aşağıdaki olasılık dağılımına göre i işini seçer:

$$p_k(i, j) = \frac{(\sum_{q=1}^j \tau(i, q)]^\alpha [\eta(i, j)]^\beta}{\sum_{u \in S_k(i)} (\sum_{q=1}^j \tau(i, q)]^\alpha [\eta(u, j)]^\beta} \quad (6.2)$$

Relative Pheromone Değerlendirme Kuralı (R-Eval): Bu seçim kuralı Merkle ve Middendorf (2001a, 2001b) tarafından tek makine çizelgeleme problemi için önerilen kuraldır. Bu kural, iz matrisindeki her bir satırdaki pheromone miktarlarını doğrudan kullanmak yerine, normalize edilmiş değerleri kullanır. Buna göre bir çizelge oluşturulurken, j pozisyonu için karınca, q_0 olasılıkla aşağıdaki ifadeyi en büyükleyecek i işini seçer:

$$\tau(i, j) *^\alpha \eta(i, j)^\beta \quad (6.3)$$

ve $1 - q_0$ olasılıkla aşağıdaki olasılık dağılımına göre i işini seçer:

$$p_k(i, j) = \frac{\tau(i, j)^{\alpha} \eta(i, j)^{\beta}}{\sum_{u \in S_k(i)} \tau(i, j)^{\alpha} \eta(u, j)^{\beta}}$$

burada; (6.4)

$$\tau(i, j)^{\alpha} = \left(\frac{\sum_{q=1}^n \tau(i, q)}{\sum_{q=1}^j \tau(i, q)} \right)^{\gamma} \cdot \tau(i, j) \quad (6.5)$$

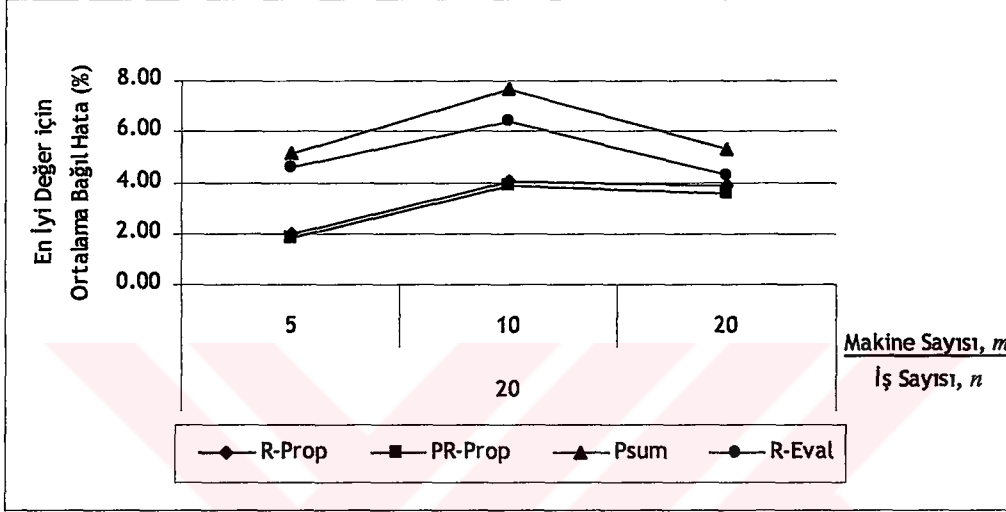
$0 \leq \gamma \leq 1$, normalizasyonun şiddetini belirleyen bir parametredir.

Geçiş olasılığı kurallarının çözüm performansı üzerindeki etkisini görmek üzere Taillard' ın 20x5, 20x10 ve 20x20 boyutundaki 30 test problemi üzerinde denemeler yapılmıştır. Başlangıç parametre değerleri $\alpha = 2$, $\beta = 0.5$, $\rho_l = 0.2$, $\rho_g = 0.2$, $q_0 = 0.9$, $k_{\max} = 20$ ve $t_{\max} = 5000$ olarak sabitlenmiştir. Bu testler için algorithmada yerel arama ve aday liste özellikleri kullanılmamıştır. Her bir deneme 10 kez tekrarlanmış, böylece toplam 1200 adet deneme yapılmıştır. Deneme sonuçları ve en iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması Taillard' ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Ek F' de verilmektedir. Algoritmaların performans sonuçları problem hacmi bazında Tablo 6.17 ve Şekil 6.22' de özetlenmiştir.

Tablo 6.17 Problem Hacmi Bazında Geçiş Olasılıklarının Çözümüne Etkisi

n	m	Problem No	R-Prop				PR-Prop				Psum				R-Eval			
			Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}
20	5	ta001-ta010	1.98	2.64	3.18	5.01	1.81	2.63	3.33	5.91	5.14	5.66	6.21	4.81	4.57	5.04	5.75	4.66
	10	ta011-ta020	4.01	4.91	5.66	8.43	3.88	4.74	5.47	7.92	7.69	8.96	10.44	12.55	6.43	7.67	9.14	13.82
	20	ta021-ta030	3.85	4.54	5.08	8.84	3.58	4.46	5.23	11.07	5.30	6.78	8.40	21.11	4.29	5.32	6.32	14.93
Ortalama=			3.28	4.03	4.64	7.43	3.09	3.94	4.68	8.30	6.04	7.14	8.35	12.82	5.10	6.01	7.07	11.14

Deneme sonuçlarına göre geçiş olasılığı kurallarının çözüm performansı üzerinde karınca koloni sistemi algoritması seçim kuralı (PR-Prop) en iyi ve pheromone toplama kuralı (Psum) en kötü sonuçları vermiştir. Karınca sistemi algoritması seçim kuralı (R-Prop), karınca sistemi algoritması seçim kuralına yakın bir performansa sahipken, relative pheromone değerlendirme kuralı (R-Eval) sonuçları ise pheromone toplama kuralına yakın bir performansa sahip olduğu görülmüştür.



Şekil 6.22 Problem Hacmi Bazında Geçiş Olasılıklarının Çözümüne Etkisi

6.1.6 Pheromone İzi Güncelleme Kuralının Çözümüne Etkisi

Bir çok pheromone güncelleme kuralı vardır. Pheromone izi global veya yerel olarak güncellenebilir. Ayrıca, hangi karıncaların güncelleme yapmak için seçilmeleri de önemlidir. Aşağıda test edilecek çeşitli pheromone güncelleme kuralları (PGK) tanımlanmıştır.

Karınca Sistemi Pheromone İzi Güncelleme Kuralı (AS-PGK): Bu güncelleme kuralı karınca sistemi algoritması tarafından kullanılan kuraldır. Bu kurala göre karıncalar turlarını tamamladıktan sonra, geçtikleri yollar üzerine $1/L^k$ miktar pheromone bırakırlar. Burada, L^k , k karıncasının oluşturduğu turun uzunluğudur. Karıncaların bıraktığı pheromone miktarı aşağıdaki kurala göre güncellenir:

$$\tau(i, j) = \rho\tau(i, j) + \Delta\tau(i, j) \quad (6.6)$$

burada,

$$\Delta\tau(i, j) = \sum_{k=1}^m \Delta\tau_k(i, j) \quad (6.7)$$

$$\Delta\tau_k(i, j) = 1 / L^k \quad (6.8)$$

Elitist Stratejisi Pheromone İzi Güncelleme Kuralı (AS_{elit}-PGK): Karınca sistemini güçlendirmek üzere kullanılan elitist stratejisi güncelleme kuralıdır. Bu strateji algoritmanın çalışmaya başladığı andan itibaren bulunan en iyi çözüme (T^{gb} , burada gb , global-en iyiyi temsil etmektedir) ek bir ağırlık verilmesinden oluşur. Daha açık bir ifade ile, her bir zamanda pheromone iz miktarı güncellenirken, en iyi tura ait yollar ek bir miktar pheromone alır. Karıncaların bıraktığı pheromone miktarı aşağıdaki kurala göre güncellenir:

$$\tau(i, j) = \rho\tau(i, j) + \Delta\tau(i, j) + \Delta\tau^*(i, j) \quad (6.9)$$

burada,

$$\Delta\tau(i, j) = \sum_{k=1}^m \Delta\tau_k(i, j), \quad (6.10)$$

$$\Delta\tau_k(i, j) = 1 / L^k, \quad (6.11)$$

$$\Delta\tau^*(i, j) = \begin{cases} e \cdot \frac{1}{L^{gb}} & \text{eger } (i, j) \in T^{gb} \\ 0 & \text{diger} \end{cases} \quad (6.12)$$

Burada, L^{gb} , T^{gb} , nin uzunluğu ve e , elitist karınca sayısıdır.

Karınca Koloni Sistemi Pheromone İzi Güncelleme Kuralı: Bu güncelleme kuralı karınca koloni sistemi pheromone izi güncelleme kuralıdır. Bu kural daha önce Denklem(5.6) ve Denklem(5.7) ile tanımlanmıştır. Burada global güncelleme kuralında, global-eniyi L_{gb} (ACS_{gb}-PGK) ve iterasyon-eniyi L_{ib} (ACS_{ib}-PGK) kullanımı ayrı ayrı değerlendirilmiştir.

Ant-Q Pheromone İzi Güncelleme Kuralı (Ant-Q-PGK): Bu güncelleme kuralı Ant-Q sistemi tarafından kullanılan güncelleme kuralıdır. Ant-Q algoritmasında, pheromone izi güncelleme işlemi, bir sonraki aşamanın tahmini değerine göre yapılmaktadır:

$$\tau(i, j) := (1 - \varphi) \cdot \tau(i, j) + \varphi \cdot \gamma \cdot \max_{z \in S_k(j)} \tau(j, z) \quad (6.13)$$

Burada, γ , discount faktörüdür.

Max-Min Karınca Sistemi Pheromone İzi Güncelleme Kuralı (MMAS-PGK): Bu güncelleme kuralı max-min karınca sistemi tarafından kullanılan kuralıdır. MMAS algoritmasında, her iterasyon sonunda tüm karıncalar bir çözüm oluşturduktan sonra, sadece tek bir karıncanın iz güncellemesine izin verilir. Güncelleme kuralı şu şekildedir:

$$\tau(i, j) := \rho \cdot \tau(i, j) + \Delta \tau_{ij}^{best} \quad (6.14)$$

burada,

$$\Delta \tau_{ij}^{best} = 1 / L^{best} \quad (6.15)$$

Eğer izlerin güncellenmesi için iterasyon-en iyi karınca seçilirse, tur uzunluğu L_{ib}^{best} , global-en iyi karınca seçilirse, tur uzunluğu L_{gb}^{best} göz önüne alınır.

Sıralamaya Dayalı Karınca Sistemi Pheromone İzi Güncelleme Kuralı (AS_{rank}-PGK): Bu güncelleme kuralı sıralamaya dayalı karınca sistemi tarafından kullanılmaktadır. Karıncalar, tur uzunluklarına göre ($L^1(t) \leq L^2(t) \leq \dots \leq L^m(t)$) sıralanır ve karıncanın depolayabileceği pheromone miktarı karıncanın r sırasına göre ağırlıklandırılır. Sadece her iterasyonda, $(w - 1)$ karıncanın pheromone depolamasına izin verilir. En güçlü geri beslemeyi veren global-en iyi çözüme, w ağırlığı verilir. O anki iterasyonun en iyi r karıncası, verilen $\max\{0, w - r\}$ ağırlığı ile pheromone güncellemeye katkıda bulunur. Böylece güncelleme kuralı şöyle ifade edilir:

$$\tau(i, j) := (1 - \rho) \cdot \tau(i, j) + \sum_{r=1}^{w-1} (w - r) \cdot \Delta \tau^r(i, j) + w \cdot \Delta \tau^*(i, j) \quad (6.16)$$

burada,

$$\Delta \tau^r(i, j) = 1/L^r, \quad (6.17)$$

$$\Delta \tau^*(i, j) = 1/L^{gb}. \quad (6.18)$$

Pheromone izi güncelleme kuralının çözüm performansı üzerindeki etkisini görmek üzere Taillard' ın 20x5, 20x10 ve 20x20 boyutundaki 30 test problemi üzerinde denemeler yapılmıştır. Başlangıç parametre değerleri için:

Karınca sisteminde $q_0 = 0.9, \alpha = 2, \beta = 0.5, \rho = 0.8$,

Elitist stratejisinde $e = 20, q_0 = 0.9, \alpha = 2, \beta = 0.5, \rho = 0.8$,

ACS global-eniyi stratejisinde $q_0 = 0.9, \alpha = 2, \beta = 0.5, \rho_l = 0.2, \rho_g = 0.2$,

ACS iterasyon-eniyi stratejisinde $q_0 = 0.9, \alpha = 2, \beta = 0.5, \rho_l = 0.2, \rho_g = 0.2$,

ANT-Q sisteminde $\gamma = 0.3, q_0 = 0.9, \alpha = 2, \beta = 0.5, \rho = 0.8, t_{\max} = 3000$,

MMAS sisteminde $q_0 = \frac{n-4}{4}, \alpha = 2, \beta = 0.5, \rho = 0.8, \tau_{\max} = \frac{1}{1-\rho} \cdot \frac{1}{L^*}, \tau_{\max} = \frac{\tau_{\max}}{5}$,

AS_{rank} sisteminde $w = 20, q_0 = 0.9, \alpha = 2, \beta = 0.5, \rho = 0.8$,

$k_{\max} = 20$ ve $t_{\max} = 5000$ olarak sabitlenmiştir. Bu testler için algoritmada yerel arama ve aday liste özellikleri kullanılmamıştır. Her bir deneme 10 kez tekrarlanmış, böylece toplam 2100 adet daha deneme yapılmıştır. Deneme sonuçları ve en iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması, Taillard' ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Ek G' de verilmektedir. Algoritmaların performans sonuçları problem hacmi bazında Tablo 6.18, Tablo 6.19 ve Şekil 6.23' de özetlenmiştir.

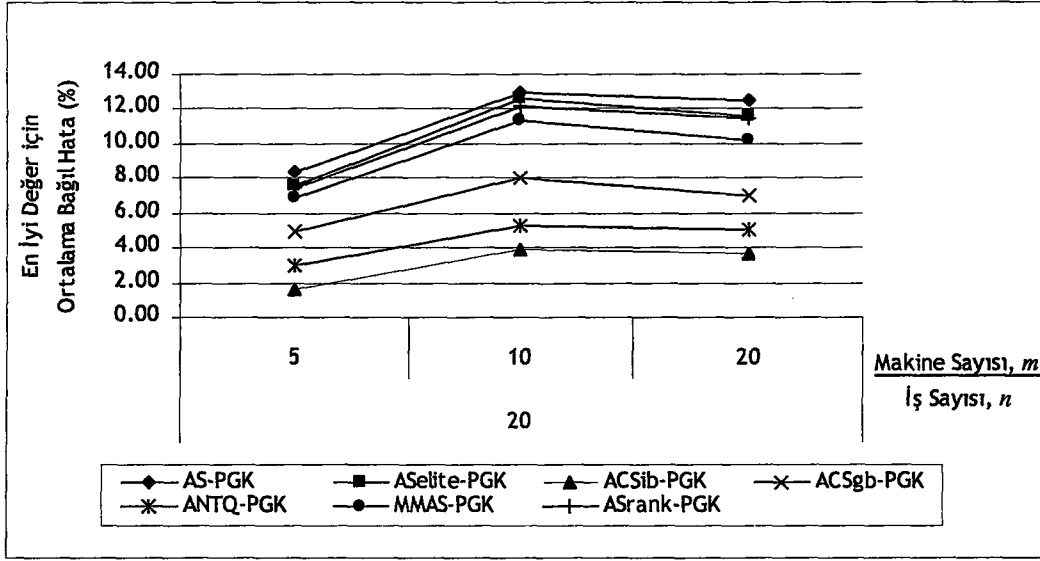
Tablo 6.18 Problem Hacmi Bazında Pheromone İzi Güncelleme Kurallarının (AS, AS_{elite}, ACS_{ib}, ACS_{gb}) Çözüm Etkisi

<i>n</i>	<i>m</i>	Problem No	AS-PGK				AS _{elite} -PGK				ACS _{ib} -PGK				ACS _{gb} -PGK			
			Δ_{eniyl}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyl}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyl}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyl}	Δ_{ort}	Δ_{enkt}	Δ_{std}
20	5	ta001-ta010	8.37	8.79	9.67	6.60	7.59	8.28	9.03	7.14	1.66	2.69	3.26	6.74	4.94	6.66	7.47	10.85
	10	ta011-ta020	13.00	13.54	14.77	10.73	12.60	13.36	14.66	12.13	3.92	4.90	5.77	8.88	7.99	9.75	11.63	18.68
	20	ta021-ta030	12.55	13.16	14.14	14.87	11.59	12.60	14.04	20.02	3.66	4.46	5.01	9.57	6.95	8.98	10.55	25.75
Ortalama=			11.31	11.83	12.86	10.73	10.60	11.41	12.57	13.10	3.08	4.02	4.68	8.40	6.63	8.46	9.88	18.42

Tablo 6.19 Problem Hacmi Bazında Pheromone İzi Güncelleme Kurallarının (ANTQ, *MMAS*, AS_{rank}) Çözüm Etkisi

<i>n</i>	<i>m</i>	Problem No	ANTQ-PGK				<i>MMAS</i> -PGK				AS _{rank} -PGK			
			Δ_{eniyl}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyl}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyl}	Δ_{ort}	Δ_{enkt}	Δ_{std}
20	5	ta001-ta010	3.02	4.36	7.45	8.18	6.90	7.75	8.32	6.44	7.45	8.18	9.70	9.97
	10	ta011-ta020	5.29	7.60	12.21	13.18	11.41	12.35	13.70	12.23	12.21	13.18	14.54	13.18
	20	ta021-ta030	5.00	7.06	11.42	12.48	10.26	11.72	12.97	20.22	11.42	12.48	13.99	21.05
Ortalama=			4.44	6.34	8.16	19.47	9.52	10.61	11.66	12.96	10.36	11.28	12.74	14.73

Yapılan testlerde en iyi pheromone izi güncelleme kuralı karınca koloni sistemi pheromone izi güncelleme kuralı olmuştur. Ant-Q pheromone izi güncelleme kuralı da ikinci en iyi sonucu vermiştir. Karınca sistemi pheromone izi güncelleme kuralı, elitist stratejisi pheromone izi güncelleme kuralı, max-min karınca sistemi pheromone izi güncelleme kuralı ve sıralamaya dayalı karınca sistemi pheromone izi güncelleme kuralı da kötü çözümler üretmişlerdir. Burada karınca koloni sistemi pheromone izi güncelleme kuralı için uygulanan iki global güncelleme kuralından iterasyon-eniyi stratejisi en iyi sonucu verirken, global-eniyi stratejisi daha kötü sonuçlar vermiştir.



Şekil 6.23 Problem Hacmi Bazında Pheromone İzi Güncelleme Kurallarının Çözümüne Etkisi

6.1.7 Sezgisel Bilginin Önemi ve Çözümüne Etkisi

Sezgisel bilgi kurallarının algoritmanın performansı üzerine etkileri araştırılacaktır. Bunun için aşağıda sezgisel bilgi olarak kullanılabilir bazı kurallar tanımlanmıştır.

En Kısa İşlem Süresi Toplamı (TSPT): Her i işi için, $T_i = \sum_{j=1}^m t_{i,j}$ toplamı bir sıralama ölçüsü olarak düşünülebilir. Buna göre i ($i=1,2,...,n \cup N$) işi ile u ($u=1,2,...,n$) işi arasındaki uzaklık şöyle tanımlanır:

$$\delta(i,u) = 1/\eta(i,u) \quad (6.19)$$

burada,

$$\eta(i,u) = 1/T_u \quad (i \neq u). \quad (6.20)$$

i ($i=1,2,...,n \cup N$) işi ile u ($u=1,2,...,n$) işi arasındaki uzaklık, u işinin işlem süresi toplamıdır.

Gupta' nın İş İndeksi (GJI): Gupta iş indeksi, komşu makineler üzerinde bir işin işlem zamanları toplamını karşılıklı olarak en küçükler.

Bu iş indeksi revize edilirse i ($i=1,2,...,n \cup N$) işi ile u ($u=1,2,...,n$) işi arasındaki uzaklık şöyle tanımlanır:

$$\delta(i, u) = 1/\eta(i, u) \quad (6.21)$$

burada,

$$\eta(i, u) = s_u - \min_u \{\eta(i, u)\} + 1 \quad (i \neq u). \quad (6.22)$$

i ($i=1,2,...,n \cup N$) işi ile u ($u=1,2,...,n$) işi arasındaki uzaklık, u işi iş indeksine eşittir.

Entegre Edilmiş Yasak Arama Tekniği ile Yeni bir Çözüm Oluşturan Sıralama Problemi (Sequencing Problem Involving a Resolution by Integrated Taboo Search Techniques-SPIRIT): Widmer ve Hertz (1989) tarafından kullanılan iki iş arası uzaklık (distance between two jobs) aşağıdaki şekilde tanımlanır:

$$d_{ij} = t_{i1} + \sum_{k=2}^m (m-k) * |t_{ik} - t_{j,k-1}| + t_{jm} \quad (6.23)$$

Buna göre i ($i=1,2,...,n \cup N$) işi ile u ($u=1,2,...,n$) işi için sezgisel bilgi şöyle tanımlanır:

$$\eta(i, u) = \frac{1}{d_{iu}} \quad (i \neq u) \quad (6.24)$$

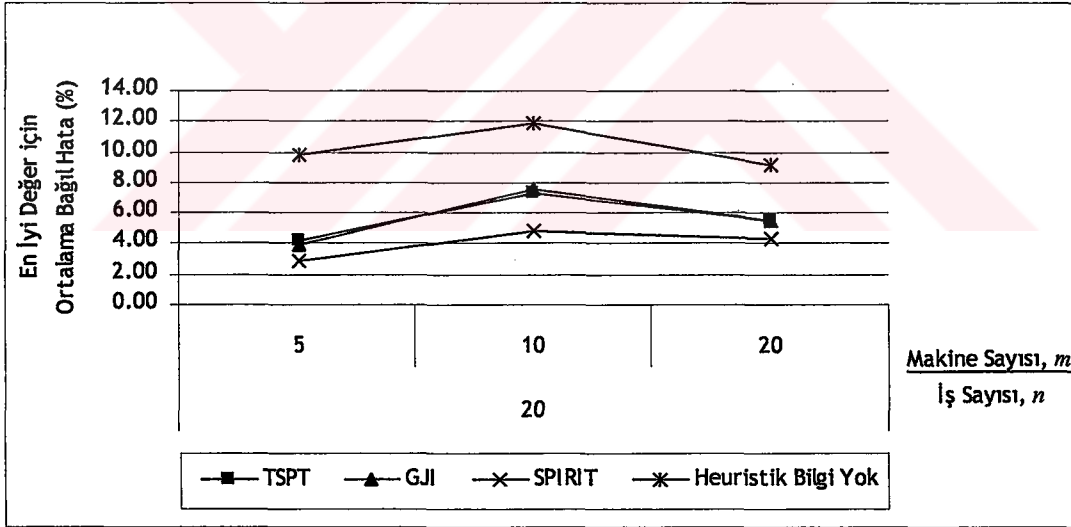
Sezgisel bilginin çözüm performansı üzerindeki etkisini görmek üzere Taillard' ın 20x5, 20x10 ve 20x20 boyutundaki 30 test problemi üzerinde denemeler yapılmıştır.

Başlangıç parametre değerleri $\alpha = 2$, $\beta = 0.5$, $\rho l = 0.2$, $\rho g = 0.2$, $q_0 = 0.9$, $k_{\max} = 20$ ve $t_{\max} = 5000$ olarak sabitlenmiştir. Bu testler için algoritmada yerel arama ve aday liste özellikleri kullanılmamıştır. Her bir deneme 10 kez tekrarlanmış, böylece toplam 1200 adet daha deneme yapılmıştır. Deneme sonuçları ve en iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması, Taillard' ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Ek H' da verilmektedir. Algoritmaların performans sonuçları problem hacmi bazında Tablo 6.20 ve Şekil 6.24' de özetlenmiştir.

Tablo 6.20 Problem Hacmi Bazında Sezgisel Bilginin Çözümüne Etkisi

n	m	Problem No	TSPT				GJI				SPIRIT				Sezgisel Bilgi Yok			
			Δ_{enlyi}	Δ_{ort}	$\Delta_{enkı}$	Δ_{std}	Δ_{enlyi}	Δ_{ort}	$\Delta_{enkı}$	Δ_{std}	Δ_{enlyi}	Δ_{ort}	$\Delta_{enkı}$	Δ_{std}	Δ_{enlyi}	Δ_{ort}	$\Delta_{enkı}$	Δ_{std}
20	5	ta001-ta010	4.23	5.73	7.22	12.07	3.99	5.89	7.66	14.42	2.92	4.01	5.04	8.95	9.86	12.04	15.02	22.07
	10	ta011-ta020	7.34	9.02	10.42	14.46	7.57	9.44	11.37	19.77	4.86	6.44	7.78	14.82	11.95	15.94	18.81	32.73
	20	ta021-ta030	5.54	6.78	7.88	17.14	5.44	7.10	8.58	22.08	4.35	5.66	6.97	17.78	9.10	12.44	14.69	42.55
Ortalama=			5.70	7.18	8.50	14.56	5.66	7.48	9.20	18.76	4.04	5.37	6.60	13.85	10.30	13.47	16.17	32.45

Deneme sonuçları, sezgisel bilgi kullanımının algoritmanın performansını olumlu yönde etkilediğini göstermiştir. Ayrıca, SPIRIT stratejisinin en iyi sezgisel bilgiyi verdiğini görülmüştür. Sezgisel bilgi kullanılması durumunda en iyi strateji için iyileşme ortalama %60 oranında olmuştur. Diğer iki sezgisel bilgi TSPT ve GJI ise, sezgisel bilginin kullanılmadığı duruma göre daha iyi sonuçlar verirken, SPIRIT stratejisine göre daha kötü çözümler üretmişlerdir.



Şekil 6.24 Problem Hacmi Bazında Sezgisel Bilginin Çözümüne Etkisi

6.1.8 Aday Liste Kullanımının Çözümüne Etkisi

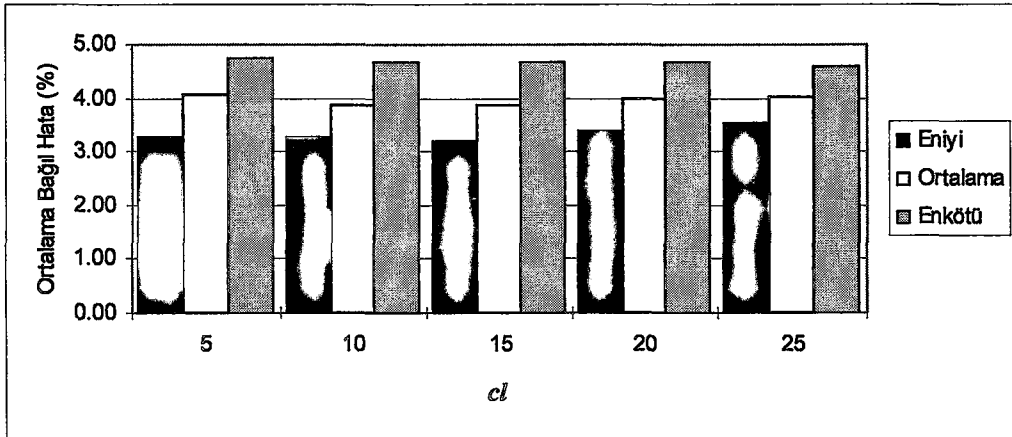
Öncelikle aday listenin büyüklüğünün ne kadar olması gerektiği araştırılmıştır. Bu amaçla Taillard' ın 100x10 boyutundaki 5 test problemi üzerinde denemeler yapılmıştır. Problemin boyutunun büyük seçilmesinin nedeni, her aday liste büyüklüğü için yapılacak denemelerde oluşacak etkinin daha net ortaya konulabilmesi içindir. Aday liste büyüklükleri için $cl = \{5, 10, 15, 20, 25\}$ olarak

belirlenmiş ve her liste büyüklüğü için denemeler yapılmıştır. Başlangıç parametre değerleri ise, $\alpha = 2$, $\beta = 0.5$, $\rho_l = 0.2$, $\rho_g = 0.2$, $q_0 = 0.9$, $k_{\max} = 20$ ve $t_{\max} = 1000$ olarak sabitlenmiştir. Bu testlerde algoritma yerel arama özelliği kullanılmamıştır. Her bir deneme 10 kez tekrarlanmış, böylece toplam 250 adet daha deneme yapılmıştır. Deneme sonuçları Ek I' da verilmektedir. En iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması, Taillard' ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Tablo 6.21 ve Şekil 6.25' de özetlenmiştir.

Tablo 6.21 Aday Liste Büyüklüğünün Çözümüne Etkisi

cl	Δ_{eniyi}					Δ_{ort}					$\Delta_{\text{enkötü}}$					Ortalama		
	Problem No					Problem No					Problem No					Δ_{eniyi}	Δ_{ort}	$\Delta_{\text{enkötü}}$
	ta071	ta072	ta073	ta074	ta075	ta071	ta072	ta073	ta074	ta075	ta071	ta072	ta073	ta074	ta075			
5	3.22	3.85	1.99	3.47	3.88	3.89	4.74	2.69	4.70	4.36	4.38	5.83	3.38	5.35	4.77	3.28	4.07	4.75
10	3.19	3.91	2.15	3.73	3.33	3.73	4.39	2.96	4.34	3.98	4.18	5.80	3.79	5.15	4.46	3.26	3.88	4.67
15	2.50	3.25	2.25	3.82	4.15	3.25	4.28	2.95	4.43	4.48	4.11	5.12	3.82	5.25	4.92	3.19	3.88	4.64
20	3.17	3.33	2.31	3.95	4.19	3.64	3.90	3.10	4.48	4.74	4.04	4.58	4.21	4.99	5.41	3.39	3.97	4.65
25	3.52	3.42	2.78	3.71	4.17	3.87	4.11	3.22	4.33	4.56	4.35	4.77	3.86	5.11	4.79	3.52	4.02	4.58

Deneme sonuçları, kullanılan aday liste büyüklükleri için algoritma performans sonuçlarının çok büyük fark oluşturmadığını göstermiştir. En iyi çözüm sonuçları aday liste büyüklüğü $cl = 15$ ve en kötü çözüm sonuçları aday liste büyüklüğü $cl = 25$ olduğunda bulunmuştur.



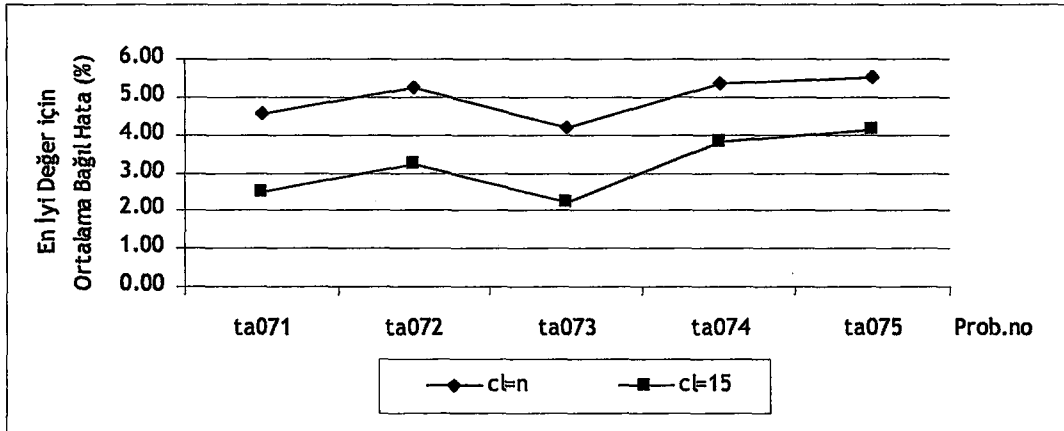
Şekil 6.25 Aday Liste Büyüklüğünün Çözümüne Etkisi

Yapılan denemeler sonucu bulunan en iyi aday liste büyüklüğü $cl = 15$ kullanılarak aday liste kullanımının çözüm üzerinde etkisi olup olmadığı araştırılmıştır. Bu amaçla Taillard' ın 100x10 boyutundaki 5 test problemi üzerinde denemeler yapılmıştır. Başlangıç parametre değerleri $\alpha = 2$, $\beta = 0.5$, $\rho_l = 0.2$, $\rho_g = 0.2$, $q_0 = 0.9$, $k_{\max} = 20$ ve $t_{\max} = 1000$ olarak alınarak her bir deneme 10 kez tekrarlanmış, böylece toplam 100 adet deneme yapılmıştır. Bu testlerde yerel arama özelliği kullanılmamıştır. Deneme sonuçları Ek I' da verilmektedir. En iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması, Taillard' ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Tablo6.22 ve Şekil 6.26' da özetlenmiştir.

Tablo 6.22 Aday Liste Kullanımının Çözüme Etkisi

n	m	Problem No	cl=n				cl=15			
			Δ_{eniyi}	Δ_{ort}	$\Delta_{\text{enkötü}}$	Δ_{std}	Δ_{eniyi}	Δ_{ort}	$\Delta_{\text{enkötü}}$	Δ_{std}
100	10	ta071	4.54	5.20	5.93	25.01	2.50	3.25	4.11	25.39
	10	ta072	5.25	6.00	6.47	20.70	3.25	4.28	5.12	38.79
	10	ta073	4.19	4.57	4.99	13.16	2.25	2.95	3.82	25.11
	10	ta074	5.39	5.82	6.46	19.45	3.82	4.43	5.25	26.68
	10	ta075	5.54	5.88	6.33	14.58	4.15	4.48	4.92	15.67
Ortalama=			4.98	5.49	6.03	18.58	3.19	3.88	4.64	26.33

Aday liste kullanımının çözüm performansı üzerinde bir etkisinin denemelerde en iyi, ortalama ve en kötü değerlere göre sırasıyla %37, 30 ve 23 olduğu bulunmuştur.



Şekil 6.26 Aday Liste Kullanımının Çözüme Etkisi

6.1.9 Yerel Geliştirme Yönteminin Çözüm Etkisi

Burada yerel geliştirme yönteminin çözüm performansına olan etkisi karşılaştırılacaktır. Kullanılacak yerel arama komşuluk yapıları:

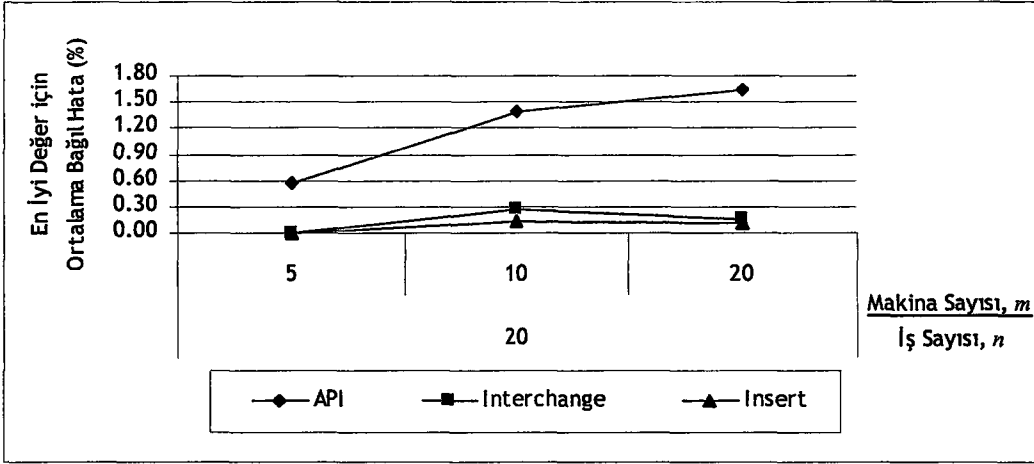
- Transpose (Adjacent Pairwise Interchange (API))
- Interchange (General Pairwise Interchange)
- Insert (Shift)

Yerel geliştirme yönteminin çözüm performansı üzerindeki etkisini görmek üzere Taillard'ın 20x5, 20x10 ve 20x20 boyutundaki 30 test problemi üzerinde denemeler yapılmıştır. Başlangıç parametre değerleri $\alpha = 2$, $\beta = 0.5$, $\rho_l = 0.2$, $\rho_g = 0.2$, $q_0 = 0.9$, $k_{\max} = 20$ ve $t_{\max} = 1000$ olarak alınarak her bir deneme 10 kez tekrarlanmış, böylece toplam 900 adet daha deneme yapılmıştır. Deneme sonuçları ve en iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması, Taillard'ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Ek J' de verilmektedir. Algoritma performans sonuçları problem hacmi bazında Tablo 6.23 ve Şekil 6.27' de özetlenmiştir.

Tablo 6.23 Problem Hacmi Bazında Yerel Geliştirme Yöntemlerinin Çözüm Etkisi

n	m	Problem No	API				Interchange				Insert			
			Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}
20	5	ta001-ta010	0.56	0.93	1.24	3.01	0.00	0.06	0.13	0.64	0.00	0.03	0.14	0.64
	10	ta011-ta020	1.39	2.09	2.73	6.75	0.28	0.58	0.96	3.49	0.13	0.49	0.76	3.18
	20	ta021-ta030	1.64	2.27	2.79	8.25	0.16	0.55	0.92	5.61	0.11	0.37	0.62	3.80
Ortalama=			1.20	1.76	2.25	6.00	0.15	0.40	0.67	3.25	0.08	0.30	0.51	2.54

Yapılan çalışmalar sonucu insert yerel geliştirme yönteminin çözüm performansı üzerinde en iyi etkiye sahip olduğu görülmüştür. Interchange yerel geliştirme yönteminin çözüm performansı üzerinde etkisi insert yöntemine çok yakın iken, en kötü sonucu API yönteminin verdiği görülmüştür.



Şekil 6.27 Problem Bazında Yerel Geliştirme Yöntemlerinin Çözümüne Etkisi

6.2 Performans Değerlendirmesi

Geliştirilen algoritmayı değerlendirmek için Taillard'ın 90 test problemi üzerinde başlangıç parametre değerleri $\alpha = 2$, $\beta = 0.5$, $\rho_l = 0.2$, $\rho_g = 0.2$, $q_0 = 0.9$, $k_{\max} = 20$, $t_{\max} = 1000$ ve $cl = 15$ olarak test edilmiştir. 20 ve 50 işli problemlerde her bir deneme 10 kez, 100 işli problemlerde 5 kez tekrarlanarak böylece toplam 750 adet deneme yapılmıştır.

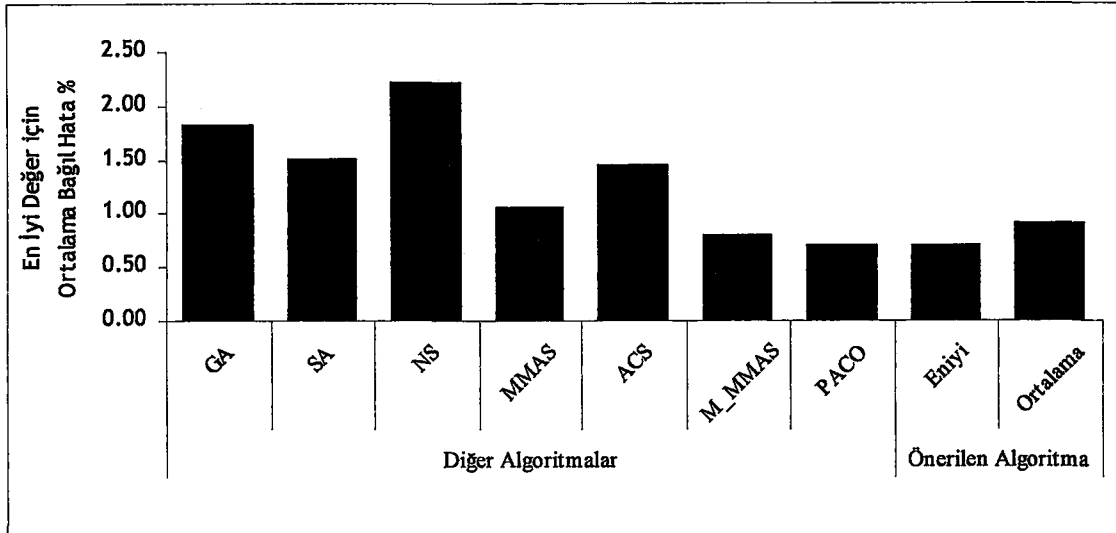
KKE algoritmasında geçiş kuralı için sanki-rasgele-oran kuralı, pheromone güncelleme kuralı için iterasyon eniyi stratejisinin olduğu karınca koloni sistemi pheromone güncelleme kuralı, sezgisel bilgi için SPIRIT yaklaşımı kullanılmıştır. Yerel arama için insert komşuluk yapısı ve en iyi gelişim yaklaşımı kullanılmıştır.

Deneme sonuçları ve en iyi, ortalama, en kötü değerlere göre denemelerin aritmetik ortalaması, Taillard'ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Ek K' da verilmektedir. Toplam tamamlanma zamanı en küçüklenmesi için literatürde mevcut diğer metasezgiseller (GA, SA ve NS) ve geliştirilen karınca kolonileri algoritmaları (MMAS, ACS, M-MMAS ve PACO) ile önerilen algoritmanın sonuçları ile karşılaştırılmıştır. Sonuçlar, Taillard'ın eniyi toplam tamamlanma zamanı üst sınır değerlerinden ortalama yüzde fark olarak Tablo 6.24 ve Şekil 6.28' de görülmektedir.

Tablo 6.24 Algoritmanın Toplam Tamamlanma Zamanı Amacı için Performans Değerlendirmesi

<i>n</i>	<i>m</i>	Problem No	Diğer Algoritmalar							Önerilen Algoritma			
			GA	SA	NS	MMAS	ACS	M_MMAS	PACO	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötu}$	Δ_{std}
20	5	ta001-010	1.610	1.270	1.460	0.408	1.190	0.762	0.704	0.000	0.099	0.232	1.135
	10	ta011-020	2.290	1.710	2.020	0.591	1.700	0.890	0.843	0.183	0.489	0.817	3.359
	20	ta021-030	1.950	0.860	1.100	0.410	1.600	0.721	0.720	0.097	0.332	0.655	4.266
50	5	ta031-040	0.450	0.780	0.790	0.145	0.430	0.144	0.090	0.014	0.075	0.155	1.423
	10	ta041-050	2.280	1.980	3.210	2.193	1.890	1.118	0.746	1.323	1.714	2.024	6.937
	20	ta051-060	3.440	2.860	3.900	2.475	2.710	2.013	1.855	2.487	2.957	3.312	9.919
100	5	ta061-070	0.230	0.560	0.760	0.196	0.220	0.084	0.072	-0.027	-0.005	0.019	1.096
	10	ta071-080	1.250	1.330	2.690	0.928	1.220	0.451	0.404	0.465	0.599	0.732	6.378
	20	ta081-090	2.910	2.320	3.980	2.238	2.220	1.030	0.985	1.777	2.018	2.190	10.689
Ortalama=			1.823	1.519	2.212	1.065	1.464	0.801	0.713	0.702	0.920	1.126	5.022

Burada algoritmanın, bu problem için literatürde mevcut olan diğer sezgisellerden ve karınca algoritmalarından daha iyi sonuçlar verdiği görülmüştür. Diğer iki karınca algoritması M-MMAS ve PACO, önerilen algoritmaya yakın çözümler vermiştir.



Şekil 6.28 Algoritmanın Toplam Tamamlanma Zamanı Amacı için Ortalama Performans Değerlendirmesi

Buradan sonra algoritmanın çok-amaçlı akış tip çizelgeleme problemi için kullanılması durumunda nasıl sonuçlar verdiğini görmek üzere denemeler yapılmıştır.

6.3 Çok-Amaçlı Durum için Denemeler ve Hesaplama Sonuçları

Çok-amaçlı akış tipi çizelgeleme problemi için önerilen algoritmanın değerlendirilmesi yapılmıştır. Çok-amaçlı problem için ise Reeves (1995) tarafından sunulan 5, 10 ve 15 makine ve 20, 30 ve 50 işten oluşan 18 test problemi kullanılacaktır. Her problem büyüklüğü için 3 örnek oluşturulmuştur. Bu test problemlerinin dosyaları OR-library web sitesinden (URL:<http://mscmga.ms.ic.ac.uk/info.html>) elde edilebilir.

Test problemi kullanılarak elde edilecek çözümlerin performansı NEH sezgiselinin ve literatürde bu problem için önerilmiş ilk algoritma olan HC sezgiselinin performans sonuçları ile karşılaştırılmıştır.

HC sezgiseli ardışık işler arasındaki boşlukları en küçükmeye çalışır. Bu boşluklar aşağıdaki gibi hesaplanır:

$$d_{ij}^k = t_{i,k+1} - t_{j,k} \quad i, j = 1, 2, \dots, n, \quad k = 1, 2, \dots, m-1 \text{ ve } i \neq j \quad (6.25)$$

Çizelgede i işi j işinden önce ise, pozitif d_{ij}^k değeri, j işinin, i işi bitene dek en az d_{ij}^k kadar $k+1$ makinasında beklemesi gerektiğini belirtir. Negatif d_{ij}^k değeri, $k+1$ makinasında i işi ve j işi arasında d_{ij}^k kadar aylak zamanı tanımlar. Negatif değerleri azaltmak için, belli bir faktör değeri ($faktör(k)$) tanımlanmıştır. Bu faktör, pozitif boşlukları azaltmak için ilk makinaya yüksek ve son makinaya düşük atanır.

$$faktör(k) = \frac{(1.0 - 0.1)(m - k - 1)}{(m - 2)} + 0.1 \quad k = 1, 2, \dots, m-1 \quad (6.26)$$

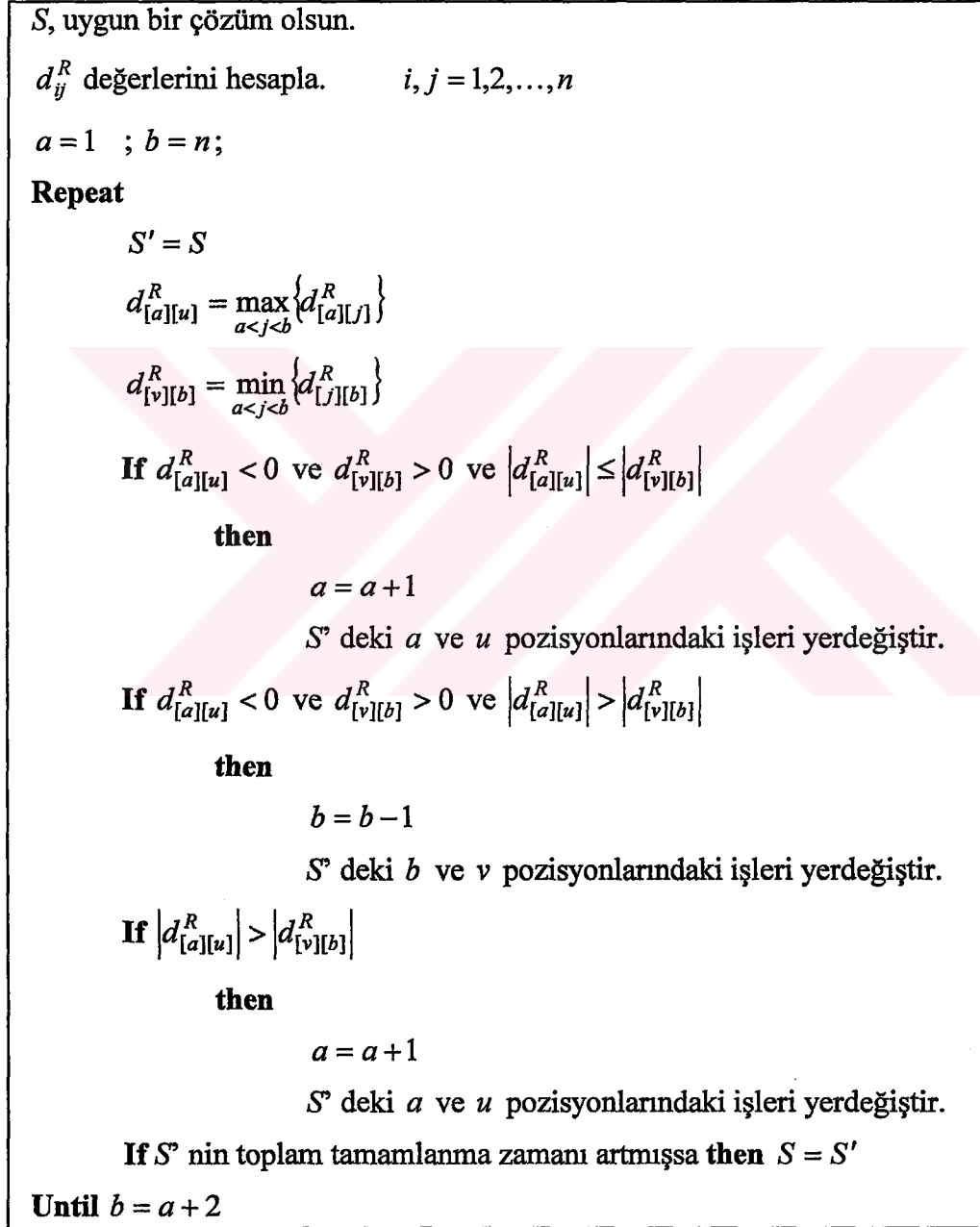
Bu faktör değerine dayanarak, azaltma fonksiyonu aşağıdaki gibi tanımlanır:

$$\delta_{ij}^k = \begin{cases} faktör(k) & \text{eger } d_{ij}^k < 0 \\ 1 & \text{diger} \end{cases} \quad i, j = 1, 2, \dots, n, \quad k = 1, 2, \dots, m-1 \quad (6.27)$$

Tüm düzeltilmiş boşluklar d_{ij}^R , d_{ij}^k ve azaltma fonksiyonu kullanılarak hesaplanır:

$$d_{ij}^R = \sum_{k=1}^{m-1} d_{ij}^k \delta_{ij}^k \quad i, j = 1, 2, \dots, n \quad (6.28)$$

HC sezgiselinin çalışma mekanizması Şekil 6.29' da özetlenmiştir (Ho ve Chang, 1991).



Şekil 6.29 HC Sezgiseli (Ho ve Chang, 1991)

Başlangıç iz yoğunluğu (τ_0) için, çok amaçlı problem durumunda test problemleri için eniyi değerler bilinmediği için bir başlangıç çözümü gerekir. Bu çalışmada

başlangıç çözümü NEH sezgiseli tarafından belirlenmektedir. Bu sezgiselinin seçilmesinin nedeni uygulaması basit olmasının yanında, bu sezgiselin performansının karşılaştırıldığı diğer sezgiselden (en iyi olduğu bilinen CDS sezgiseli) daha hızlı ve güçlü sonuçlar vermesidir (Turner ve Booth, 1987). NEH sezgiselinin çalışma mekanizması Şekil 6.30’ da özetlenmiştir. NEH sezgiseli, çok amaçlı durum için bazı değişiklikler ile kullanılabilir. Buna göre, NEH sezgiseli toplam tamamlanma zamanı amacı (f_1) için çalıştırılır. Ortaya çıkan sıralama için, diğer amaç değerleri (f_2 , toplam akış zamanı ve f_3 , toplam makine aylak kalma zamanı) hesaplanır. Böylece başlangıç pheromone miktarı $\tau_0 = [n.(f_1 + f_2 + f_3)]^{-1}$ olarak belirlenir. Burada n , problem büyüklüğüdür.

Adım1. Her i işi için, $t_{i,j}$, i işinin j makinesindeki işlem süresi olmak üzere;

$$T_i = \sum_{j=1}^m t_{i,j}$$

Adım2. T_i azalan sırasına göre işleri düzenle.

Adım3. Adım2’ deki listenin ilk ve ikinci pozisyonundaki işleri seç ve iki mümkün sıralama için, amaç değerini hesapla ve bu iki iş için en iyi sıralamayı bul. Algoritmanın diğer adımlarında bu iki işin birbirlerine göre göreceli sıralamasını değiştirme. $i = 3$.

Adım4. Adım2’ deki listenin i .inci pozisyonundaki işi seç ve atanan işlerin birbirlerine göre göreceli sıralamasını değiştirmeksizin önceki adımda bulunan kısmi sıralamada tüm mümkün i pozisyonlarına bu işi yerleştirerek en iyi sıralamayı bul. Bu adımdaki hesaplama sayısı i ’ ye eşittir.

Adım5: Eğer $n = i$ ise, dur. Aksi halde, $i = i + 1$ yap ve Adım4’ e git.

Şekil 6.30 NEH Sezgiseli (Nawaz ve diğ., 1983)

Başlangıç parametre değerleri olarak, tek amaçlı durum için yapılan deneme çalışmaları sonucu bulunmuş en iyi değerler $\alpha = 2$, $\beta = 0.5$, $\rho_l = 0.2$, $\rho_g = 0.2$, $q_0 = 0.9$, $k_{\max} = 20$, $t_{\max} = 1000$ ve $cl = 15$ kullanılacaktır. Her bir deneme 10 kez tekrarlanmış, böylece toplam 180 adet deneme yapılmıştır.

KKE algoritmasında geçiş kuralı için sanki-rasgele-oran kuralı, pheromone güncelleme kuralı için iterasyon en iyi stratejisinin olduğu karınca koloni sistemi pheromone güncelleme kuralı, sezgisel bilgi için SPIRIT yaklaşımı kullanılmıştır. Yerel arama için insert komşuluk yapısı ve en iyi gelişim yaklaşımı kullanılmıştır.

Toplam amaç değeri için sırasıyla toplam tamamlanma zamanı, toplam akış zamanı ve toplam makine aylak kalma zamanı amaçlarına eşit ağırlık ($a_1 = a_2 = a_3 = 0,333$) verilmiştir.

Önerilen algoritma tarafından oluşturulan T_1 çizelgesinin toplam tamamlanma zamanı amacı için bağıl yüzde artış:

$$\frac{f_1 - \min(f_1, f'_1, f''_1)}{\min(f_1, f'_1, f''_1)} * 100 \quad (6.29)$$

Önerilen algoritma tarafından oluşturulan T_1 çizelgesinin toplam akış zamanı için bağıl yüzde artış:

$$\frac{f_2 - \min(f_2, f'_2, f''_2)}{\min(f_2, f'_2, f''_2)} * 100 \quad (6.30)$$

Önerilen algoritma tarafından oluşturulan T_1 çizelgesinin toplam makine aylak kalma zamanı için bağıl yüzde artış:

$$\frac{f_3 - \min(f_3, f'_3, f''_3)}{\min(f_3, f'_3, f''_3)} * 100 \quad (6.31)$$

Önerilen algoritma tarafından oluşturulan T_1 çizelgesinin toplam amaç için bağıl yüzde artış:

$$a_1 \left(\frac{f_1 - \min(f_1, f'_1, f''_1)}{\min(f_1, f'_1, f''_1)} \right) + a_2 \left(\frac{f_2 - \min(f_2, f'_2, f''_2)}{\min(f_2, f'_2, f''_2)} \right) + a_3 \left(\frac{f_3 - \min(f_3, f'_3, f''_3)}{\min(f_3, f'_3, f''_3)} \right) \quad (6.32)$$

HC sezgiseli tarafından oluşturulan T_2 çizelgesinin toplam tamamlanma zamanı amacı için bağıl yüzde artış:

$$\frac{f'_1 - \min(f'_1, f_1, f''_1)}{\min(f'_1, f_1, f''_1)} * 100 \quad (6.33)$$

HC sezgiseli tarafından oluşturulan T_2 çizelgesinin toplam akış zamanı için bağıl yüzde artış:

$$\frac{f'_2 - \min(f'_2, f_2, f''_2)}{\min(f'_2, f_2, f''_2)} * 100 \quad (6.34)$$

HC sezgiseli tarafından oluşturulan T_2 çizelgesinin toplam makine aylak kalma zamanı için bağıl yüzde artış:

$$\frac{f'_3 - \min(f'_3, f_3, f''_3)}{\min(f'_3, f_3, f''_3)} * 100 \quad (6.35)$$

HC sezgiseli tarafından oluşturulan T_2 çizelgesinin toplam amaç için bağıl yüzde artış:

$$a_1 \left(\frac{f'_1 - \min(f'_1, f_1, f''_1)}{\min(f'_1, f_1, f''_1)} \right) + a_2 \left(\frac{f'_2 - \min(f'_2, f_2, f''_2)}{\min(f'_2, f_2, f''_2)} \right) + a_3 \left(\frac{f'_3 - \min(f'_3, f_3, f''_3)}{\min(f'_3, f_3, f''_3)} \right) \quad (6.36)$$

NEH sezgiseli tarafından oluşturulan T_3 çizelgesinin toplam tamamlanma zamanı amacı için bağıl yüzde artış:

$$\frac{f''_1 - \min(f''_1, f_1, f'_1)}{\min(f''_1, f_1, f'_1)} * 100 \quad (6.37)$$

NEH sezgiseli tarafından oluşturulan T_3 çizelgesinin toplam akış zamanı için bağıl yüzde artış:

$$\frac{f''_2 - \min(f''_2, f_2, f'_2)}{\min(f''_2, f_2, f'_2)} * 100 \quad (6.38)$$

NEH sezgiseli tarafından oluşturulan T_3 çizelgesinin toplam makine aylak kalma zamanı için bağıl yüzde artış:

$$\frac{f''_3 - \min(f''_3, f_3, f'_3)}{\min(f''_3, f_3, f'_3)} * 100 \quad (6.39)$$

NEH sezgiseli tarafından oluşturulan T_3 çizelgesinin toplam amaç için bağıl yüzde artış:

$$a_1 \left(\frac{f''_1 - \min(f''_1, f_1, f'_1)}{\min(f''_1, f_1, f'_1)} \right) + a_2 \left(\frac{f''_2 - \min(f''_2, f_2, f'_2)}{\min(f''_2, f_2, f'_2)} \right) + a_3 \left(\frac{f''_3 - \min(f''_3, f_3, f'_3)}{\min(f''_3, f_3, f'_3)} \right) \quad (6.40)$$

Burada, f'_1 , f'_2 ve f'_3 , HC sezgiseli tarafından oluşturulan ve f''_1 , f''_2 ve f''_3 , NEH sezgiseli tarafından oluşturulan sırasıyla toplam tamamlanma zamanı, toplam akış zamanı ve toplam makine aylak kalma zamanı amaç değerleridir.

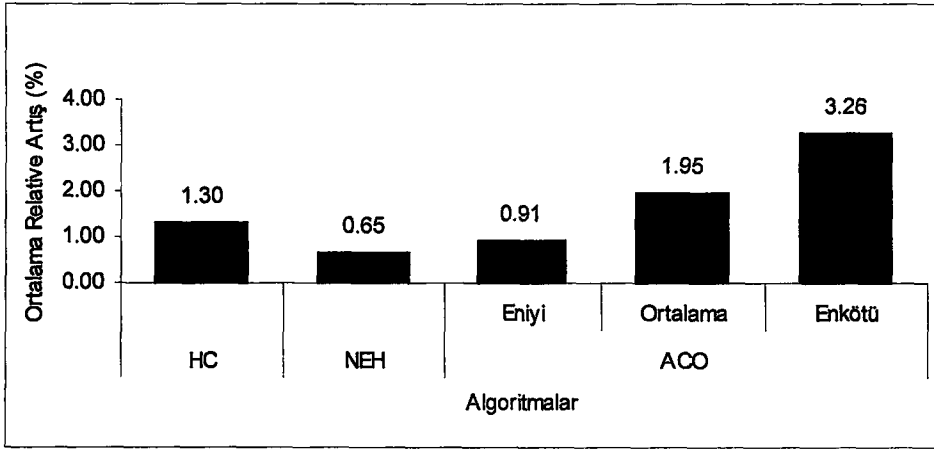
Önerilen algoritmanın deneme sonuçları ve en iyi, ortalama ve en kötü değerlere göre denemelerin aritmetik ortalaması Ek L' de verilmektedir. Ayrıca, HC ve NEH sezgiselinin çözüm sonuçları da Ek L' de verilmektedir.

Problem hacmi bazında ortalama yüzde bağıl artış olarak toplam tamamlanma zamanı için Tablo 6.25 ve ortalama yüzde bağıl artış olarak Şekil 6.31' de özetlenmiştir.

Tablo 6.25 Toplam Tamamlanma Zamanı Amacı için Algoritmaların Performans Sonuçları

$n \times m$	Problem No	HC	NEH	KKE		
		Δ_{ort}	Δ_{ort}	Δ_{eni}	Δ_{ort}	$\Delta_{enkötü}$
20x5	<i>ReC01-ReC05</i>	2.00	0.00	1.63	3.01	4.57
20x10	<i>ReC07-ReC11</i>	1.42	1.42	0.00	0.01	0.04
20x15	<i>ReC13-ReC17</i>	1.05	0.45	0.58	1.37	1.99
30x10	<i>ReC19-ReC23</i>	1.14	1.19	0.18	1.73	3.77
30x15	<i>ReC25-ReC29</i>	1.59	0.86	0.45	1.60	3.96
50x10	<i>ReC31-ReC35</i>	0.62	0.00	2.61	3.97	5.25
Ortalama=		<i>1.30</i>	<i>0.65</i>	<i>0.91</i>	<i>1.95</i>	<i>3.26</i>

Yapılan testler sonucu, toplam tamamlanma zamanı için en iyi performansı NEH sezgiseli vermiştir. Burada önerilen KKE algoritması ise NEH algoritmasından daha iyi ve HC sezgiseliden daha kötü sonuçlar verdiği görülmüştür.



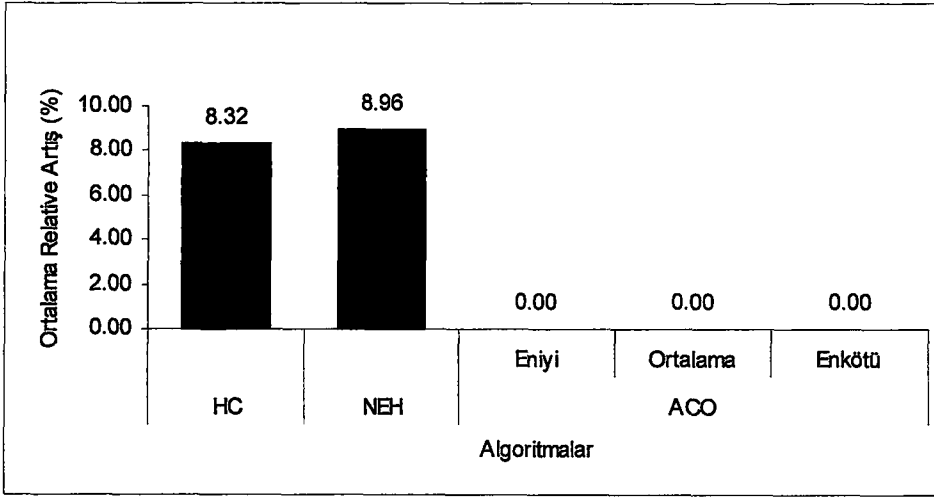
Şekil 6.31 Toplam Tamamlanma Zamanı Amacı için Algoritmaların Ortalama Performans Sonuçları

Problem hacmi bazında ortalama yüzde bağıl artış olarak toplam akış zamanı için Tablo 6.26 ve ortalama yüzde bağıl artış olarak Şekil 6.32’ de özetlenmiştir.

Tablo 6.26 Toplam Akış Zamanı Amacı için Algoritmaların Performans Sonuçları

$n \times m$	Problem No	HC	NEH	KKE		
		Δ_{ort}	Δ_{ort}	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$
20x5	ReC01-ReC05	11.10	12.75	0.00	0.00	0.00
20x10	ReC07-ReC11	7.43	7.43	0.00	0.00	0.00
20x15	ReC13-ReC17	5.57	5.65	0.00	0.00	0.00
30x10	ReC19-ReC23	9.77	10.59	0.00	0.00	0.00
30x15	ReC25-ReC29	5.99	6.23	0.00	0.00	0.00
50x10	ReC31-ReC35	10.04	11.09	0.00	0.00	0.00
Ortalama=		8.32	8.96	0.00	0.00	0.00

Yapılan testler sonucu, toplam akış zamanı için en iyi performansı önerilen KKE algoritması vermiştir. HC ve NEH sezgiselleri ise benzer çözüm kalitesi göstererek önerilen algorithmadan daha kötü sonuçlar verdiği görülmüştür.



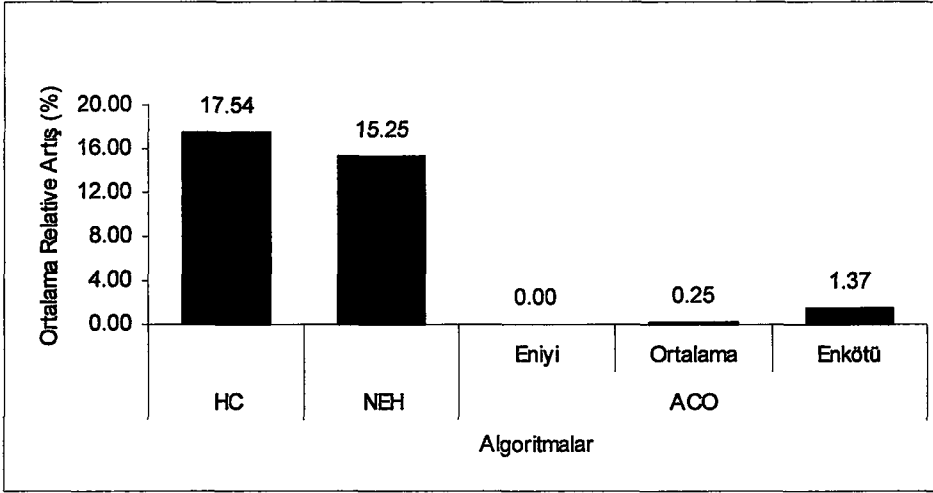
Şekil 6.32 Toplam Akış Zamanı Amacı için Algoritmaların Ortalama Performans Sonuçları

Problem hacmi bazında ortalama yüzde bağıl artış olarak toplam makine aylak kalma için Tablo 6.27 ve ortalama yüzde bağıl artış olarak Şekil 6.33’ de özetlenmiştir.

Tablo 6.27 Toplam Aylak Kalma Zamanı Amacı için Algoritmaların Performans Sonuçları

$n \times m$	Problem No	HC	NEH	KKE		
		Δ_{ort}	Δ_{ort}	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$
20x5	ReC01-ReC05	21.94	10.12	0.00	0.00	1.91
20x10	ReC07-ReC11	18.43	18.43	0.00	0.00	0.00
20x15	ReC13-ReC17	13.71	16.25	0.00	0.00	0.00
30x10	ReC19-ReC23	19.59	19.16	0.00	0.00	1.43
30x15	ReC25-ReC29	17.81	16.73	0.00	0.00	0.00
50x10	ReC31-ReC35	13.78	10.80	0.00	1.52	4.91
Ortalama=		17.54	15.25	0.00	0.25	1.37

Yapılan testler sonucu, toplam aylak kalma zamanı için en iyi performansı önerilen KKE algoritması vermiştir. HC ve NEH sezgiselleri ise benzer çözüm kalitesi göstererek önerilen algoritmadan daha kötü sonuçlar verdiği görülmüştür.



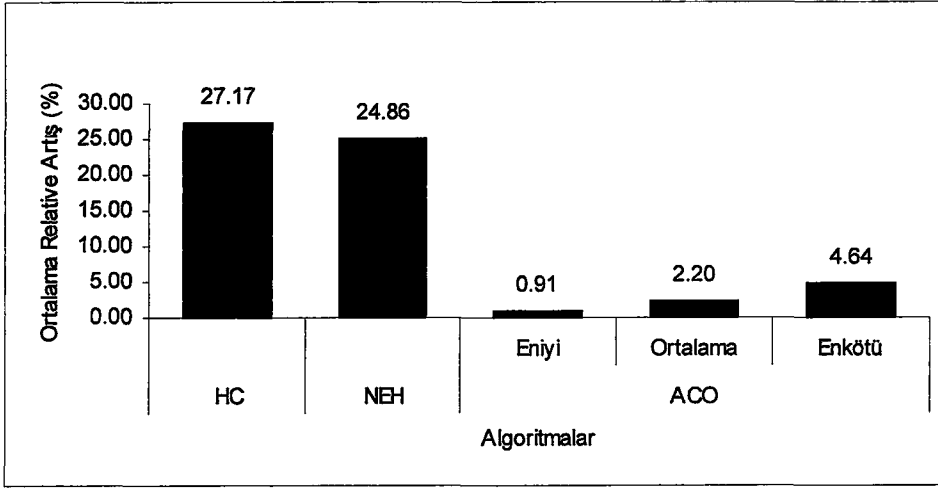
Şekil 6.33 Toplam Aylak Kalma Zamanı Amacı için Algoritmaların Ortalama Performans Sonuçları

Problem hacmi bazında ortalama yüzde bağıl artış olarak tüm amaçların toplamı için Tablo 6.28 ve ortalama yüzde bağıl artış olarak Şekil 6.34’ de özetlenmiştir.

Tablo 6.28 Tüm Amaçların Toplamı için Algoritmaların Performans Sonuçları

$n \times m$	Problem No	HC	NEH	KKE		
		Δ_{ort}	Δ_{ort}	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$
20x5	<i>ReC01-ReC05</i>	35.04	22.87	1.63	3.01	6.47
20x10	<i>ReC07-ReC11</i>	27.29	27.29	0.00	0.01	0.04
20x15	<i>ReC13-ReC17</i>	20.33	22.35	0.58	1.37	1.99
30x10	<i>ReC19-ReC23</i>	30.51	30.95	0.18	1.73	5.20
30x15	<i>ReC25-ReC29</i>	25.40	23.83	0.45	1.60	3.96
50x10	<i>ReC31-ReC35</i>	24.43	21.89	2.61	5.49	10.16
Ortalama=		<i>27.17</i>	<i>24.86</i>	<i>0.91</i>	<i>2.20</i>	<i>4.64</i>

Çalışmalar sonucu, tüm amaçların toplamı için en iyi performansı önerilen KKE algoritması vermiştir. HC ve NEH sezgiselleri ise benzer çözüm kalitesi göstererek önerilen algoritmadan daha kötü sonuçlar verdiği görülmüştür.



Şekil 6.34 Tüm Amaçların Toplamı için Algoritmaların Ortalama Performans Sonuçları

7. SONUÇLAR ve TARTIŞMA

Akış tipi çizelgeleme problemi, günümüzde birçok üretim koşulunda yaygın şekilde kullanıldığı için üzerinde dikkatle durulan bir problem olmuştur. Akış tipi çizelgeleme problemi için mevcut teknolojik kısıtlar nedeniyle işlerin makinelerden aynı sırada geçmesi gerekir. Johnson algoritması ile n iş ve 2 makineli akış tipi çizelgeleme problemi için işlerin toplam tamamlanma zamanını en küçükleyecek bir eniyi çözüm bulmak mümkündür. Johnson algoritması ile n iş ve 3 makineli akış tipi çizelgeleme problemi için bazı özel şartlar altında işlerin toplam tamamlanma zamanını en küçükleyecek yine bir eniyi çözüm bulmak mümkündür. Ancak, çok sayıda iş ve makine içeren akış tipi çizelgeleme problemleri birleşik eniyileme problemi özelliğindedir ve NP-zor problemi sınıfındadır. Bu tip problemler için son zamanlarda bilgisayarların CPU gücünün artması ile iterasyona dayanan yaklaşık çözüm yöntemlerinin kullanılması söz konusu olmuştur.

Literatüre bakıldığında bu problem için çok sayıda çalışma yapıldığı ve birçok çözüm yaklaşımı önerildiği görülmektedir. Ancak bu çalışmaların bir çoğu tek kriter üzerine yoğunlaşmıştır. Günümüz koşullarında birçok gerçek hayat problemi, birden çok kriteri dikkate alacak çözüm yaklaşımlarına gerek duymaktadır. Bu nedenle son zamanlarda yapılan çalışmalar, sistemlerin çok amaçlı durumları için çözüm oluşturacak yaklaşımların bulunması yönünde olmaktadır. Bu çalışmada da bu amaçla çok amaçlı durum dikkate alınmıştır. Gözönüne alınan amaçlar sırasıyla, toplam tamamlanma zamanı, toplam akış zamanı ve toplam makine aylak kalma zamanıdır.

Son zamanlarda literatürde birleşik eniyileme problemlerine çözüm sağlamak üzere metasezgisel yaklaşımların kullanımı gösterdikleri performans açısından tercih edilir olmuştur. Akış tipi çizelgeleme problemi için son zamanlarda yapılan çalışmalara bakıldığında ise metasezgisel yaklaşımlara dayanan çözüm yöntemlerinin çok sık önerildiği görülmektedir. Bunlar arasında tek amaçlı akış tipi problem için çözüm oluşturacak çok az sayıda karınca kolonileri eniyileme metasezgisel yaklaşımı vardır ve çözüm kalitesi açısından diğer yöntemlerden daha başarılı sonuçlar vermiştir.

Ancak henüz çok amaçlı akış tipi problemi için önerilmiş KKE algoritması bulunmamaktadır. Bu nedenle tez kapsamında, tek amaçlı akış tipi problem üzerindeki başarısından hareketle çok amaçlı problem durumu için KKE metasezgisel yaklaşımı önerilmiş ve göstereceği çözüm performansı araştırılmıştır.

Birkaç KKE metasezgisel çeşiti vardır. Burada önerilen yaklaşım, karınca koloni sistemine dayanmaktadır. Çalışma kapsamında çözüm yaklaşımı için önerilen algoritmanın uygun değerler üretilip üretilmediğini tespit etmek ve bunu literatürdeki diğer yöntemlerle kıyaslamak üzere öncelikle problem tek amaçlı (toplam tamamlanma zamanı) olarak ele alınmıştır. Tek amaçlı problem için algoritmanın en iyi sonuçlar üretmesini sağlayacak çalışmalardan sonra elde edilen gelişmelerle yaklaşım çok amaçlı problem için uyarlanmıştır.

Algoritmanın oluşturulmasında şu özellikler belirlenmiştir:

- Problemin Gösterimi
- Yapay Karıncaların Özellikleri
- Otokatalitik Süreç
- Yasaklar Listesi
- Karınca Sayısı
- Geçiş Olasılığı
- Pheromone İz Güncelleme Kuralı
- Sezgisel Bilgi
- Aday Liste
- Yerel Geliştirme Yöntemi

Öncelikle önerilen karınca algoritmasının en iyi şekilde çalışabilmesi için algoritmanın yapısında kullanılan parametrelerin (geçiş kuralı parametreleri, iterasyon sayısı, karınca sayısı) en uygun şekilde belirlenmesi gerekir. Bu amaçla, bu çalışma kapsamında iki seviyeli bir deneysel tasarım uygulanmış ve parametrelerin uygun değerleri belirlenmiştir.

Diğer yandan ele alınan problem için algoritmanın en iyi çözümleri üretecek şekilde çalışmasını sağlamak üzere bazı temel mekanizmalarının uygun özelliklere sahip

olması gerekir. Bu amaçla algoritmada önemli her mekanizma için (karıncalar arası işbirliği, geçiş olasılığı, pheromone izi güncelleme kuralı, sezgisel bilgi, aday liste büyüklüğü, yerel geliştirme yöntemi) uygun bir yapı tespit edecek deneysel çalışmalar yapılmıştır.

Sonuçlar, karıncalar arası işbirliğinin kullanılması gerektiğini göstermiştir. Geçiş olasılığı için en uygun kuralın karınca koloni sisteminde kullanılan sanki rasgele-oran kuralı olduğu tespit edilmiştir. Pheromone izi güncelleme kuralı için en uygun yöntem ise yine karınca koloni sistemi güncelleme kuralıdır. Burada global güncelleme kuralı için denemeler iterasyon-en iyi stratejisinin global-en iyi stratejisinden daha iyi sonuçlar verdiğini göstermiştir. Çalışmalarda, sezgisel bilgi için SPIRIT algoritmasında kullanılan işler arası uzaklık tanımının karşılaştırılan diğer sezgisel bilgi kavramlarından daha iyi sonuçlar verdiği görülmüştür. Aday liste büyüklüğü için en uygun değerin 15 olduğu tespit edilmiştir. Yerel geliştirme yöntemi için INSERT komşuluk yapısı karşılaştırılan yaklaşımlar içinde en iyi çözümü vermiştir.

Yapılan bu iyileştirme çalışmaları sonrası elde edilen algoritmanın akış tipi problemi toplam tamamlanma zamanı en küçüklenmesi için sonuçları literatürde mevcut diğer metasezgisellerin (GA, SA ve NS) ve geliştirilen diğer karınca kolonileri algoritmalarının (*MMAS*, ACS, M-MMAS ve PACO) sonuçları ile karşılaştırılmıştır. Önerilen algoritmanın, karşılaştırıldığı bu metasezgisellerden ve karınca algoritmalarından daha iyi sonuçlar verdiği görülmektedir. Buradan sonra algoritma çok amaçlı problem için uyarlanmış ve bu durumunda nasıl sonuçlar verdiğini görmek üzere denemeler yapılmıştır. Yapılan analizler, önerilen KKE algoritmasının, çok amaçlı akış tipi problemi için HC ve NEH sezgisellerinden daha iyi sonuçlar verdiğini göstermiştir. Ayrıca önerilen algoritma uygun hesaplama süresine sahip olması ile de kullanılabilir bir algoritma özelliğine sahiptir.

Burada akış tipi çizelgeleme problemi için KKE yaklaşımı kullanılarak daha etkin çözümler geliştirmek için mekanizmalar üzerinde daha ileri çalışmalar yapılabilir. Örneğin yerel geliştirme yöntemi seçiminde melez bir yaklaşım oluşturmak için daha ileri teknikler (değişken komşuluk arama, ardaşık yerel arama vb.) tercih edilebilir.

Bundan başka bu problemin farklı kriterleri (ortalama akış zamanı, toplam pozitif gecikme, maksimum pozitif gecikme vb.) gözönüne alan tek ve çok amaçlı durumları

iin nerilen KKE algoritması kullanılabilir ve yine iyi zmler retmesi beklenebilir.

nerilen KKE algoritmasının atlye tipi, hcresel retim, esnek retim, montaj hatları gibi retim ortamlarındaki izelgeleme problemleri iin kullanılması mmkndr.

Bunun yanında literatrde bu problem zerinde uygulanmamıř veya temel CO problemleri (TSP, QAP vs.) iin henz deneneyen yapay baėıřıklık sistemleri, diferansiyel geliřim, partikl kme eniyileme gibi farklı metasezgisel yaklařımlar ile alıřmalar yapılabilir.



KAYNAKLAR

- Aggoune, R.**, 2001. Minimizing the makespan for the flow shop scheduling problem with availability constraints, *ORP³ 2001*, Paris.
- Ahmadi, R. and Bargchi, U.**, 1990. Improved lower bound for minimizing the sum of flowtimes of n jobs over m machines in a flow shop, *European Journal Of Operational Research*, **44**, 331-336.
- Aldowaisan, T. and Allahverdi, A.**, 2003. New heuristics for no-wait flowshops to minimize makespan, *Computers and Operations Research*, **30**, 8, 1219-1231.
- Allahverdi, A.**, 2003. The two- and m-machine flowshop scheduling problems with bicriteria of makespan and mean flowtime, *European Journal of Operational Research*, **147**, 2, 373-396.
- Allahverdi, A.**, 2004. A new heuristic for m-machine flowshop scheduling problem with bicriteria of makespan and maximum tardiness, *Computers and Operations Research*, **31**, 2, 157-180.
- Allahverdi, A. and Aldowaisan, T.**, 2004. No-wait flowshops with bicriteria of makespan and maximum lateness, *European Journal Of Operational Research*, **152**, 132-147.
- Ashour, S.**, 1970. An experimental investigation and comparative evaluation of flowshop sequencing techniques, *Operations Research*, **18**, 3, 541-549.
- Baker, K.R.**, 1974. Introduction to sequencing and scheduling, John Wiley, NewYork.
- Baker, K.R.**, 1975. A comparative study of flow shop algorithms, *Operations Research*, **23**, 62-73.
- Barán, B. and Schaerer, M.**, 2003. A multiobjective ant colony system for vehicle routing problem with time windows, *Proceedings of the 21st IASTED International Conference Applied Informatics*, Innsbruck, Austria, 97-102.
- Basseur, M.; Seynhaeve, F. and Talbi, E-G.**, 2002. Design of multi-objective evolutionary algorithms: Application to the flow-shop scheduling problem, *Proceedings of the 2002 Congress on Evolutionary Computation (CEC02)*, Honolulu, 1151-1156.
- Bauer, A.; Bullnheimer, B.; Hartl, R.F. and Strauss, C.**, 1999. An ant colony optimization approach for the single machine total tardiness problem, *Proceedings of the 1999 Congress on Evolutionary Computation (CEC99)*, Washington D.C., USA, 6-9 July ,1445-1450.

- Baykasoğlu, A.; Dereli, T.; Erol, R. and Sabuncu, İ.,** 2003. An ant colony based optimization algorithm for solving assembly line balancing problems, *International XII. Turkish Symposium on Artificial Intelligence and Neural Networks-TAINN 2003*.
- Beasley, J.E.,** 1990. Distributing test problems by electronic mail, *Journal of the Operational Research Society*, **41**, 11, 1069-1072.
- Becerra, R.L. and Coello Coello, C.A.,** 2004. A cultural algorithm with differential evolution to solve constrained optimization problems, *Advances in Artificial Intelligence*, pp. 881-890, Eds. Lemaître, C.; Reyes, C.A. and González, J.A., Springer-Verlag.
- Blazewicz, J.; Ecker, K. H.; Pesch, E.; Schmidt, G. and Weglarz, J.,** 1996. Scheduling Computer and Manufacturing Processes, Springer-Verlag, Heidelberg.
- Blum, C.,** 2002. Ant Colony Optimization for FOP Shop scheduling: A case study on different pheromone representations, *Proceedings of the 2002 Congress on Evolutionary Computation (CEC02)*, 1558-1563.
- Blum, C.,** 2005. Beam-ACO—hybridizing ant colony optimization with beam search: an application to open shop scheduling, *Computers and Operations Research*, **32**, 6, 1565-1591.
- Blum, C. and Roli, A.,** 2003. Metaheuristics in combinatorial optimization: overview and conceptual comparison, *ACM Computing Surveys*, **35**, 3, 268-308.
- Bonabeau, E.; Dorigo, M. and Theraulaz, G.,** 1999. Swarm intelligence: from natural to artificial systems, Oxford University Press, New York, NJ.
- Brah, S.A. and Hunsucker J.L.,** 1991. Branch and bound algorithm for the flow shop with multiple processors, *European Journal Of Operational Research*, **51**, 88-99.
- Bullnheimer, B.; Hartl, R. F. and Strauss, C.,** 1999a. A new rank-based version of the ant system: A computational study, *Central European Journal for Operations Research and Economics*, **7**, 1, 25-38.
- Bullnheimer, B.; Hartl, R. F. and Strauss, C.,** 1999b. Applying the ant system to the vehicle routing problem, *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, pp. 285-296, Eds. Voß, S.; Martello, S.; Osman, I.H. and Roucairol, C., Kluwer Academic Publishers, Dordrecht.
- Bullnheimer, B.; Hartl, R.F. and Strauss, C.,** 1999c. An improved ant system algorithm for the vehicle routing problem, *Annals of Operations Research*, **89**, 319-328.
- Cao, J. and Bedworth, D.D.,** 1992. Flow shop scheduling in serial multi-product processes with transfer and set-up times, *International Journal of Production Research*, **30**, 1819-1830.
- Caraffa, V.; Ianes, S.; Bagchi, T.P. and Sriskandarajah, C.,** 2001. Minimizing makespan in a blocking flowshop using genetic algorithms, *International Journal of Production Economics*, **70**, 2, 101-115.

- Cardoso, P.; Jesus, M. and M'arquez, A., 2003.** Multiple objective TSP based on ACO, *III Encuentros Andaluzes de Matemáticas Discretas*, Universidade de Almeria.
- Cartwright, H.M. and Tuson, A.L., 1994.** Genetic algorithms and flow shop scheduling: towards the development of a real-time process control system, *Proceedings of the AISB workshop on Evolutionary Computing*, Leeds, UK, 177-290.
- Cavalieri, S. and Gaiardelli, P., 1998.** Hybrid genetic algorithms for a multiple-objective scheduling problem, *Journal of Intelligent Manufacturing*, **9**, 4, 361-367.
- Chakravarty, K. and Rajendran, C., 1999.** A heuristic for scheduling in flowshop with bicriteria of makespan and maximum tardiness minimization, *Production Planning and Control*, **10**, 707-714.
- Chan, W.T. and Hu, H., 2001.** An application of genetic algorithms to precast production scheduling, *Computers and Structures*, **79**, 17, 1605-1616.
- Chan, W.T. and Hu, H., 2002.** Production scheduling for precast plants using a flowshop sequencing model, *Journal of Computing in Civil Engineering*, **16**, 3, 165-174.
- Chang, P.-C.; Hsieh, J.-C. and Lin, S.-G., 2002.** The development of gradual-priority weighting approach for the multi-objective flowshop scheduling problem, *International Journal of Production Economics*, **79**, 3, 171-183.
- Chen, C.L. and Bulfin, R.L., 1993.** Complexity of single machine, multi-criteria scheduling problem, *European Journal Of Operational Research*, **70**, 115-125.
- Chen, C.L.; Vempati, V.S. and Aljaber, N., 1995.** An application of genetic algorithms for flow shop problems, *European Journal Of Operational Research*, **80**, 389-396.
- Chen, S. and Smith, S.F., 1999.** Improving genetic algorithms by search space reductions (with application to flowshop scheduling), *Proceedings of the Genetic and Evolutionary Computation Conference*, 135-140.
- Cheng, R. and Gen, M., 1997.** Parallel machine scheduling problems using memetic algorithms, *Computers and Industrial Engineering*, **33**, 3-4, 761-764.
- Cheng, T.C.E. and Wang, G., 2000.** An improved heuristic for the two-machine flowshop scheduling with an availability constraint, *Operations Research Letters*, **26**, 223-229.
- Cleveland, G.A. and Smith, F.S., 1989.** Using genetic algorithm to schedule flowshop release, *Proceedings of the 3rd Int. Conf. On Genetic Algorithms Applications*, 160-169.
- Colorni, A.; Dorigo, M. and Maniezzo, V., 1991.** Distributed optimization by ant colonies, *Proceedings of ECAL91-European Conference on Artificial Life*, Paris, France, 134-142.

- Colorni, A.; Dorigo, M. and Maniezzo, V., 1992.** An investigation of some properties of an ant algorithm, *Proceedings of the Parallel Problem Solving from Nature Conference (PPSN 92)*, Brussels, Belgium, 509-520.
- Colorni, A.; Dorigo, M.; Maniezzo, V. and Trubian, M., 1994.** Ant system for job-shop scheduling, *JORBEL - Belgian Journal of Operations Research, Statistics and Computer Science*, **34**, 1, 39-53.
- Cordon, O.; Fernandez de Viana, I.; Herrera, F. and Moreno, L., 2000.** A new ACO model integrating evolutionary computation concepts: The best-worst ant system, *Abstract Proceedings of ANTS2000-From Ant Colonies to Artificial Ants: A Series of International Workshops on Ant Algorithms*, pp. 22-29, Eds. Dorigo, M.; Middendorf, M. and Stützle, T., Université Libre de Bruxelles.
- Cordon, O.; Herrera, F. and Stützle, T., 2002.** A review on the ant colony optimization metaheuristic: basis, models and new trends, *Mathware and Soft Computing*, **9**, 2-3, 141-175.
- Cotta, C. and Troya, J. M., 1998.** Genetic form recombination in permutation flowshop problems, *Evolutionary Computation*, **6**, 1, 25-44.
- Daniels, R.L. and Chambers, R.J., 1990.** Multiobjective flowshop scheduling, *Naval Research Logistics Quarterly*, **37**, 981-995.
- Dannenbring, D.G., 1977.** An evaluation of flowshop sequencing heuristics, *Management Science*, **23**, 11, 1174-1182.
- de Castro, L.N., 2002.** Immune, swarm and evolutionary algorithms Part I: basic models, *Proceedings of the ICONIP Conference (International Conference on Neural Information Processing)*, Singapura, **3**, 1464-1468.
- de Castro, L.N. and Timmis, J., 2002.** Artificial immune systems: a novel paradigm to pattern recognition. *Artificial Neural Networks in Pattern Recognition*, pp. 67-84, Eds. Corchado, J.M.; Alonso, L. and Fyfe, C., University of Paisley, UK.
- den Besten, M., 2000.** Ants for the single machine total weighted tardiness scheduling problem, *Ms Thesis*, University of Amsterdam, Amsterdam.
- den Besten, M.; Stützle, T. and Dorigo, M., 2000.** Ant colony optimization for the total weighted tardiness problem. *Proceedings of PPSN-VI, Sixth International Conference on Parallel Problem Solving from Nature*, volume 1917 of Lecture Notes in Computer Science, pp. 611-620, Eds. Deb et al., Springer Verlag, Berlin, Germany.
- Diaz, F.R. and French, S., 1983.** A survey of multi-objective combinatorial scheduling, *Multi-objective Decision Making*, 59-77.
- Dorigo, M. and Di Caro, G., 1999.** The ant colony optimization meta-heuristic, *New Ideas in Optimization*, pp. 11-32, Eds. Corne, D.; Dorigo, M. and Glover, F., McGraw-Hill, London, UK.
- Dorigo, M. and Gambardella, L.M., 1997a.** Ant colonies for the traveling salesman problem, *BioSystems*, **43**, 73-81.

- Dorigo, M. and Gambardella, L.M.**, 1997b. Ant colony system: A cooperative learning approach to the traveling salesman problem, *IEEE Transactions on Evolutionary Computation*, 1, 1, 53-56.
- Dorigo, M.; Di Caro, G. and Gambardella, L.M.**, 1999. Ant Algorithms for Discrete Optimization, *Artificial Life*, 5, 2, 137-172.
- Dorigo, M.; Maniezzo, V. and Colorni, A.**, 1991a. Positive feedback as a search strategy, *Technical Report*, No. 91-016, Politecnico di Milano, Italy.
- Dorigo, M.; Maniezzo, V. and Colorni, A.**, 1991b. The ant system: An autocatalytic optimizing process, *Technical Report*, No. 91-016 (Revised), Politecnico di Milano, Italy.
- Dorigo, M.; Maniezzo, V. and Colorni, A.**, 1996. The ant system: Optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26, 1, 29-41.
- Dubek, R.A.; Panwalker, S.S. and Smith, M.L.** 1992. The lesson of flowshop scheduling research, *Operations Research*, 40, 1, 7-13.
- Fink, A. and Voß, S.**, 2003. Solving the continuous flow-shop scheduling problem by metaheuristics, *European Journal Of Operational Research*, 400-414.
- Framinan, J. M.; Leisten, R. and Ruiz-Usano, R.**, 2005. Comparison of heuristics for flowtime minimisation in permutation flowshops, *Computers and Operations Research*, 32, 5, 1237-1254.
- Gagné, C.; Gravel, M. and Price, W.**, 2000. Scheduling a single machine where setup times are sequence dependent using an ant-colony heuristic, *Abstract Proceedings of ANTS'2000*, Brussels, Belgium, 157-160.
- Gagné, C.; Gravel, M. and Price, W.**, 2001. A look-ahead addition to the ant colony optimization metaheuristic and its application to an industrial scheduling problem, *Proceedings of the 4th Metaheuristics International Conference*, Porto, Portugal, 16-20 july, 52, 7, 896-905.
- Gagné, C.; Price, W. and Gravel, M.**, 2002. Comparing an ACO algorithm with other heuristics for the single machine scheduling problem with sequence dependent setup times, *Journal of the Operational Research Society*, 53, 8, 896-906.
- Gambardella, L.M. and Dorigo, M.**, 1995. Ant-Q: A reinforcement learning approach to the traveling salesman problem, *Proceedings of ML-95, Twelfth International Conference on Machine Learning*, pp. 252-260, Eds. Frieditis, A. and Russell, S., Morgan Kaufmann, Tahoe City, CA.
- Gambardella, L.M. and Dorigo, M.**, 1996. Solving symmetric and asymmetric TSPs by ant colonies, *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation (ICEC'96)*, IEEE Press, Piscataway, NJ, 622-627.
- Gambardella, L.M. and Dorigo, M.**, 1997. HAS-SOP: An hybrid ant system for the sequential ordering problem, *Technical Report*, No. IDSIA 97-11, IDSIA, Lugano, Switzerland.

- Gambardella, L.M. and Dorigo, M.**, 2000. An ant colony system hybridized with a new local search for the sequential ordering problem, *INFORMS Journal on Computing*, **12**, 3, 237-255.
- Gambardella, L.M.; Taillard, E. and Agazzi, G.**, 1999. MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows, *New Ideas in Optimization*, pp. 63-76, Eds. Corne, D.; Dorigo, M. and Glover, F., McGraw-Hill, UK.
- Gambardella, L.M.; Taillard, E. and Dorigo, M.**, 1999. Ant colonies for the quadratic assignment problem, *Journal of the Operational Research Society*, **50**, 167-176.
- Gangadharan, R. and Rajendran, C.**, 1993. Heuristic algorithms for scheduling in the no-wait flowshop, *International Journal of Production Economics*, **32**, 285-290.
- Garey, M.R.; Johnson, D.S. and Sethi, R.**, 1976. The complexity of flowshop and job-shop scheduling, *Mathematics of Operations Research*, **1**, 2, 117-129.
- Gelders, L.F. and Samdandam, N.**, 1978. Four simple heuristics for scheduling a flowshop, *International Journal of Production Research*, **16**, 221-231.
- Gen, M. and Cheng, R.**, 2000. Genetic algorithms and engineering optimization, John Wiley&Sons, USA.
- Goldberg, E.D.**, 1989. Genetic algorithms in search, optimization and machine learning, Addison-Wesley Publishing Company, USA.
- Gonzalez, T.; Sahni, S.**, 1978. Flowshop and jobshop schedules: Complexity and approximations, *Operations Research*, **26**, 1, 36-52.
- Gottlieb, J., Puchta, M., and Solnon, C.**, 2003, A study of greedy, local search and ant colony optimization approaches for car sequencing problems, *Lecture Notes in Computer Science*, **2611**, 247-258.
- Gravel, M.; Price, W.L. and Gagné, C.**, 2002. Scheduling continuous casting of aluminum using a multiple objective ant colony optimization metaheuristic, *European Journal of Operational Research*, **143**, 1, 218-229.
- Guerrieri, B.**, 2000. Use of a colony of cooperating agents and MAPLE to solve the traveling salesman problem, *Electronic Proceedings of the Thirteenth Annual International Conference on Technology in Collegiate Mathematics*.
- Gupta, J.N.D.; Hennig, K. and Werner, F.**, 2001. Minimizing total flowtime in a 2-machine flow shop problem with minimum makespan, *International Journal of Production Economics*, **69**, 323-338.
- Gupta, J.N.D.; Hennig, K. and Werner, F.**, 2002. Local search heuristics for two-stage flow shop problems with secondary criterion, *Computers and Operations Research*, **29**, 2, 123-149.

- Gupta, J.N.D.; Kruger, K.; Lauff, V.; Werner, F. and Sotskov, Y.N., 2002.** Heuristics for hybrid flow shops with controllable processing times and assignable due dates, *Computers and Operations Research*, **29**, 10, 1417-1439.
- Gupta, J.N.D.; Palanimuthy, N. and Chen, C.L., 1999.** Desining a tabu search algorithm for the two-stage flowshop problem with secondary criterion, *Production Planning and Control*, **10**, 251-265.
- Gupta, J.N.D., 1969.** A improved combinatorial algorithm for the flowshop scheduling problem, *Operations Research*, **19**, 1753-1758.
- Hansen. P. and Mladenovic, N., 2003.** Variable neighborhood search, *Handbook of Metaheuristics*, pp. 145-184, Eds. Glover F. and Kochenberger, G., Kluwer Academic Publisher, Norwell, MA.
- Ho, J.C., 1995.** Flowshop sequencing with mean flowtime objective, *European Journal Of Operational Research*, **81**, 571-578.
- Ho, J.C. and Chang, Y.L., 1991.** A new heuristic for the n-job, m-machine flowshop problem, *European Journal Of Operational Research*, **52**, 194-202.
- Hundal, T.S. and Rajgopal, J., 1988.** An extension of Palmer' s heuristic for flowshop scheduling problem, *International Journal of Production Research*, **26**, 1119-1124.
- Ignall, E. and Schrage, L., 1965.** Application of the branch and bound technique to some flow shop scheduling problems, *Operations Research*, **13**, 3, 400-412.
- Ishibashi, H.; Aguirre, H.E.; Tanaka, K. and Sugimura, T., 2000.** Multi-objective optimization with improved genetic algorithm, *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 3852-3857.
- Ishibuchi, H. and Murata, T., 1998.** A multiobjective genetic local search algorithm and its applications to flowshop scheduling, *IEEE Transactions on Systems Man And Cybernetics-Part C: Applications and reviews*, **28**, 3, 392-403.
- Ishibuchi, H.; Misaki, S. and Tanaka, H., 1995.** Modified simulated annealing algorithms for the flow shop sequencing problem, *European Journal Of Operational Research*, **81**, 2, 388-398.
- Ishibuchi, H.; Yoshida, T. and Murata, T., 2003.** Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling, *Evolutionary Computation*, **7**, 2, 204-223.
- Iyer, S.K. and Saxena, B., 2004.** Improved genetic algorithm for the permutation flowshop scheduling problem, *Computers and Operations Research*, **31**, 4, 593-606.
- Janiak, A. and Portmann, M.C., 1998.** Genetic algorithm for the permutation flow-shop scheduling problem with linear models of operations. *Annals of Operations Research*, **83**, 95-114.

- Johnson, S.M.**, 1954. Optimal two- and three-stage production schedules with setup times included, *Naval Research Logistics Quarterly*, **1**, 1, 61-68.
- Kennedy, J.; Eberhart, R.C. and Shi, Y.**, 2001. Swarm intelligence, Morgan Kaufmann, San Mateo, CA, USA.
- Kim, Y.D.**, 1995. Minimizing total tardiness in permutation flowshops, *European Journal Of Operational Research*, **85**, 541-555.
- King, J.R. and Spachis, A.S.**, 1980. Heuristics for flowshop scheduling, *International Journal of Production Research*, **18**, 3, 345.
- Koulamas, C.**, 1998. An new constructive heuristic for flowshop scheduling problem, *European Journal Of Operational Research*, **105**, 66-71.
- Krishnaiyer, K. and Cheraghi, S.H.**, 2002. Ant algorithms: Review and future applications, *Proceedings Industrial Engineering Research Conference (IERC '02)*, Orlando, Florida, USA, May 2002.
- Kumar, S.; Bagchi, T.P. and Sriskandarajah, C.**, 2000. Lot streaming and scheduling heuristics for m-machine no-wait flowshops, *Computers and Industrial Engineering*, **38**, 1, 149-172.
- Lai, T.-C.**, 1996. The note on heuristics of flow shop scheduling, *Operations Research*, **44**, 648-652.
- Lampinen, J.**, 1999. Differential evolution-New naturally parallel approach for engineering design optimization, *Developments in Computational Mechanics with High Performance Computing*, pp. 217-228, Ed. Barry H.V. Topping, Civil-Comp Press, Edinburgh (Scotland).
- Lampinen, J. and Zelinka, I.**, 2000. On stagnation of the differential evolution algorithm, *Proceedings of MENDEL 2000, 6th International Mendel Conference on Soft Computing*, pp. 76-83, Ed. Pavel, O., Brno, Czech Republic, June 2000.
- Lee, C.Y.**, 1997. Minimizing the makespan in the two-machine flowshop scheduling problem with an availability constraint, *Operations Research Letters*, **20**, 129-139.
- Lee, I. and Shaw, K.J.**, 2000. A neural-net approach to real time flow-shop sequencing, *Computers and Industrial Engineering*, **38**, 1, 125-147.
- Leguizamón, G. and Schütz, M.**, 2003. An improved ant colony algorithm for the job shop scheduling problem, *Proceedings VIII Congreso Argentino de Ciencias de la Computación (CACIC 2002)*, Buenos Aires, Argentina, 617-627.
- Leu, S.-S. and Hwang, S.-T.**, 2002. GA-based resource-constrained flow-shop scheduling model for mixed precast production, *Automation in Construction*, **11**, 4, 439-452 .
- Li, Y. and Gong, S.**, 2003. Dynamic ant colony optimisation for TSP, *International Journal of Advanced Manufacturing Technology*, **22**, 7-8, 528-533.
- Loirenço, H.L.**, 1996. Sevast' janos' s algorithms for the flowshop scheduling problem, *European Journal Of Operational Research*, **91**, 176-189.

- Lourenço, H.R.; Martin, O. and Stützle, T., 2003.** Iterated local search, *Handbook of Metaheuristics*, pp. 321-353, Eds. Glover, F. and Kochenberger, G., Kluwer Academic Publisher, Norwell, MA.
- Mabed, M.H.; Rahoual, M.; Talbi, E.-G. and Dhaenens, C., 2000.** Using genetic algorithms to schedule multicriteria flow-shop, *PPSN/SAB Workshop on Multiobjective Problem Solving from Nature (MPSN)*, Paris.
- Maniezzo, V., 1999.** Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem, *INFORMS Journal on Computing*, **11**, 4, 358-369.
- Maniezzo, V. and Colorni, A., 1999.** The ant system applied to the quadratic assignment problem, *IEEE Transactions on Data and Knowledge Engineering*, **11**, 5, 769-778.
- Maniezzo, V.; Colorni, A. and Dorigo, M., 1994.** The ant system applied to the quadratic assignment problem, *Technical Report, No. IRIDIA/94-28*, Université Libre de Bruxelles, Belgium.
- Maniezzo, V.; Gambardella, L. and De Luigi, F., 2003.** Ant colony optimization, *New Optimization Techniques in Engineering*, pp. 101-117, Eds. Onwubolu, G.C. and Babu, B.V., Springer-Verlag.
- Mariano Romero, C.E. and Morales Manzanares, E., 1999.** MOAQ an Ant-Q algorithm for multiple objective optimization problems, *Genetic and Evolutionary Computing Conference (GECCO 99)*, **1**, pp. 894-901, Eds. Banzhaf, W.; Daida, J.; Eiben, A.E.; Garzon, M.H.; Honavar, V.; Jakiela, M. and Smith, R.E., San Francisco, California, Morgan Kaufmann.
- McMahon, G.B. and Burton, P., 1967.** Flowshop scheduling with branch and bound method, *Operations Research*, **15**, 473-481.
- McMullen, P.R., 2001.** An ant colony optimization approach to addressing a JIT sequencing problem with multiple objectives, *Artificial Intelligence in Engineering*, **15**, 309-317.
- McMullen, P.R. and Tarasewich, P., 2003.** Using ant techniques to solve the assembly line balancing problem, *IIE Transactions*, **35**, 7, 605-617.
- Merkle, D. and Middendorf, M., 2000.** An ant algorithm with a new pheromone evaluation rule for total tardiness problems, *Proceedings of the EvoWorkshops 2000*, volume **1803** of Lecture Notes in Computer Science, pp. 287-296, Springer-Verlag, Berlin.
- Merkle, D. and Middendorf, M., 2001a.** A new approach to solve permutation scheduling problems with ant colony optimization, *Applications of Evolutionary Computing: Proceedings of EvoWorkshops 2001*, volume **2037** of Lecture Notes in Computer Science, pp. 484-493, Eds. Boers, E.J.W. et al., Springer Verlag, Lake Como, Italy.
- Merkle, D. and Middendorf, M., 2001b.** On the behaviour of ant algorithms: Studies on simple problems, *Proceedings 4th Metaheuristics International Conference*, 573-577.

- Merkle, D. and Middendorf, M.**, 2003. An ant colony optimization with global pheromone evaluation for scheduling a single machine, *Applied Intelligence*, **18**, 1, 105-111.
- Merkle, D.; Middendorf, M. and Schmeck, H.**, 2000. Ant colony optimization for resource-constrained project scheduling, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, Morgan Kaufmann Publishers, San Francisco, CA, 893-900.
- Merz, P. and Freisleben, B.**, 1999. A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem, *Congress on Evolutionary Computation (Conference paper)*, 2063-2070.
- Michels, R. and Middendorf, M.**, 1998. An ant system for the shortest common supersequence problem, *Proceedings Fifth International Conference On Parallel Problem Solving From Nature (PPSN'98)*, Amsterdam, Netherlands, Springer Verlag, LNCS **1498**, 692-701.
- Morton, T.E. and Pentico, D.W.**, 1993. Heuristic scheduling systems, John Wiley, USA.
- Mulkens, H.**, 1994. Revisiting the Johnson algorithm for flowshop scheduling with genetic algorithms, *Knowledge-Based Reactive Scheduling*, **15**, 69-80.
- Murata, T. and Ishibuchi, H.**, 1997. Performance of multi-objective genetic algorithms for flowshop scheduling problems, *Proceedings of the 14th International Conference on Production Research*, Osaka, 498-501.
- Murata, T.; Ishibuchi, H. and Lee, K.H.**, 1996. Application of two-objective genetic algorithm to flowshop scheduling problems with interval processing time, *Proceedings of the EUFIT '96-Aachen*, Germany, 443-447.
- Murata, T.; Ishibuchi, H. and Tanaka, H.**, 1996. Genetic algorithms for flowshop scheduling problems, *Computers and Industrial Engineering*, **30**, 4, 1061.
- Nagano, M.S. and Moccellin, J.V.**, 2002. A high quality solution constructive heuristic for flow shop sequencing, *Journal of the Operational Research Society*, **53**, 12, 1374-1379.
- Nagar, A.; Haddock, J. and Heragu, S.S.**, 1995, Multiple and bicriteria scheduling: A literature survey, *European Journal Of Operational Research*, **81**, 88
- Nagar, A.; Heragu, S.S. and Haddock, J.**, 1996. A combined branch and bound and genetic algorithm based approach for a flowshop scheduling problem, *Annals of Operations Research*, **63**, 397-414.
- Nawaz, M.; Ensore, Jr., E. and Ham, I.**, 1983. A heuristic algorithm for the m-machine, n-job flowshop sequencing problem, *OMEGA*, **11**, 1, 91-95.
- Neppalli, V.R.; Chen, C.-L. and Aljaber, N.J.**, 1994. An effective heuristic for the flowshop problem with weighted tardiness, *Proceedings of the 3rd Industrial Engineering Research Conference*, 634-638.

- Neppalli, V.R.; Chen, C.-L. and Gupta, J.N.D.**, 1996. Genetic algorithms for the two-stage bicriteria flowshop problem, *European Journal Of Operational Research*, **95**, 2, 356-373.
- Nowichi, E. and Smutnicki, C.**, 1996. A fast tabu search algorithm for the permutation flow shop problem, *European Journal Of Operational Research*, **91**, 160-175.
- Ogbu, F.A. and Smith, D.K.**, 1991. Simulated annealing for the permutation flowshop problem, *OMEGA*, **19**, 64-67.
- Oğuz, C.; Janiak, A. and Lichtenstein, M.**, 2001. Metaheuristic algorithms for hybrid flow-shop scheduling problem with multiprocessor tasks, *Proceedings of the MIC'2001 - 4th Metaheuristics International Conference*, 477-481.
- Oğuz, C.; Zinder, Y.; Ha Do, V.; Janiak, A. and Lichtenstein, M.**, 2004. Hybrid flow-shop scheduling problems with multiprocessor task systems, *European Journal Of Operational Research*, **152**, 115-131.
- Onwubolu, G.C. and Mutingi, M.**, 1999. Genetic algorithm for minimizing tardiness in flow-shop scheduling, *Production Planning and Control*, **10**, 5, 462-471.
- Palmer, D.S.**, 1965. Sequencing jobs through a multi-stage process in the minimum total time-a quick method of obtaining near optimum, *Journal of the Operational Research Society*, **16**, 1, 101-107.
- Panwalkar, S.S. and Khan, A.W.**, 1976. A ordered flowshop sequencing problem with mean completion time criterion, *International Journal of Production Research*, **14**, 631-635.
- Pilat, M. and White, T.**, 2002. Using genetic algorithms to optimize ACS-TSP, *Proceedings of the 3rd International Workshop on Ant Algorithms*, Brussels, Belgium, September 12-14.
- Portmann, M.-C.; Vignier, A.; Dardilhac, D. and Dezalay, D.**, 1998. Branch and bound crossed with GA to solve hybrid flowshops, *European Journal Of Operational Research*, **107**, 2, 389-400.
- Pugazhendhi, S.; Thiagarajan, S.; Rajendran, C. and Anantharaman, N.**, 2002. Performance enhancement by using non-permutation schedules in flowline-based manufacturing systems, *Computers and Industrial Engineering*, **44**, 133-157.
- Rajendran, C.**, 1992. Two-stage flowshop scheduling problem with bicriteria, *Journal of the Operational Research Society*, **43**, 9, 871-884.
- Rajendran, C.**, 1993. Heuristic algorithm for scheduling in a flowshop to minimize total flowtime, *International Journal of Production Economics*, **29**, 65-73.
- Rajendran, C.**, 1994a. A heuristic algorithm for scheduling in flowshop and flowline-based manufacturing cell with multi-criteria, *International Journal of Production Research*, **32**, 2541-2558.
- Rajendran, C.**, 1994b. A no wait -flowshop scheduling heuristic to minimize makespan, *Journal of the Operational Research Society*, **45**, 472-478.

- Rajendran, C.**, 1995. Heuristics for scheduling in flowshop with multiple objectives, *European Journal Of Operational Research*, **82**, 540-555.
- Rajendran, C. and Chaudhuri, D.**, 1991. An effecient heuristic approach to the scheduling of jobs in flowshop, *European Journal Of Operational Research*, **61**, 3, 318-325.
- Rajendran, C. and Ziegler, H.**, 2004. Ant-colony algorithms for flowshop scheduling to minimize makespan/total flowtime of jobs, *European Journal Of Operational Research*, **155**, 2, 426-438.
- Reeves, C.R.**, 1995. A genetic algorithm for flow shop sequencing, *Computers and Operations Research*, **22**, 1, 5-13.
- Reeves, C.R. and Yamada, T.**, 1998. Genetic algorithms, path relinking and the flow shop sequencing problem, *Evolutionary Computation*, **6**, 1, 230-234.
- Riezebos, J. and Gaalman, G.J.C.**, 1998. Time lag size in multiple operations flow shop scheduling heuristics, *European Journal Of Operational Research*, **105**, 1, 72-90.
- Riezebos, J.; Gaalman, G.J.C. and Gupta, J.N.D.**, 1995. Flow shop scheduling with multiple operations and time lags, *Journal of Intelligent Manufacturing*, **6**, 105-115.
- Sayın, S. and Karabatı, S.**, 1999. A bicriteria approach to the two-machine flowshop scheduling problem, *European Journal Of Operational Research*, **112**, 435-449.
- Silva, J.D.L.**, 2003. Metaheuristic and multiobjective approaches for space allocation, University of Nottingham, *PhD Thesis*, School of Computer Science and Information Technology.
- Sivrikaya-Serifoğlu, F. and Ulusoy, G.**, 1998. A bicriteria two-machine permutation flowshop problem, *European Journal Of Operational Research*, **107**, 414-430.
- Smith, M.L. and Dubek, R.A.**, 1967. A general algorithm for solution of the n-job, m-machine sequencing problem of the flowshop, *Operations Research*, **15**, 71-82.
- Solimanpur, M.; Vrat, P. and Shankar, R.**, 2005. An ant algorithm for the single row layout problem in flexible manufacturing systems, *Computers and Operations Research*, **32**, 3, 583-598.
- Solimanpur, M.; Vrat, P. and Shankar, R.**, 2003b. Ant colony optimization algorithm to the inter-cell layout problem in cellular manufacturing, *European Journal of Operational Research*, **157**, 3, 592-606.
- Sridhar, C. and Rajendran, C.**, 1996. Scheduling in flowshop and cellular manufacturing systems with multiple objectives: A genetic algorithmic approach, *Production Planning and Control*, **7**, 374-382.
- Sriskandarajah, C. and Sethi, S.P.**, 1989. Scheduling algorithms for flexible flowshops: Worst and average case performance, *European Journal Of Operational Research*, **43**, 143-160.

- Storn, R. and Price, K.**, 1997. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization*, 11, 4, 341–359.
- Stöppler, S. and Bierwirth, C.**, 1992. The application of a parallel genetic algorithm to the n/m/P/C_{max} flowshop problem, *New Directions for Operational Research in Manufacturing*, 161-179.
- Stützle, T.**, 1997. MAX-MIN ant system for the quadratic assignment problem, *Technical Report, No. AIDA-97-04*, Darmstadt University of Technology, Computer Science Department, Intellectics Group.
- Stützle, T.**, 1998a. An ant approach to the flow shop problem, *Proceedings of the 6th European Congress on Intelligent Techniques and Soft Computing (EUFIT' 98)*, Verlag Mainz, Wissenschaftsverlag, Aachen, Germany, 1560-1564.
- Stützle, T.**, 1998b. Applying iterated local search to the permutation flow shop problem, *Technical Report, AIDA-98-04*, Darmstadt University of Technology, Computer Science Department, Intellectics Group.
- Stützle, T. and Dorigo, M.**, 1999a. ACO algorithms for the quadratic assignment problem, *New Ideas in Optimization*, pp. 33-50, Eds. D. Corne, M. Dorigo and F. Glover, McGraw-Hill, London, UK.
- Stützle, T. and Dorigo, M.**, 1999b. ACO algorithms for the traveling salesman problem, *Evolutionary Algorithms in Engineering and Computer Science*, pp. 163-183, Eds. Miettinen, K.; Makela, M.; Neittaanmaki, P. and Periaux, J., Wiley, New York.
- Stützle, T. and Dorigo, M.**, 2003. The ant colony optimization metaheuristic: Algorithms, applications, and advances, *Handbook of Metaheuristics*, pp. 251-285, Eds. Glover, F. and Kochenberger, G., Kluwer Academic Publishers, Norwell, MA.
- Stützle, T. and Hoos, H.**, 1996. Improving the ant system: A detailed report on MAX-MIN ant system, *Technical Report, AIDA-96-12* (Revised version), Darmstadt University of Technology, Computer Science Department, Intellectics Group.
- Stützle, T. and Hoos, H.**, 1997. The MAX-MIN ant system and local search for the traveling salesman problem, *Proceedings of the IEEE International Conference on Evolutionary Computation*, Indianapolis, Indiana, USA, 308-313.
- Stützle, T. and Hoos, H.**, 1998a. Improvements on the ant system: Introducing the MAX-MIN ant system, *Proceedings of Artificial Neural Nets and Genetic Algorithms 1997*, pp. 245-249, Eds. Smith, G.D.; Steele, N.C. and Albrecht, R.F., Springer Verlag, Wien.
- Stützle, T. and Hoos, H.**, 1998b. The MAX-MIN ant system and local search for combinatorial optimization problems, *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, pp. 313-329, Eds. Voss, S.; Martello, S.; Osman, I.H. and Roucairol, C., Kluwer Academic Publishers.

- Stützle, T. and Hoos, H.**, 2000. MAX-MIN ant system, *Future Generation Computer Systems Journal*, **16**, 8, 889-914.
- Szwarc, W.**, 1973. Optimal elimination methods in the mxn flowshop scheduling problem, *Operations Research*, **21**, 1250-1259.
- Tagami, T. and Kawabe, T.**, 1998. Genetic algorithms with a Pareto partitioning method for multiobjective flowshop scheduling, *Proceedings of the 1998 International Symposium of Nonlinear Theory and its Applications (NOLTA'98)*, Crans-Montana, 1069-1072.
- Tagami, T. and Kawabe, T.**, 1999. Genetic algorithms based on Pareto neighborhood search for multiobjective optimization, *Proceedings of the 1999 International Symposium of Nonlinear Theory and its Applications (NOLTA'99)*, Hawaii, 331-334.
- Taillard, E.**, 1990. Some efficient heuristic methods for the flowshop-sequencing problem, *European Journal Of Operational Research*, **47**, 1, 65-74.
- Taillard, E.**, 1993. Benchmarks for basic scheduling problems, *European Journal Of Operational Research*, **64**, 278-285.
- Talbi, E.-G.; Rahoual, M.; Mabed, M.H. and Dhaenens, C.**, 2001. A hybrid evolutionary approach for multicriteria optimization problems: Application to the flow shop, *Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization*, 416-428.
- Talbi, E.-G.; Roux, O.; Fonlupt, C. and Robillard, D.**, 2001. Parallel ant colonies for the quadratic assignment problem, *Future Generation Computer Systems*, **17**, 441-449.
- Tamura, H.; Shibata, T.; Tomiyama, S. and Hatono, I.**, 1999. A meta-heuristic satisficing tradeoff method for solving multiobjective combinatorial optimization problems-with application to flowshop scheduling-, *IEEE International Conference on Systems, Man, and Cybernetics*, **3**, 539-544.
- Tang, L. and Liu, J.**, 2002. A modified genetic algorithm for the flow shop sequencing problem to minimize mean flow time, *Journal of Intelligent Manufacturing*, **13**, 61-67.
- Thornton, H.W. and Hunsucker, J.L.**, 2004. A new heuristic for minimal makespan in flow shops with multiple processors and no intermediate storage, *European Journal Of Operational Research*, **152**, 96-114.
- T'Kindt, V.; Gupta, J.N.D. and Billaut, J.-C.**, 2003. Two-machine flowshop scheduling with a secondary criterion, *Computers and Operations Research*, **30**, 4, 505-526.
- T'Kindt, V.; Monmarche, N.; Laugt, D. and Tercinet, F.**, 2000. Combining ant colony optimization and simulated annealing to solve a 2-machine flowshop bicriteria scheduling problem, *Proceedings of the European Chapter on Combinatorial Optimization*, Capri, Italy, 129-130.
- T'Kindt, V.; Monmarché, N.; Tercinet, F. and Laügt, D.**, 2002. An ant colony optimization algorithm to solve a 2-machine bicriteria flowshop scheduling problem, *European Journal Of Operational Research*, **142**, 250-257.

- Tsai, C.-F.; Tsai, C.-W. and Tseng, C.-C.**, 2003. A new and efficient ant-based heuristic method for solving the traveling salesman problem, *Expert Systems*, **20**, 4, 179-186.
- Turner, B. and Booth, D.**, 1987. Comparison of heuristic flowshop sequencing, *OMEGA*, **15**, 1, 75-78.
- van den Bergh, F.**, 2001. An analysis of particle swarm optimizers, *PhD Thesis*, Faculty of Natural and Agricultural Science University of Pretoria, Pretoria.
- Voudouris, C.**, 1997. Guided local search for combinatorial optimisation problems, *PhD Thesis*, Department of Computer Science University of Essex.
- Wang, C.; Chu, C. and Proth, J.M.**, 1997. Heuristic approaches for n/m/F/ΣCi scheduling problem, *European Journal Of Operational Research*, **96**, 636-644.
- Wang, L. and Zheng, D.Z.**, 2003. An effective hybrid heuristic for flow shop scheduling, *International Journal of Advanced Manufacturing Technology*, **21**, 1, 38-44.
- Werner, F.**, 1993. On the heuristic solution of the permutation flow shop problem by path algorithms, *Computers and Operations Research*, **20**, 7, 707-722.
- Widmer, M. and Hertz, A.**, 1989. A new heuristic method for the flowshop sequencing problem, *European Journal Of Operational Research*, **41**, 186-193.
- Wilson, J.M.**, 1989. Alternative formulations of a flowshop scheduling problem, *Journal of Operations Research Society*, **40**, 4, 395-399.
- Wismer, M.**, 1972. Solution of flowshop scheduling problem with no intermediate queues, *Operations Research*, **20**, 689-697.
- Woo, H.S. and Yim, D.S.**, 1998. A heuristic algorithm for mean total flowtime objective in flowshop scheduling, *Computers and Operations Research*, **25**, 3, 175-182.
- Yamada, T. and Reeves, C.**, 1997. Permutation flowshop scheduling by genetic local search, *Proceedings of the 2nd IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems (GALESIA' 97)*, London, UK, 232-238.
- Yamada, T. and Reeves, C.**, 1998. Solving the C_{sum} permutation flowshop scheduling problem by genetic local search, *Proceedings of the 1998 IEE International Conference on Evolutionary Computation*, 230-234.
- Yeh, W.-C.**, 1999. A new branch-and-bound approach for the n/2/flowshop/αF+βCmax flowshop scheduling problem, *Computers and Operations Research*, **26**, 13, 1293-1310.
- Yeh, W.-C.**, 2002. A memetic algorithm for the n/2/Flowshop/φF+γCmax scheduling problem, *The International Journal of Advanced Manufacturing Technology*, **20**, 6.
- Ying, K.-C. and Liao, C.-J.**, 2003. An ant colony system approach for scheduling problems, *Production Planning and Control*, **14**, 1, 68-75.

- Ying, K.-C. and Liao, C.-J.**, 2004. An ant colony system for permutation flow-shop sequencing, *Computers and Operations Research*, **31**, 5, 791-801.
- Yong, Z. and Sannomiya, N.**, 2001. An improvement genetic algorithms by search space reductions in solving large-scale flowshop problems, *T.IEE Japan*, **121-C**, 6, 1010-1015.
- Yoon, S.H. and Ventura, J.A.**, 2002. An application of genetic algorithms to lot-streaming flow shop scheduling, *IIE Transactions*, **34**, 9, 779-787.
- Yu, I.K. and Song, Y.H.**, 2001. A novel short-term generation scheduling technique of thermal units using ant colony search algorithms, *Electrical Power and Energy Systems*, **23**, 6, 471-479.
- Zegordi, S.H., Itoh, K., and Enkawa, T.**, 1995. Minimizing makespan for flow shop scheduling by combining simulated annealing with sequencing knowledge, *European Journal Of Operational Research*, **85**, 3, 515-531.
- Zhao, Y.; Brizuela, C.A. and Sannomiya, N.**, 2001. Application of a partial enumeration selection method in genetic algorithms to solving a multi-objective flowshop problem, *IEEE International Conference on Systems, Man, and Cybernetics*, **4**, 2365-2370.
- Zhixin, L.; Jianguo, L.; Jinxing, X. and Jiefang, D.**, 2002. A heuristic for two-stage no-wait hybrid flowshop scheduling with a single machine in either stage, *The Journal Tsinghua Science and Technology*, **8**, 1, 43-48

<http://iridia.ulb.ac.be/~mdorigo/aco/aco.html>

<http://www.idsia.ch/~eric/>

OR-Library: <http://mscmga.ms.ic.ac.uk/info.html>

EKLER



Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Johnson, 1954	Eniyileme algoritması ve teorem-ispat	n -iş, 2- ve 3-makine akış tipi çizelgeleme problemi	Toplam elapsed zamanı en küçüklenmesi	2-makinelili problemi için bir çözüm algoritması önerilmiştir ve problemin formülasyonu teorem-ispat şeklinde sunulmaktadır. Ayrıca 3-makinelili problem için, özel durumlar altında çözüm önerilmiştir.	Çözüm yapısı sadece bir örnekle gösterilmiştir.	-
Ignall ve Schrage, 1965	Dal-sınır algoritması	n -iş, 2- ve 3-makine akış tipi çizelgeleme problemi	2-makine için: Ortalama tamamlama zamanı en küçüklenmesi. 3-makine için: Toplam tamamlama zamanı en küçüklenmesi	Öncelikle akış tipi probleminin ağaç yapısı ile nasıl gösterileceği verilmiştir. Daha sonra 3-makine için bir alt sınır fonksiyonu önerilmiş ve bir dal-sınır tekniği tanımlanmıştır. Ayrıca 2-makine için de bir alt sınır tanımlanmıştır. 2-makinelili problem için 9 işe kadar, 3-makinelili problem için 10 işe kadar algoritma test edilmiştir.	Elde edilen sonuçlar sunulmuştur.	2-makinelili problem için daha farklı amaç fonksiyonları ele alınabilir. 3- ve daha fazla makinelili problem için, makineler üzerinde aynı iş sıralamalarının olmadığı durumlar için çalışma yapılabilir.
Palmer, 1965	Sezgisel	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlama zamanı en küçüklenmesi	Probleminin çözümünü için, bir eğitim indeksi tanımlanmış ve buna bağlı bir yaklaşık çözüm önerilmiştir. 3-makinelili problem için 4-9 iş için sonuçlar verilmiştir. Daha büyük problemler için de deneyler yapılmıştır.	Önerilen yaklaşım eniyi çözüme oldukça yakın çözümler vermektedir.	Eğitim indeksine dayalı çizelgeleme, rasgele arama için veya aramayı güçlendirmek için düzenlenebilir.
McMahon ve Burton, 1967	Dal-sınır algoritması	n -iş, 3-makine akış tipi çizelgeleme problemi	Toplam tamamlama zamanı en küçüklenmesi	Problem için yeni işe-dayalı sınırlar önerilmiş ve bir dal-sınır tekniği tanımlanmıştır.	45 işe kadar elde edilen sonuçlar sunulmuştur. Oluşturulan kural algoritmanın etkinliğini %10' a kadar güçlendirmektedir. Bu etki büyük problemler için daha güçlüdür.	Gelişme, garanti edilememesine rağmen, sınır, makine sıralama ve iş sıralama kullanılarak elde edilen genel avantaj, küçük miktarda ön hesaplamaya değerdir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Smith ve Dubek, 1967	Sezgisel, teorem-ispata ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Probleminin çözümü için, bir karar kuralı tanımlanmış ve buna bağlı bir çözüm önerilmiştir. Çözüm yapısı bir örnek gösterilmiştir, ayrıca 3 ve 5 makine için ve 3-8 işli problemler üzerinde denemeler yapılmıştır. Sonuçlar makaleye zemin oluşturan Dubek ve Teuton tarafından geliştirilen algoritma sonuçları ve Ignall ve Schrage (1965) tarafından geliştirilen algoritma sonuçları ile karşılaştırılmıştır.	Sonuçlar önerilen yapının karşılaştırılan algoritmalar ile kıyaslamaya değer olduğunu göstermektedir.	1.Algoritma, gerekli kısıtlar probleme uygulandığında eniyi sıralama üretebilir. 2.İş sayısı arttıkça, uygun çözüm sayısı ve hesaplama için gerekli süre de artmaktadır, ancak, toplamda her birinin yüzdesi düşmektedir. 3.Algoritma, rahatlıkla diğer kriterler için de uygulanabilir.
Gupta, 1969	Birleşik algoritma ve deneysel hesaplama	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Temel baskınlık koşullarında modifikasyon yapılarak güçlendirilmiş algoritma üzerinde deneysel tasarım yapılmıştır.	Önerilen algoritmanın hesaplama süresi ve üretilen çizelgelerin ortalama sayısı Smith ve Dubek (1967) algoritmasından daha düşüktür.	Önerilen algoritma, baskın ve baskın olmayan koşullarda Smith ve Dubek algoritmasından daha etkindir.
Ashour, 1970	Deneysel hesaplama ve karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi Karşılaştırma kriterleri: 1.Çözüm kalitesi 2.Bilgisayar etkinliği	Literatürdeki mevcut akış tipi çizelgeme problemleri için önerilen: 1.Eniyileme teknikleri (switch-and-check, geri dönüşümlü dal-sınır (branch and bound with backtracking)) 2.Yarı-eniyileme teknikler (geri dönüşümlü dal-sınır (branch and bound without backtracking), modifiye bozulma (modified decomposition)) üzerinde test problemleri uygulanarak karşılaştırma yapılmıştır.	Eniyileme teknikleri hesaplama süresi açısından sadece küçük ölçekli problemler için etkindir. Yarı-eniyileme teknikler makul hesaplama süresi ile iyi sonuçlar üretmektedir. Modifiye bozulma tekniği iyi sonuçlar üretmektedir, ama çok fazla hesaplama süresi gerektirir. Ancak, büyük ölçekli problemler için eniyileme teknikleri çözüm kapasitesini artırır.	Lineer programlama gelecekte bir potansiyele sahip olmasına rağmen, çözüm sonuçları hesaplama etkinliğini garantileyen eniyi değerlere yeterli yakınlıkta görünmemektedir.
Wisner, 1972	Sezgisel algoritma	n -iş, m -makine bekleme olmayan akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Probleminin çözümü için, TSP gibi modellenen bir algoritma önerilmiştir. Çözüm yapısı bir örnekle gösterilmiş, ayrıca problemler üzerinde denemeler yapılmıştır.	Elde edilen sonuçlar sunulmuştur. İyi bir çözüm, arzu edilen bir zamanda oluşturulabilir ve bu ek hesaplamalar ile güçlendirilebilir.	Bu tip problemler gözönüne alındığında, bu çözüm oldukça avantajlıdır.

Tablo A.1 Tek-Amaçlı Akış Tipi Çözgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Szwarc, 1973	Eniyi eliminasyon yöntemi ve teorem-ispat	n -ış, m -makine akış tipi çözgeleme problemi	Toplam elapsed zamanı en küçüklenmesi	Probleminin çözümü için, eniyi eliminasyon yöntemi kullanılmaktadır. Çözüm yapısı bir örnekle gösterilmiştir.	-	-
Baker, 1975	Eniyileme algoritması ve literatür incelemesi	n -ış, m -makine akış tipi çözgeleme problemi	Toplamı tamamlama zamanı en küçüklenmesi	Probleminin çözümü için, önerilmiş çeşitli dal-sınır algoritmaları (B1,B2,B3,B24) ve eliminasyon stratejileri (Form1 ve Form2) test problemleri üzerinde karşılaştırılmıştır. Daha sonra yeni ve etkin çözüm algoritmaları oluşturmak üzere bunlar birleştirilmiştir.	En iyi dal-sınır algoritması B24 bulunmuştur. Dal sınır yaklaşımı ile eliminasyon stratejisi Form2' den oluşan birleşik bir algoritma oluşturulmuştur ve test problemleri üzerinde karşılaştırılmıştır. Sonuçlar, bu yaklaşımın büyük problemler için uygun olduğunu göstermiştir.	1. Tam dal-sınır yaklaşımı, eliminasyon yöntemleri ile genişletilebilir. 2. Birleşik sınırlar daha iyi sonuçlar verebilmektedir. 3. İyi bir sezgiselin başlangıç çözümü olarak kullanılması, tüm algoritma üzerinde faydalıdır.
Garey ve diğ., 1976	Teorem-ispat	n -ış, $m \geq 3$ ve $m \geq 2$ makine akış tipi çözgeleme problemleri	Uzunluk ve ortalama akış zamanı en küçüklenmesi	Akış tipi çözgelemenin kompleksliğini ortaya koymak için aşağıdaki teoremlerin ispatları yapılmıştır: 1. m -makinelili akış tipinde en kısa uzunluğu olan çözgeleyi belirlemek, $m \geq 3$ için NP-tamdır. 2. m -makinelili akış tipinde ortalama akış zamanlı çözgeleyi belirlemek, her $m \geq 2$ için NP-tamdır.	Sunulan teoremler ispatlanmış ve doğrulukları gösterilmiştir.	Makine sayısı arttığında, problem NP-tam olmaktadır.
Panwalkar ve Khan, 1976	Sezgisel algoritma ve teorem-ispat	n -ış, m -makine düzenli akış tipi sıralama problemi	Ortalama tamamlama zamanı en küçüklenmesi	Probleminin çözümü için, basit bir algoritma tanımlanmıştır. Daha sonra bunun eniyileme ispatı sunulmuştur.	Ortalama tamamlama zamanını en küçükleyen klasik akış tipi problemini çözmek için, etkin bir süreç olmamasına rağmen, özel olarak yapılandırılmış bir problem için, böyle bir süreç geliştirmek mümkündür.	-

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Dannenbring, 1977	Sezgisel, deneyssel tasarım ve karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlama zamanı en küçüklenmesi	8 tane literatürdeki ve 3 tane çalışmanın bir parçası olan 11 sezgisel üzerinde 3x3 ile 50x50 arasında değişen problemlerden oluşan toplam 1580 problem 11 sezgisel üzerinde denenmiş ve sonuçlar değerlendirme kriterlerine göre karşılaştırılmıştır.	Küçük problemler için, hızlı giriş (rapid access) çözümü; büyük problemler için, geniş arama ile hızlı giriş (rapid access with extensive search) çözümü; hesaplama zamanına göre, eğitim sırası (slope order) çözümü en iyi sonuçları verir.	1. Problem büyüklüğü arttıkça bağlı hata değeri azalma eğilimindedir. 2. Yüksek korelasyon olduğunda sezgisel kullanmanın faydası artmaktadır. 3. Performans veri hatalarından etkilenebilmektedir. 4. Çeşitli yöntemler arasındaki diferansiyel azaldıkça standart hata artar.
Gelders ve Samdandam, 1978	Sezgisel ve deneyssel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Bir kompleks maliyet fonksiyonunun en küçüklenmesi	Problemin çözümü için, dört basit sezgisel (sezgisel I, II, III, IV) tanımlanmıştır. Daha sonra bunlardan elde edilen sonuçlar, eniyi çözümlerle karşılaştırılmıştır.	Sezgisel II diğer yöntemlere göre daha iyi sonuçlar vermiştir. Sezgisel I ise diğer süreçlere göre hesaplama açısından daha caziptir. Ayrıca Sezgisel II ağırlıklı pozitif gecikme zamanı toplamının en küçüklenmesinde daha iyi bir yöntemdir. Maliyet fonksiyonunun tek başına en küçüklenmesinde ise Sezgisel I daha ekonomik ve iyidir.	Denemeler küçük ölçekli problemler üzerinde yapılmıştır. Büyük ölçekli problemler içinde aynı tartışma açığa çıkarmaktadır.
Gonzalez ve Sahni, 1978	Teorem-ispat	n -iş, m -makine akış tipi çizelgeleme problemi	Akış tipi için öncelikli ve önceliksiz çizelgeleri en küçükleyen bitiş zamanını belirlemenin NP-tam olduğunu ortaya koymak.	Aşağıdaki önermelerin ispatları yapılmış ve örneklerle açıklanmıştır: 1. $m > 2$ için eniyi bitiş zamanlı öncelikli ve önceliksiz çizelgeler bulmak, NP-tamdır. 2. İy çizelgeli üreten çeşitli sezgiseller üzerine sınırlar da konulmuştur.	Sunulan önermeler ispatlanmış ve doğrulukları gösterilmiştir.	İyi süreli veya iyi ortalama zamanlı çizelgeler üreten çeşitli sezgisellerin performansı üzerine sınırlar oluşturulabilir. Bu makine sayısı küçük olduğunda ($3 \leq m \leq 4$), bu sınırlar sıkıdır.

Tablo A.1 Tek-Amaçlı Akış Tipi Çözelleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
King ve Spachis, 1980	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine iş atlamasız ve iş beklemez akış tipi çözelleme problemi	Çözelleği tamamlamak için gerekli toplam zaman veya toplam tamamlama zamanı en küçültülmesi	5 sezgisel (LFD, LBD, LBJD, LWBID, MLSS) bu iki akış tipi problemi üzerinde test edilmiş ve RANDOM kuralı ile oluşturulmuş sonuçlarla karşılaştırılmıştır.	İş atlamasız durumda; büyük değişkenliğe sahip işlem süreli problemler için, LWBID(sc) sezgiseli en iyi, MLSS(sc) sezgiseli en kötü, küçük değişkenliğe sahip işlem süreli problemler için, çok açık olmamakla birlikte, LWBID(sc), LBJD(sc) ve CDS sezgiselleri en iyi, MLSS(sc), RANDOM ve LBD(sc) sezgiselleri en kötü sonuçları vermiştir. İş beklemez durumda; büyük değişkenliğe sahip işlem süreli problemler için, MCL(sc) sezgiseli en iyi, RANDOM sezgiseli en kötü, küçük değişkenliğe sahip işlem süreli problemler için, hata ve bağlı hata açısından MLSS(sc) sezgiseli, ortalama performans açısından MCL(sc) en iyi, RANDOM sezgiseli en kötü, çok açık olmamakla birlikte, LWBID(sc), LBJD(sc) ve CDS sezgiselleri en iyi, MLSS(sc), RANDOM ve LBD(sc) sezgiselleri en kötü sonuçları vermiştir.	İş atlamasız durumunda; LBJD(sc) ve CDS sezgiselleri hesaplama yapmak, birlikte kullanmak ve en iyi çözümü seçmek kolay olduğu için uygun görülmektedir. İş beklemez durumunda; MCL(sc), MLSS(sc) ve LBJD(sc) sezgiselleri en iyidir. Tek bir sezgisel seçmek gerekirse, büyük ölçekli problemler için MCL(sc) sezgiseli en iyidir.
Navaz ve diğ., 1983	Sezgisel	n için aynı sırada m makinede işlem görüğü ve en büyük toplam işlem zamanına sahip işin en küçük işlem zamanına sahip işten daha büyük önceliğe sahip olduğu varsayımı altındaki akış tipi sıralama problemi	Toplam tamamlama zamanı en küçültülmesi	Geliştirilen algoritmanın performansı en iyi performansla sahip CDS algoritması ile karşılaştırılmıştır.	Geliştirilen algoritmanın performansı en iyi performansla sahiptir.	Sezgisel 3 ile 30 arasında değişen iş ve 4 ile 25 arasında değişen iş üzerinde denemiş ve CDS den daha iyi sonuç vermiştir. $n, m > 100$ için de bu sonuçların elde edileceği düşünülmektedir. Makine sayısı, iş sayısından fazlaysa, CDS algoritmasının performansı, önerilen sezgiselden daha düşüktür. Çünkü, CDS algoritmasının performansı, makine sayısına, önerilen sezgiselin performansı iş sayısına bağlıdır.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Turner ve Booth, 1987	Deneyisel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	NEH algoritmasının performansı daha önce en iyi performansa sahip olduğu düşünülün CDS algoritması ile karşılaştırılmıştır. Ancak en iyi performansa sahip olan CDS değil Dannenbring geniş arama algoritmasıdır (Dannenbring Extensive Search-DES) (1977).	Sonuçlar, NEH algoritmasının diğer sezgisellerden hem süre hem de amaç değeri açısından çok daha iyi sonuçlar verdiğini göstermektedir. Ayrıca NEH algoritması CDS' den makine sayısının iş sayısından fazla olduğu durumlarda da daha iyi sonuçlar vermektedir.	NEH, makinenin işten fazla olduğu problemler için daha iyidir.
Hundal ve Rajgopal, 1988	Sezgisel ve deneyisel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Palmer (1965) tarafından önerilen eğitim indeksi modifiye edilerek yeni bir süreç tanımlanmıştır. Elde edilen sonuçlar, CDS sezgiseli ile karşılaştırılmıştır.	Önerilen sezgisel oldukça küçük hesaplamaya zamanı gerektirmektedir, özellikle büyük ölçekli problemler için.	Daha fazla gelişme arzulanırsa, bu yöntem minimum maliyet için iyi bir başlangıç sağlayabilir. (örn. pairwise exchange)
Cleveland ve Smith, 1989	Genetik algoritma ve deneyisel karşılaştırma	Sürekli akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Kimyasal akış tipi çizelgeleme problemin çözümü için genetik algoritma önerilmiştir. Problem 2 şekli için değerlendirilmiştir. 1. İstasyona eklenecek reaktör sayısı var. 2. İstasyona eklenecek reaktör sayısı yok. Ayrıca diğer akış kısıtları (teslim tarihi, tamir, yer değiştirme) için GA' nın sağlamlığı test edilmiştir. Ayrıca dinamik bir GA çizelgeleme yaklaşımı gösterilmiştir.	Reaktörlerin çıkış sayısı arttıkça, toplam tamamlanma zamanı azalmaktadır. En iyi çözüm reaktör sayısı için kısıt olmadığında elde edilmiştir. GA, diğer akış kısıtları dikkate alındığında, kararlı sonuçlar vermiştir.	-
Sriskandarajah ve Sethi, 1989	Sezgisel, deneyisel karşılaştırma ve teorem-ispat	n -iş, m -makine esnek akış tipi çizelgeleme problemi	Bitiş zamanı en küçüklenmesi	2 makine merkezli esnek akış tipi için önerilen sezgiseller ile en kötü durum sınırları oluşturulmuştur.	Yapılan çalışmalar, önerilen algoritmaların en kötü durumdakinden daha iyi sonuçlar verdiğini göstermiştir.	-

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Widmer ve Hertz, 1989	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	İki aşamadan oluşan bir sezgisel (SPIRIT) geliştirilmiştir. İlk aşamada, TSP yaklaşımı kullanılarak başlangıç sıralaması bulunur, ikinci aşamada, bu çözüm yasak arama ile güçlendirilir. Bununla birlikte bu algoritma literatürdeki yedi sezgiselle ve NEH ile karşılaştırılmıştır. Bu karşılaştırma sezgiselleri orijinal olarak ve iki güçlendirici süreç kullanılarak yapılmıştır.	Standart parametreler altında önerilen sezgisel problemlerin %80' ninde en iyi sonuç vermiştir.	Eklenen yöntemde tanımlanan uzaklık, iyi bir başlangıç sağlamaktadır. Ancak tüm uzaklık değerlerini test etmek güçtür. Başka bir uzaklık değeri, yasak arama yöntemi için iyi bir başlangıç olup olmayacağı sorulabilir.
Ahnadi ve Bargchi, 1990	Dal-sınır algoritması ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Tamamlanma zamanları toplamının en küçüklenmesi	Problemin çözümü için, alt sınır değerine dayanan $m-1$ yeni makine oluşturan bir süreç tanımlanmıştır. Oluşturulan alt sınır diğer bir önerilen sınır değeri ile karşılaştırılmıştır.	Önerilen alt sınır eski alt sınır değerlerine göre %75-95 avantaj sağlamaktadır.	-
Taillard, 1990	Yasak arama ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Literatürdeki mevcut klasik akış tipi çizelgeleme algoritmalarını toplam tamamlanma zamanına göre karşılaştırılmıştır ve en iyi sonucu veren NEH algoritmasının karmaşıklığı $O(km)$ den $O(n^2m)$ şekline dönüştürülmüştür. Daha sonra probleme yasak arama uygulanmıştır. Bu sezgiselin hesaplama stresini kısaltmak için iki adet paralel algoritma önerilmiş ve önerilen bu sezgiseller üzerinde deneysel hesaplamalar yapılmıştır.	Problem yasak arama kullanılarak etkin bir şekilde çözülmüştür ve NEH' den daha iyi çözümler elde edilebilir. Çözümün eniyilemesi kanıtlanamamış durumlar için yeterli çözüm zamanı verildiğinde eniyi çözüm bulunmuştur. Yasak arama tekniğinin gerektirdiği fazla hesaplama zamanını azaltmak için sunulan iki adet paralel yöntem $O(n)$ durumunu sağlamıştır.	Böyle bir teknik oldukça esnek ve henüz mevcut sezgisellerin dizayn edilmediği daha genel akış tipi problemler çok bilyık değişiklik (hazırlık süresi, işlem zamanı vb.) yapılmaksızın test edilebilir.
Brah ve Hunsucker, 1991	Dal-sınır algoritması ve deneysel karşılaştırma	n -iş, m -makine her aşamada çok işlemciye sahip akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problemin çözümü için bir dal-sınır tekniği tanımlanmıştır. Ayrıca çeşitli alt sınırlar tanımlanmış ve bir hesaplama algoritması geliştirilmiştir. Algoritma test edilmiş ve sonuçlar basit bir $n!$ mümkün çizelge geliştirici sonuçları ile karşılaştırılmıştır.	Önerilen dal-sınır algoritması amaç değerine göre eniyi çözümler sağlamaktadır.	Önerilen dal-sınır algoritması farklı amaçlara sahip çizelgeleme problemlerini çözmek için kullanılabilir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Ogbu ve Smith, 1991	Tavhlama benzetimi ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problem için çözümün tavlama benzetimi algoritması önerilmiştir.	1.Önerilen yaklaşım en iyi çözüme oldukça yakın çözümler vermektedir. Diğer yöntemlerle karşılaştırıldığında en iyi sonucu vermektedir. 2.Akış tipi için SA uygulamasında insert/shift tekniği, interchange/pairwise exchange tekniğinden daha iyidir. 3.SA diğer sezgisellerden daha fazla CPU zamanı gerektirir.	-
Rajendran ve Chaudhuri, 1991	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam akış zamanının en küçüklenmesi	3 adet sezgisel algoritma önerilmiştir. Bunlar diğer bilinen sezgisellerle karşılaştırılmıştır.	Önerilen sezgisel algoritmalar hızlı ve oldukça kesin çözümler sağlamaktadır. Ayrıca karşılaştırılan sezgisellerin de en iyisidir.	Önerilen sezgiseller, aşamalar arasındaki aylak zamanı ve işlerin bekleme süresini en küçüklemek toplam akış zamanını en küçükledikleri kadar, çizelgeleme maliyetlerini azaltmada da etkili olabilmektedir.
Cao ve Bedworth, 1992	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine seri çok ürün işleme süreci sahip aşamalararası stoklama olan akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Aşamalararası stoklama politikası olan, makineler arası taşıma zamanı ve hazırlık zamanı olmayan problemlerin çözümü için bir sezgisel yaklaşım önerilmiştir. Algoritma test edilmiş ve sonuçlar CDS algoritmasının sonuçları ile karşılaştırılmıştır.	Önerilen sezgisel algoritma küçük ve büyük ölçekli problemler için oldukça etkilidir.	-
Dubek ve diğ., 1992	Literatür incelemesi	n -iş, m -makine akış tipi çizelgeleme problemi	Mevcut literatürdeki akış tipi çizelgeleme problemlerinin ayrıntılı araştırması yapılarak bu konudaki çalışmalar sınıflandırılmıştır. Seçim kriterleri tanımlanmış, endüstrideki uygulama eksiklikleri saptanmıştır.	Mevcut literatürdeki akış tipi çizelgeleme problemlerinin ayrıntılı araştırması yapılarak bu konudaki çalışmalar sınıflandırılmıştır. Seçim kriterleri tanımlanmış, endüstrideki uygulama eksiklikleri saptanmıştır.	Akış tipi problemi sanayi de en çok kimyaya endüstrisinde ve esnek üretim sistemlerinde görülmektedir.	Gelecekte akış tipi probleminin kullanımı hakkında yorum yapılmaktadır.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Stöppler ve Bierwirth, 1992	Genetik algoritma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problem çözümü için bir paralel genetik algoritma önerilmiştir. Yaklaşım, simetrik TSP' nin kanonik kodlamasına dayanmaktadır. Kanonik kodlamanın etkisi altında genetik operatörlerin ve pairwise-exchange sezgiselinin davranışları analiz edilmiştir. Karşılaştırma için, pairwise-exchange sezgiseli, SPT sezgiseli ve alt - sınır değeri kullanılmıştır.	Önerilen sezgisel karşılaştırılan sezgisellerin de en iyisidir.	-
Ganradharan ve Rajendran, 1993	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine bekleme olmayan akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problem çözümü için iki sezgisel algoritma önerilmiştir. Bu algoritmalar ilk olarak basit yöntemlerle başlangıç çözümü oluşturmakta, daha sonra bu çözümü güçlendirmektedir. Bu algoritmaların sonuçları diğer sezgiseller tarafından üretilen sonuçlarla karşılaştırılmıştır.	Önerilen sezgisel algoritmalar hızlı ve oldukça kesin çözümler sağlamaktadır. Ayrıca karşılaştırılan sezgisellerin de en iyisidir.	Boyle bir yöntemle olan ihtiyacı, daha önce beklenmesiz akış tipi ile ilgili yapılan çalışmaların eniyiye yakın çözümler elde edememelerinden doğmuştur.
Rajendran, 1993	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam akış zamanının en küçüklenmesi	Problem çözümü için bir sezgisel algoritma önerilmiştir. Bu algoritmanın sonuçları diğer sezgiseller tarafından üretilen sonuçlarla karşılaştırılmıştır.	Önerilen sezgisel algoritmalar hızlı ve oldukça kesin çözümler sağlamaktadır. Ayrıca karşılaştırılan sezgisellerin de en iyisidir.	-
Werner, 1993	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problem çözümü için Path Algoritması önerilmiştir. Önerilen algoritmanın sonuçları diğer algoritmalarla (NEH, CDS, Dancanbiring algoritması, DES, GA, SA TS) karşılaştırılmıştır.	Önerilen algoritma, düşük hesaplama zamanı gerektirmekte ve az sayıda çözüm oluşturmaktadır. Ayrıca kısıtlar altında problemin yapısal özellikleri kullanıldığında, diğer algoritmalarla daha iyi sonuçlar vermektedir.	-

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Cartwright ve Tuson, 1994	Genetik algoritma ve deneysel karşılaştırma	n -ış, m -makine sektör çizelgeleme problemi	Erken ve gecikme maliyeti	Problemin çözümü için genetik algoritma önerilmiştir. Problemin iki temel formülasyonu gözönüne alınmıştır: 1. Tam bir sıralama 2. Release zamanlarını içeren bir problem. 4 test problemi üzerinde GA yaklaşımı diğer sıralama kuralları ile karşılaştırılmıştır. Ayrıca çeşitli operatörler tanımlanmış ve performansları karşılaştırılmıştır.	GA yaklaşımı, diğer sıralama kurallarına göre daha iyi sonuçlar vermiştir.	Sıralama yaklaşımlarının eksikliği olan daha gerçekçi amaç fonksiyonlarının kullanımının sonuçları ortaya konmuştur.
Mulkens, 1994	Genetik algoritma ve Deneysel karşılaştırma	n -ış, 2-, 3- ve m -makine akış tipi çizelgeleme problemi	Ortalama kuyrukta bekleme süresi veya toplam tamamlanma zamanı en küçülenmesi	3-makineden daha büyük problemler için genetik algoritmaya dayanan bir sezgisel geliştirilmiştir. Bu algoritmanın, n ış, 2- makine ve 3-makine akış tipi problemleri için ortalama kuyrukta bekleme süresinin, Johnson algoritmasından daha iyi olduğunu gösterebilmek testler yapılmıştır. Ayrıca önerilen sezgiselin etkinliği CDS ve NEH algoritmaları ile karşılaştırılmıştır.	Önerilen sezgisel; 1. 2-makine ve 3-makine problem için, Johnson algoritması ile aynı toplam tamamlanma zamanı vermiştir. 2. Johnson algoritmasının oluşturduğu tek çözüm yerine birkaç çözüm bulunmuştur. 3. Diğer parametreler gözönüne alındığında (ortalama kuyrukta bekleme süresinin vb.) Johnson algoritmasından daha iyi sonuçlar almak mümkündür. 4. Diğer kısıtları da (alet değiştirme, malzeme taşıma vb.) hesaba kattığı için, daha gerçekçi çizelgeleme problemleri ile uğraşma fırsatı verir. 5. CDS ve NEH algoritmaları ile karşılaştırıldığında, daha etkindir, ama benzer sonuçları vermesi daha muhtemeldir.	Bu tür algoritmaların uygulanması kolay, çalışması hızlıdır. Bu sonuçlar, gerçek atölye çalışmaları için, çizelgeleme yazılımlarına GA' nın entegre edilmesi ve kullanımını için başlangıç olarak görülebilir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çözümleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Neppalli ve diğ., 1994	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine akış tipi çözümleme problemi	Toplam ağırlıklı gecikmenin en küçüklenmesi	Problemin çözümünü için genetik algoritma yaklaşımını önerilmiştir. Algoritmanın performansı en iyi dispatching kuralı olduğu bilinen, aylak zaman kuralı olan makine darboğazlarına odaklanmış bir sezgisel ile karşılaştırılmıştır.	GA yaklaşımı en iyi sonucu vermiştir.	GA yaklaşımı ile, diğer kriterler için akış tipi problemini kolaylıkla çözülebilir.
Rajendran, 1994	Sezgisel ve Deneysel karşılaştırma	n -iş, m -makine bekleme olmayan akış tipi çözümleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problemin çözümünü için öncelik ilişkileri ve iş ekleme temeline dayanan basit bir sezgisel önerilmiştir. Önerilen algoritmanın sonuçları literatürdeki mevcut iki sezgiselle karşılaştırılmıştır.	Önerilen algoritma diğer algoritmalarından daha iyi sonuçlar vermektedir.	-
Chen ve diğ., 1995	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine akış tipi çözümleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Genetik algoritmaya dayanan bir sezgisel geliştirilmiştir. Algoritmanın performansı, SPIRIT algoritması ve HC sezgiseli ile karşılaştırılmıştır.	Önerilen sezgisel algoritma oldukça etkindir. Ayrıca karşılaştırılan sezgisellerin de en iyisidir. CPU zamanı da kısadır.	Algoritma, gerçek problemlere kolaylıkla uygulanabilir. Ayrıca diğer kriterler (toplam akış zamanı, maksimum pozitif gecikme, toplam gecikme vb.) ile de uygulanabilir. Ayrıca gelecek çalışmalarda, GA' nın karışım yaklaşımlarını ve önemini belirlemek için sistematik yaklaşımlar geliştirilecektir. Başlangıç operatörlerini, nüfus büyüklüğünü, seçim olasılığını, genetik operatörleri, genetik operatör oranlarını ve bitiş kriterini geliştirecek yaklaşımlar daha sistematik şekilde devam ettirilebilir.
Ho, 1995	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine akış tipi çözümleme problemi	Ortalama akış zamanının en küçüklenmesi	Problemin çözümünü için basit bir sezgisel önerilmiştir. Önerilen algoritmanın sonuçları literatürdeki mevcut iki sezgiselle karşılaştırılmıştır.	Önerilen algoritma diğer algoritmalarından daha iyi sonuçlar vermektedir.	-

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Islubuchi ve diğ., 1995	Tavlama benzetimi ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçükleme	Problemin çözümü için modifiye edilen bir tavlama benzetimi algoritması önerilmiştir. Daha sonra bu algoritmanın performansı diğer yerel arama yöntemleri, (katlı başlangıç inişi (multi-start descent) ve TS), SA algoritması ve rasgele örnekleme tekniği ile karşılaştırılmıştır.	Rasgele örnekleme tekniği, katlı başlangıç inişi ve TS algoritmaları ile en iyi hareket tipi algoritması ile karşılaştırıldığında: 1. İteratif geliştirme süreçlerine dayanan üç algoritmanın performansı, rasgele örnekleme tekniğine göre daha iyi sonuçlar vermektedir. 2. Katlı başlangıç inişi algoritması 10 iş probleminde en iyi iken, 20 ve 50 işli problemlerde ikinci en iyi olmaktadır. 3. Önerilen algoritma, 20 ve 50 işli problemlerde en iyi olmaktadır. 4. TS, önerilen algoritmadan sonra ikinci en iyidir. SA algoritması, katlı başlangıç inişi ve TS algoritmaları ile ilk hareket tipi algoritması ile karşılaştırıldığında: 1. Katlı başlangıç inişi algoritması 10 iş probleminde en iyi iken, 20 ve 50 işli problemlerde ikinci en iyi olmaktadır. 2. Önerilen algoritma, 20 işli problemlerde en iyi olmaktadır. 3. SA, 50 işli problemlerde en iyi olmaktadır. 4. TS, SA ve önerilen algoritmadan sonra ikinci en iyidir. 5. İlk hareket tipi algoritması, en iyi hareket tipi algoritmasından daha iyidir.	SA algoritması doğası gereği paralel uygulamalara uygun değilken, önerilen en iyi hareket tipi stratejisine sahip önerilen algoritma paralel uygulamalar için uygundur. Ayrıca iyi hareket tipi stratejisine sahip katlı başlangıç inişi ve TS algoritmaları da paralel uygulamalar için uygundur.

Tablo A.1 Tek-Amaçlı Akış Tipi Çözme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Kim, 1995	Dal-sınır algoritması ve deneysel karşılaştırma	n -iş, m -makine akış tipi çözme problemi	Toplam pozitif gecikme zamanının en küçüklenmesi	Alt problemlerin toplam pozitif gecikme zamanı üzerine alt sınırlar geliştirilmiştir. Ayrıca bir baskınlık koşulu da tanımlanmıştır. Çözüm için, bu alt-sınırları ve baskınlık koşulunu kullanan bir dal-sınır algoritması önerilmektedir. Algoritmanın performansı diğer bir algoritma ile kıyaslanmıştır.	Algoritmanın performansı, karşılaştırılan algoritmadan daha iyidir.	Problem için, daha sıkı sınırlar ve daha iyi baskınlık kuralları geliştirilebilir.
Reeves, 1995	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine akış tipi çözme problemi	Toplam tamamlama zamanı en küçüklenmesi	Genetik algoritmaya dayanan bir sezgisel geliştirilmiştir. Algoritmanın performansı, hem de Taillard'ın tasgele oluşturulmuş, basit bir konşuluk arama algoritması ve tavlama benzetimi algoritması ile karşılaştırılmıştır.	Burada özellikle problem boyutu büyüdükçe, genetik algoritmalar daha iyi performans göstermiştir. Eniyiye yakın çözüme hızlı bir şekilde ulaşmaktadır. SA'ın aksine parametre belirlemede çok miktarda denemeyi gerektirmez. GA, parametre belirlemede oldukça dayanıklıdır. Bu özellik nüfus büyüklüğünün, değişim oranının seçiminde, vb. beklenebilir.	1. Parametre uzayını eniyilemeye çalışarak kayda değer iyileşmeler sağlanabilir. (Tam GA) 2. Melez GA algoritmaları oluşturulabilir. 3. Paralel uygulamalar oluşturulabilir.
Riezebos ve diğ., 1995	Sezgisel ve deneysel karşılaştırma	n -iş, her m -aşamada tek işleme sahip akış tipi çözme problemi	Toplam tamamlama zamanı en küçüklenmesi	Öncelikle her aşamada tek işleme sahip zaman boşluğu özelliğini gözönüne alındaki problem için matematiksel formülasyon oluşturulmuştur. Daha sonra toplam tamamlama zamanı üzerine alt sınırlar geliştirilmiştir. Problemin çözümü için bir yaklaşık algoritma önerilmiş ve bilinen sezgisellerle performans sonuçları karşılaştırılmıştır.	Önerilen sezgisel algoritma oldukça etkin kesin çözümler sağlamaktadır.	Çok makine ve iş için benzer bir çalışma yapılabilir. Ayrıca özel baskınlık kriterleri tanımlanabilir. TA, SA ve GA gibi algoritmalar kullanılarak çözümler elde edilebilir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Zegordi ve diğ., 1995	Tavlama benzetimi ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problem çözümü için probleme özel sıralama bilgisini kullanan bir tavlama benzetimi algoritması önerilmiştir. Daha sonra bu algoritmanın performansı diğer klasik sezgisellerle (NEH ve CDS) karşılaştırılmıştır.	1.Önerilen algoritma, hesaplama maliyeti açısından klasik sezgisellerden daha iyi sonuç vermektedir. 2.Küçük ölçekli problemler için, önerilen algoritma diğer SA yöntemleri kadar eniyi sonuca ulaşabilmektedir. 3.Büyük ölçekli problemler için, önerilen algoritma diğer SA yöntemlerinden daha az kaliteli sonuçlar vermektedir. 4.Çözüm kalitesi ve CPU zaman açısından takas yapıldığında, önerilen algoritmanın, akış tipi problemlere uygulanabilir olduğu görülmektedir.	Bu yaklaşım, diğer çizelgeleme problemleri için de kolaylıkla uygulanabilir.
Lai, 1996	Sezgisel	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problem çözümü için bir $O(mn)$ sezgisel önerilmiştir. Önerilen algoritma için bir en kötü durum performansı analizi yapılmıştır. Ayrıca literatürdeki diğer birkaç sezgisel için de en kötü durum sınırları oluşturulmuştur.	Önerilen algoritmanın en kötü durum performans oranı $(m+1)/2$ dir.	-
Loirenço, 1996	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problem çözümü için linear programlamaya dayanan Sevas'tyanos algoritmasının farklı bir versiyonu önerilmiştir. Önerilen algoritmanın sonuçları NEH sezgiseliyle karşılaştırılmıştır.	Problem küçük boyutta veya çalışma zamanı kritik bir faktör değilse, NEH algoritmasının performansı daha iyidir. Ancak, çalışma zamanı kritik bir faktörse, önerilen algoritma özellikle küçük makine sayısı olan büyük ölçekli problemler için oldukça iyidir.	-

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Murata ve diğ., 1996	Melez genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problem için çözüm için genetik algoritma önerilmiştir. Öncelikle, çeşitli genetik operatörler (10 çaprazlama ve 4 değişim operatörü) ve parametreler (nüfus büyüklüğü, çaprazlama olasılığı ve değişim oranı) tanımlanmış ve bunların performansı değerlendirilmiştir. Daha sonra, genetik algoritma diğer arama algoritmaları (yerel arama, yasadak arama, tavlama benzetimi) ile karşılaştırılmıştır. Son olarak, iki melez genetik algoritma önerilmiş ve performansları karşılaştırılmıştır.	Genetik algoritma için en iyi operatörler iki-nokta çaprazlama ve shift change değişimidir. Simülasyon sonuçlarına göre, genetik algoritma, diğer arama tekniklerinden biraz daha iyidir ve melez algoritmalar, diğer algoritmalara (yerel arama, yasadak arama, tavlama benzetimi) göre daha güçlüdür.	-
Nowichi ve Smutnicki, 1996	Yasadak arama ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problem için çözüm için yasadak arama yaklaşımı önerilmiştir. Ayrıca önerilen yerel araştırma yaklaşımı değerlendirilmiş ve performansı en iyi olan bir algoritma ve NEH algoritması ile karşılaştırılmıştır.	Algoritmanın performansı, karşılaştırılan algoritmalarından daha iyidir. Ayrıca çözüm çok kısa sürede ulaşabilmektedir.	-

Tablo A.1 Tek-Amaçlı Akış Tipi Çözelleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Lee, 1997	Dinamik programlama, sezgisel ve teorem-ispat	n -iş, 2-makine kullanılabilirlik kısıtı altında akış tipi çözelleme problemi	Toplam tamamlama zamanının en küçükleme	Öncelikle problemin $(F2/r-a(M_j)/C_{max})$: M_j devamlılığı olan kullanılabilirlik kısıtı altında iken iki-makine akış tipi çözelleme toplam tamamlama zamanı en küçükleme problemi) NP-zor olduğu gösterilmiştir. Daha sonra $F2/r-a(M_1)/C_{max}$ problemi için dinamik programlama yaklaşımı uygulanmıştır. Ancak problem NP-zor olduğu için, çözümü için iki sezgisel önerilmiştir. İlk sezgisel, ilk makine devamlılığı olan kullanılabilirlik kısıtı altında iken iki-makine akış tipi çözelleme toplam tamamlama zamanı en küçükleme problemini çözmeyle çalışır ve 1/2 değeri için hata kısıtı analizi yapılmıştır. İkinci sezgisel, ikinci makine devamlılığı olan kullanılabilirlik kısıtı altında iken iki-makine akış tipi çözelleme toplam tamamlama zamanı en küçükleme problemini çözmeyle çalışır ve 1/3 değeri için hata kısıtı analizi yapılmıştır.	$F2/r-a(M_j)/C_{max}$ problemi, NP-zor dur. Her iki sezgiselde $O(n \log n)$ dir.	Problem tersinir değildir. Yani, makine 1 ve 2 de işlem süreleri değişirse, eşdeğer bir problem oluşmaz ki bu standart akış tipi problemlerden farklıdır.
Wang ve diğ., 1997	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine akış tipi çözelleme problemi ($n/m/F2/C_1$)	Tamamlama zamanları toplamının en küçükleme	Problemin çözümü için iki sezgisel algoritma önerilmiştir. İlk algoritma, makine aylak kalma zamanını azaltması üzerine odaklanmış, ikinci algoritma, hem makine aylak kalma zamanını hem de işin kuyrukta bekleme zamanını azaltmaya çalışmaktadır. Algoritmalar küçük ve büyük ölçekli problemler üzerinde test edilmiş ve en iyi çözümle karşılaştırılmıştır. Ayrıca bir alt sınır değeri ile karşılaştırılmıştır.	Önerilen algoritmalar, etkin şekilde eniyiye yakın sonuçlar vermektedir. Ayrıca problem büyüklüğüne karşı oldukça sağlamdır.	NP-tam çözelleme problemleri için problemin yapısını araştırarak sınırlandırılmış polinomsal algoritmalar bulunabilir. Böyle algoritmalar, daha iyi çözümler oluşturabilir ve gerçek problemlere uygulanabilir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Yamada ve Reeves, 1997	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problemin çözümü için, bir kritik bölge dayanan komşuluk yapısı ve bir genetik yerel arama yöntemi kullanılarak big valley yapısını da kullanılır kılan bir yaklaşım önerilmiştir.	Deneme sonuçları önerilen yöntemin etkinliğini göstermektedir.	-
Cotta ve Troya, 1998	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Akış tipi probleminin farklı gösterimlerinin kalitesini değerlendirmek için form analizi kullanılmıştır. Öncelikle genetik algoritmalarındaki klasik şema işlemi ele alınmış ve daha genel varlıklar (formae) için yeniden formüle edilmiştir. Genel varlıkların özellikleri ve nasıl işleneceği açıklanmıştır. Ardından permutasyon akış tipinin tanımı ve forma tabanlı gösterimi ele alınmıştır. Daha sonra, klasik birleşim operatörleri analiz edilmiş, her bir operatör için forma yapısı tanımlanmış ve en iyi operatör belirlenmiştir. Daha sonra, dört yeni operatör tanımlanarak etkinlikleri test edilmiş ve en iyi seçilen bu operatörle karşılaştırılmıştır.	Bir çizelgeleme problemi verildiğinde, ölçüm değeri için, en çarpıcı özellik, işlemlerin sırası veya öncelik ilişkilerinden çok pozisyonlarına bağlıdır. Bundan başka, sürekli pozisyonları arasında güçlü bir ilişki vardır.	Çalışmada toplam tamamlanma zamanı dikkate alınmıştır. Bu sonuçlar genelleştirilebilir. Çalışma, diğer yeni tanımlanmış operatörler ile de yapılabilir. Melez operatörler tanımlanmakta mümkündür. Ancak baskın gösterimi belirlemek zor olduğu için, analiz güç olabilir. Bu baskın gösterim, araştırma ilerledikçe değişebilir. Bunun için daha derin bir araştırma gerekebilir.
Janiak ve Portmann, 1998	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine lineer modelli işlem süresine sahip akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Genetik algoritmaya dayanan bir sezgisel geliştirilmiştir. Bu algoritma için başlangıç çözümü CDS, Dannenbring ve NEH algoritmalarından elde edilmiştir. Daha sonraki aşamalarda (survival, çaprazlama, değişim) problemin tanımlanan özelliklerine uygun klasik, stokastik veya spesifik teknikler kullanılmıştır. Bu şekilde tüm kombinasyonlardan 4 adet GA oluşturulmuştur. Algoritmaların performansları birbirleri ile karşılaştırılmıştır.	GA3 (özel çaprazlama ve klasik değişim) ortalamada en iyi çözümü verir.	GA'lar, makalede tanımlanan yeni sezgisellerin başlangıç aşamasında oluşturduğu en iyi çözümleri güçlendirmektedirler. O halde oldukça güçlüdür.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Koulamas, 1998	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine permütasyon olmayan akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Önerilen algoritmanın performansı en iyi performansa sahip olan NEH algoritması ile karşılaştırılmıştır.	Öncelikle, her iki algoritma permütasyon iş akışı durumunda eniyiye çok yakın sonuçlar vermiştir. Permütasyon olmayan iş akışı durumunda ise, önerilen algoritma, NEH algoritmasından daha iyidir.	Önerilen algoritma, permütasyon olmayan iş akışı durumunda en iyi hatıra eniyi olabilen bir sezgisel algoritmadır.
Portmann, ve diğ., 1998	Dal-sınır, genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine melez (atama+çizelgeleme) akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Büyük ölçütlü problemler için kullanılamaz olan bir dal-sınır algoritması, bu tür problemlerin çözümünde de kullanılabilemesi için güçlendirilmiştir. Bu amaçla, öncelikle bu algoritmanın alt sınır değeri güçlendirilmiştir. Başlangıç değerlerini üreten sezgisellerin tanımlanmasından sonra üst sınırı geliştirmek için GA yaklaşımı önerilmiştir. Daha sonra dal-sınır yaklaşımı sonuçları ile dal-sınır+GA yaklaşımı sonuçları test problemleri üzerinde test edilmiştir.	GA, hem ele alınan kriterin değerini hem de eniyi değeri iyileştirmiştir.	Bir çok dal-sınır yaklaşımında olduğu gibi alt sınır değerinin sıklığı ile ilgili daha fazla gelişmeler bulunabilir.
Reeves ve Yamada, 1998	Genetik algoritma, path relinking ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Reeves (1995), daha önceki çalışmasında akış tipi çizelgeleme için genetik algoritma yaklaşımının konşuluk arama tekniğinden ve tavlama benzetimi algoritmasından daha iyi sonuçlar verdiğini vurgulamıştır. Ancak, Nowicki ve Smutnicki (1996), yasak arama metodunun en iyi olduğunu ifade etmişlerdir. Bu çalışmada, GA algoritması yeniden ele alınmış ve operatörler kullanılarak geliştirilen alan özelliklerinin ele alınışı gösterilmiştir. Bunun için Thailand'ın algoritması kullanılarak çeşitli operatörlerle gelişme sağlanmıştır. Sonuçlar, Nowicki ve Smutnicki yasak arama algoritması, eniyi çözüm ve alt sınır değerleri ile karşılaştırılmıştır.	Önerilen algoritma, önemli şekilde iyileştirmeler sağlamış ve karşılaştırılan yöntemlere göre daha iyi sonuçlar vermiştir.	Çalışma da bulunan çözümlerin kalitesini azaltmaksızın, hesaplama zamanını azaltmanın mümkün olup olmadığını araştırılacaktır. Farklı tip akış tipi problemleri için (toplam iş tamamlama zamanının en küçüklenmesi, hazırlık zamanına bağlı sıralamalar oluşturma) çalışmalar geliştirilebilir, path relinking uygulamasında alternatif yollar geliştirilebilir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Riezebos ve Gaalman, 1998	Sezgisel ve deneysel karşılaştırma	n -iş, her m -aşamada çok işlemciye sahip akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Öncelikle her aşamada çok işlemciye sahip zaman boşluğunu gözönüne alındığı problemin çözümü için bir alt sınıra dayalı bir sezgisel algoritmalar önerilmiştir. Bu sezgisellerin performansı farklı zaman boşluğu büyüklüklerinde test edilmiştir ve büyüklüğün etkisi gösterilmiştir. Ayrıca literatürdeki diğer sezgisellerle karşılaştırma yapılmıştır.	Bir alt sınıra dayalı bir sezgisel algoritmaların en iyi performansla sahip olduğu görülmektedir. Ayrıca sezgisellerin performansı üzerinde zaman boşluğu büyüklüğünün etkisi olduğu görülmektedir. Zaman boşluğu büyüklüğü en iyi sonucu SPT vermektedir.	TA, SA ve GA gibi algoritmalar kullanılarak çözümler elde edilebilir.
Stützle, 1998a	Karınca kolonileri eniyileme ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Çözüm algoritması olarak yerel aramanın yapıldığı max-min karınca sistemi kullanılmıştır. Algoritmanın performansı, diğer mevcut sezgiseller (NEH, NEH+yerel arama, tavlama benzetimi, katlı iniş) ile karşılaştırılmıştır.	Sonuçlar, yaklaşımın karşılaştırılan diğer mevcut sezgisellerden daha iyi olduğunu göstermektedir.	-
Stützle, 1998b	Ardaşık yerel arama ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Çözüm algoritması olarak ardaşık yerel arama kullanılmıştır. Algoritmanın performansı, diğer mevcut sezgiseller (NEH, NEH+yerel arama, tavlama benzetimi, katlı iniş) ile karşılaştırılmıştır.	Sonuçlar, yaklaşımın karşılaştırılan diğer mevcut sezgisellerden daha iyi olduğunu göstermektedir.	-
Woo ve Yim, 1998	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Ortalama toplam akış zamanı en küçüklenmesi	Problemin çözümü için, iş ekleme kuralına dayanan bir sezgisel geliştirilmiştir. Önerilen algoritma, o anki kısmi sıralamaya ekleyeceği işi, o anki kısmi sıralamaya eklenmemiş işler arasından seçer. Ne zaman yeni bir kısmi sıralama oluşturulduğunda, algoritma tüm mümkün sıralamaları, her bir zamanda iş ekleyerek değerlendirir. Ayrıca önerilen sezgiselin etkinliği CDS, NEH ve Rajendran algoritmaları ile karşılaştırılmıştır.	Önerilen algoritmanın performansı, diğer algoritmalarından daha iyi sonuçlar vermiştir. Ancak CPU zamanı da oldukça uzundur. İş makine oranı 2' den büyük olduğunda, önerilen algoritma tam sonuçlar elde etmektedir.	Mevcut iş ekleme kuralı, bir iş ekleme sırası belirlerken, önerilen algoritma, o anki kısmi sıralamaya eklenilecek tüm kalan işleri değerlendirir. Bu nedenle CPU zamanının daha büyük olması doğaldır.

Tablo A.1 Tek-Amaçlı Akış Tipi Çözelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Yamada ve Reeves, 1998	Genetik algoritma, yerel arama ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi ($n/m/F/C_{\max}$)	Toplam bitiş zamanının en küçüklenmesi	Probleminin çözümü için yeni bir metasezgisel (genetik yerel arama) yaklaşım önerilmiştir.	Deneme sonuçları önerilen yöntemin etkinliğini göstermektedir.	-
Onwubolu ve Mutingi, 1999	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	1. Toplam pozitif gecikme zamanının en küçüklenmesi 2. Geciken işlerin sayısının en küçüklenmesi 3. Her iki amacın arda arda en küçüklenmesi	Problemin çözümü için iki yaklaşım önerilmiştir. İlk yaklaşım, klasik sezgisellere dayanmakta, ikinci yaklaşım ise bir genetik algoritmadır.	İlk yaklaşım küçük boyutlu problemler için uygunken, ikinci yaklaşım, kısa sürede orta ve büyük ölçekli problemlerin çözümü için uygundur.	-
Cheng ve Wang, 2000	Sezgisel ve teorem-ispat	n -iş, 2-makine kullanılabilirlik kısıtı altında akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Daha önce $F2(t-\alpha(M)/C_{\max})$ için sezgisel bir yaklaşım önerilmiştir. Öncelikle, bu algoritmanın hata değeri olan $1/2$ 'nin zayıf olduğu gösterilmiştir. Daha sonra, hata değerini $1/3$ olan bir sezgisel önerilmiştir.	Daha önce önerilen algoritmanın hata değeri olan $1/2$ zayıftır. Önerilen sezgiselin hata değeri $1/3$ tür.	Önerilen sezgisel $O(n \log n)$ dir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Kumar ve diğ., 2000	Sezgisel, genetik algoritma ve deneysel karşılaştırma	n -ış, m -makine bekleme olmayan yığın akışlı akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçükleme	Problem çözümü için sezgisel yöntemler ele alınmıştır. Tek ürün durumunda, eniyi sürekli alt-yığın sayısını bulmak için, linear programlama kullanılmıştır. Tek ürün durumunda, eniyi tamsayı alt-yığın sayısını bulmak için, literatürdeki mevcut bir heuristic kullanılmıştır. Çok ürün durumunda, eniyi sürekli alt-yığın sayısını bulmak için, TSP formülasyonu kullanılmıştır. Çok ürün, eniyi tamsayı alt-yığın sayısını bulmak için, farklı bir sezgisel geliştirilmiştir. Daha sonra, çok ürün yığın akışı ve çizelgeleme probleminin çözümü için genetik algoritmaların kullanımı değerlendirilmiştir. Problemin üç karar değişkenini (alt-yığın sayısı, alt-yığın büyüklüğü ve sıralama) bulmak için bir metasezgisel kullanılmış ve performansı tam GA kullanımı ile karşılaştırılmıştır.	Melez GA, daha etkin sonuçlar çıkarmıştır.	Boyle spesifik problemlerin çözümü için geliştirilmiş sezgisellerin avantajlarını kullanan bir GA ile yığın sayısı da eniyilenebilir. Böyle bir metot henüz mevcut değildir.
Lee ve Shaw, 2000	Sinirsel ağırlar ve deneysel karşılaştırma	n -ış, m -makine gerçek zamanlı akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçükleme	Problem çözümü için sinirsel ağ yaklaşımı ele alınmıştır. Gerçek zamanlı bilgilerin sinirsel ağ yaklaşımı ile öğrenilmesi ve bu bilginin çizelgeleme için kullanılması hedeflenmiştir. Öncelikle öğrenme etkisi test edilmiştir. Ardından, çeşitli geliştirme sezgiselleri (GA, SA, pairwise exchange) karşılaştırılmıştır. Daha sonra, önerilen yaklaşım, çeşitli sıralama sezgiselleri (GA ve NEH) ile karşılaştırılmıştır. Son olarak, sinirsel ağ ile melez algoritmanın performansı tam GA ile karşılaştırılmıştır.	1. Sinirsel ağ öğrenme etkisi çözüm kalitesini artırmaktadır. 2. Geliştirme sezgiselleri içinde en iyi sonucu GA vermektedir. 3. Önerilen yaklaşımın sonuçları, diğer sezgisellerden daha iyi ancak GA' dan sadece 3.4% farklıdır. 4. Sinirsel ağ +GA melez algoritmanın performansı, tam GA' dan daha iyidir.	Sinirsel ağ, uygulaması kolaydır, uygun çözümler sunar ve çözüm kalitesi ve hesaplama zamanı iyidir. Farklı akış tipleri, sıralama kararları üzerinde farklı zaman kısıtlarına sahip olduğunda, sinirsel ağ, geliştirici bir sezgisel veya melez bir algoritma olarak etkin şekilde uygulanabilir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Aggoune, 2001	Genetik algoritma, yasadışı arama ve deneysel karşılaştırma	n -iş, m -makine kullanılabilirlik kısıtı altında akış tipi çizelgeleme problemi	Toplam tamamlama zamanı en küçüklenmesi	Makinelerin öyleyici bakımı için iki model gözönüne alınmıştır: 1. Kullanılmayan periyotların yeri sabittir(FM). Amaç, bu periyotların yerine göre tüm işlerin iş kesintisi olmayan çizelgesini belirlemek. 2. İş kesintisi kısıtı, her bakım işinden önce aylak zamanların en küçüklemek için, bakım periyotlarını sabit değil çizelgenin yapısına ve belli bir planlama döneminde olacak şekilde sabitlememektir(DM). Bu amaçla bir çizelgeleme süreci tanımlanmıştır. Bu süreç, belli sıralama girdisine göre makine kullanılabilirlik ve öncelik kısıtlarına göre işlerin başlama zamanını belirlemektedir. Bu amaçla iki çözüm yaklaşımı kullanılır: GA ve TS. Her iki algoritmanın çözüm sonuçları birbirleri ile karşılaştırılmıştır.	TS, GA' ya göre biraz daha karardır. TS-DM, TS-FM' ye göre önemli bir güçlendirme sağlar. TS daha iyi sonuç verdiği biyık ölçekli problemler dışında, hangi eniyileme yönteminin baskın olduğu belirlenmemiştir. Daha fazla parametre değiştirmek ve daha fazla denemeler yapmak gerekir. İki metodun birleşimi daha iyi sonuç verebilir.	Üretim çizelgelerini kapsayan bakım süreçleri için çalışma genişletilebilir.
Caraffa ve diğ., 2001	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine bloklanan akış tipi çizelgeleme problemi	Toplam tamamlama zamanı en küçüklenmesi	Tampon bölge kullanılmaksızın, bloklanan akış tipi probleminin çözümü için, büyük ölçütlü problemleri çözecek bir genetik algoritma yaklaşımı geliştirilmiştir. Bu yaklaşım, "slowing down" kavramına dayanmaktadır. Yaklaşımın performansı benzer bir problem için geliştirilen bir sezgisel ile karşılaştırılmıştır. Ayrıca algoritmanın en iyi çalışma parametrelerinin neler olduğu tespit edilmiştir. Bunun için ANOVA analizi yapılmıştır.	Önerilen yaklaşım, Abadi tarafından geliştirilen sezgisel göre daha kısa zamanda daha iyi sonuçlar vermektedir. Ayrıca algoritmanın çalışmasını etkileyen parametreler, nüfus büyüklüğü ve değişim operatör parametreleridir.	Bloklanan akış tipi durumuna benzemeyen koşullar için hiçbir karşılaştırma literatürde bulunmamaktadır. GA' nın performansını değerlendirmek için, en azında iyi bir alt sınır sürecine ihtiyaç vardır.

Tablo A.1 Tek-Amaçlı Akış Tipi Çözümleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Chan ve Hu, 2001	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine akış tipi çözümleme problemi	1. Toplam tamamlanma zamanı en küçüklenmesi 2. Gecikme cezalarının en küçüklenmesi (Ayrı ayrı ve aynı anda)	Doküman fabrikasında akış tipi sıralama probleminin çözümü için genetik algoritma yaklaşımı önerilmiştir. Ayrıca çok amaçlı problemler için yine GA tabanlı bir yaklaşım önerilmiştir. Algoritmanın performansları, klasik sezgisellerle (Palmer, Gupta, CDS ve hızlı giriş) ve EDD kuralı ile karşılaştırılmıştır.	GA ise, akış tipi problemin çözümü için oldukça etkindir. Önerilen algoritma, kısa sürede en iyi ve/veya eniyiye yakın çözümler verebilme yeteneğindedir. Ayrıca, çok amaçlı problem şartlarında, algoritma seçim için karar vericiye birçok kaliteli uzlaşık çözümler sunabilir.	Farklı girdi değerlerine sahip diğer problemler üzerinde deneyimlerin üretimi çözümleme probleminin çözümünde GA'nın etkinliğini tam olarak belirlemek için yapılabilir.
Oğuz ve diğ., 2001	Yasak arama, tavlama benzetimi ve deneysel karşılaştırma	n -iş, her m -aşamada çok işlemciye sahip melez akış tipi çözümleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Probleminin çözümü için TS ve SA'ya dayanan iki metasezgisel yaklaşım önerilmiştir. Algoritmanın performansı en iyi dispatching kuralı olduğu bilinen, aylak zaman kuralı olan makine darboğazlarına odaklanmış diğer bir sezgisel ile karşılaştırılmıştır.	Orta ve büyük ölçekli problemler için, önerilen metasezgiseller TS'nin daha fazla iyi olduğu sonuçları vermiştir. Çok büyük ölçekli problemler için, TS'nin çalışma süresi kabul edilemez olduğu için, SA daha avantajlıdır.	Önerilen metasezgiseller, işlem zamanı, başlama ve teslim tarihleri gibi iş parametrelerine bağlı olarak, aşama sayısına ve her aşamadaki mevcut işlemci sayısına bağlıdır.
Yong ve Sannomiya, 2001	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine akış tipi çözümleme problemi	Toplam pozitif gecikme zamanının en küçüklenmesi	Büyük ölçekli problemler için çözümünde, aranan bölgenin azaltılmasına yönelik genetik algoritma tabanlı bir yaklaşım sunulmuştur. Öncelikle, aranan bölgenin azaltılma öncelik kuralları belirlenmiş, ardından standart GA kullanılarak probleme çözüm sağlanmıştır. Daha sonra, spesifik arama bölgesinde genetik arama için öncelik kısıtlama hizmet eden güçlendirici bir çaprazlama operatörü önerilmiştir. Algoritma (sabit arama bölgesi GA ve standart GA) ile karşılaştırılmıştır.	Önerilen algoritma daha etkin ve sağlam sonuçlar vermiştir. Önerilen azaltılmış arama bölgesinde genetik arama yapan genetik operatör çözüm kalitesini arttırmaya yardımcı olmuştur.	Azaltılma mekanizması çözüm kalitesini ve hesaplama hızını iyileştirir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Chan ve Hu, 2002	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	1. Toplam tamamlanma zamanı en küçüklenmesi 2. Gecikme cezalarının en küçüklenmesi (Aynı ayrı ve aynı anda)	Döküm fabrikasında akış tipi sıralama probleminin çözümü için genetik algoritma yaklaşımı önerilmiştir. Algoritmanın performansları, klasik sezgisellerle (Palmer, Gupta, CDS ve hızlı giriş) ve EDD kuralı ile karşılaştırılmıştır.	Klasik sezgiseller, toplam tamamlanma zamanını azaltmakta iken, EDD kuralı toplam gecikmenin ve/veya toplam erken cezalarının azaltılmasında etkilidir. GA ise gecikme veya toplam tamamlanma zamanı için oldukça etkilidir. Önerilen algoritma, kısa sürede eniyi ve/veya eniyeye yakın çözümler verilebileceği yeteneğindedir. Ayrıca, değişken işlem koşulları altında iyi sonuçlar üretebilir.	Önerilen algoritmanın büyük ölçekli problem verileri ile test edilmesi gerekir. Diğer yandan çalışanların sabit bir işi yaptıkları farzedilmiştir. Bu gerçek endüstri uygulamalarına uygun hale getirilebilir. Ayrıca çizelgeleme ve kaynak tahsisinin eşzamanlı olduğu durumlar için de çalışma yapılabilir.
Leu ve Hwang, 2002	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine karışık üretim paketi olan, farklı teslim zamanlarına sahip ve kaynak kısıtını gözönüne alan akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Probleminin çözümü için genetik algoritma tabanlı bir yaklaşım önerilmiştir. Algoritma, başlangıç çözümü için NEH sezgiselini kullanmaktadır. Önerilen model, Reeves' in modeli ile karşılaştırılmıştır. Öncelikle, kaynak kısıtı ve üretim karışımının etkisi analiz edilmiş, daha sonra, döküm üretiminin faktörleri ve etkileri incelenmiştir.	Tam üretim karışımı durumunda, toplam tamamlanma zamanı kısalmakta, kısmi üretim karışımı durumunda ise, toplam tamamlanma zamanı uzamaktadır. Sezgisel sadece bir uygun çözüm önerirken, genetik algoritma tabanlı yaklaşım, birkaç uygun veya yarı-eniyi çözüm önerir. Döküm üretiminin faktörleri şunlardır: işgücü, vinç, enerji sağlama kapasitesi ve destek kafes stok alanı. GA tabanlı model, akış tipi problemleri için, hesaplama süresine göre oldukça etkilidir. Kaynak kısıtı ve üretim karışımı dikkate alındığında oldukça esnek olduğu görülmektedir.	Çalışma sadece tek işlemli problemlere odaklanmıştır. Çoklu üretim hatları gibi alanlarda da benzer çalışmalar yapılabilir. Nüfus boyutu ve nesil sayısı artıkça, eniyi çözüme ulaşma şansı artmaktadır. Ancak, hesaplama süresi de o kadar artmaktadır. GA araştırması kullanıldığında, daha güçlü hesaplama ortamları (paralel hesaplama gibi) gerekmektedir. GA tabanlı araştırmaları gerçek problemlere kolaylıkla uygulamak için, GA parametreleri için fonksiyonel arayüzler ve açıklayıcı kılavuzlar büyük yarar sağlayacaktır.
Nagano ve Moccellini, 2002	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problem özelliklerindeki gelişmeleri izleyerek, bu çizelgeleme probleminin özellikleri sunulmuş ve çözümü için yeni bir geliştirici sezgisel önerilmiştir. Sezgiselin performansı en iyi olduğu bilinen NEH algoritması ile karşılaştırılmıştır.	Önerilen sezgiselin performansı, NEH algoritmasına çok yakındır.	NEH algoritması, önerilen algoritmadan daha iyi değildir. Hatta bazı yrdzeleri dikkate alındığında algoritmanın NEH sezgiseli tarafından elde edilen sonuçlara göre daha kısa toplam tamamlanma zamanına sahip olması beklenir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Tang ve Liu, 2002	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine akış tipi çizelgeleme problemi	Ortalama toplam akış zamanının en küçüklenmesi	Modifiye edilmiş genetik algoritma geliştirilmiştir. Standart genetik algoritmayı güçlendirmek için iki operatör eklenmiştir. Algoritmanın performansı, HC, Rajendran sezgiselleri ve standart genetik algoritma ile karşılaştırılmıştır.	1.Önerilen algoritmanın performansı, GA' dan daha güçlüdür. 2.Çözüm kalitesi, problem yapısı ve boyutu açısından GA kadar etkilenebilmektedir. Yani daha sağlamdır. 3.Önerilen algoritma, diğer sezgisellerden de daha iyidir. 4.Hesaplama zamanı tüm algoritmalarda, problem boyutu arttıkça artmaktadır. Önerilen algoritma ve GA, iş sayısından, makine sayısına göre daha fazla etkilenebilmektedir. 5.Algoritmanın hesaplama zamanı, GA ile benzerdir ancak diğer sezgisellerden daha uzundur.	Önerilen algoritmanın yapısı diğer eniyileme problemlerine uygulanabilecek kadar geneldir. Benzer bir süreç, tek makine çizelgeleme problemine uygulanmıştır. Gelecekteki araştırma konusu, GA yapısı içerisinde daha hızlı sezgisel çözümler nasıl elde edileceği ile ilgilidir.
Yoon ve Ventura, 2002	Melez genetik algoritma ve deneysel karşılaştırma	n -iş, m -makineyi yığın akışlı çizelgeleme problemi	Tamamlanma zamanının teslim zamanından ortalama ağırlıklı sapmasının en küçüklenmesi	Yığın akışlı akış tipi çizelgeleme problemi için, doğrusal programlama ve pairwise interchange yöntemlerini içine alan melez bir genetik algoritma yaklaşımı önerilmiştir. Algoritmanın performansı, P1 yöntemi kullanılmaksızın uygulanan genetik algoritma sonuçları ile karşılaştırılmıştır.	Önerilen algoritma, GA' ya göre daha iyidir. Ayrıca, bu tip problemlerin çözümünde, problem boyutu arttıkça GA' ya göre daha iyi sonuçlar vermektedir.	Bu çalışmada, önerilen yaklaşım, ardışık makineler arasındaki tamponların sonsuz kapasiteye sahip olduğu ve alt-yığınların eşit sayıda olduğu yığın akışlı akış tipi çizelgeleme problemi için geliştirilmiştir. Doğrusal programlama ve pairwise interchange yöntemlerini içine alan yaklaşım, ardışık makineler arasındaki tamponların sınırlı kapasiteye sahip olduğu ve alt yığınların sürekli olduğu diğer yığın akışlı akış tipi çizelgeleme problemlerine de uygulanabilir. Ayrıca, beklenmesiz yığın akışlı akış tipi çizelgeleme problemi de bu yaklaşımla çözümlenir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Zhixin ve diğ., 2002	Sezgisel ve deneysel karşılaştırma	n -iş, 2-aşamada tek işlemciye sahip bekleme olmayan melez akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problemnin çözümü için bir hırsli sezgisel önerilmiştir. Ayrıca en kötü durum analizi yapılmıştır. Hesaplama sonuçları, diğer sezgisellerle karşılaştırılmıştır.	Önerilen algoritma, diğer algoritmalarından daha üstün sonuçlar vermektedir. Ayrıca, hesaplama stresi oldukça düşük ve kolay uygulanabilir.	Benzer çalışma her aşamadaki paralel makine için yapılabilir.
Aldowaisan ve Allahverdi, 2003	Sezgisel, genetik algoritma, tavlama benzetimi ve deneysel karşılaştırma	n -iş, m -makine ve bekleme olmayan akış tipi çizelgeleme problemi	Toplam tamamlanma zamanı en küçüklenmesi	Problemnin çözümü için tavlama benzetimi ve genetik algoritmaya dayanan iki sezgisel önerilmiştir. Ayrıca, bu sezgiseller, NEH ve önerilen ekleme süreci ile birleştirilmesi ile daha iyi sezgiseller (SA1 ve GEN1) elde edilmiştir. Daha sonra, SA1 ve GEN1 sezgiselleri, pairwise interchange süreci uygulanarak daha güçlü hale getirilmiştir. Bu sezgiseller (SA, GEN, SA1, GEN1, SA2 ve GEN2), bu problem için en iyi iki sezgiselle (Gangadharan ve Rajendran (GAN-RAJ) ve Rajendran (RAJ)) karşılaştırılmıştır.	1.RAJ'ın performansı, GAN-RAJ'den daha iyidir. 2.SA, GAN-RAJ,RAJ ve GEN'den daha iyidir. 3.SA1 ve GEN1, SA ve GEN'den daha iyidir. 4.SA2 ve GEN2, SA1 ve GEN2'den daha iyidir. 5.SA1 ve GEN1, SA2 ve GEN2 hemen hemen aynı performansla sahiptir.	SA2 ve GEN2'nin kullanımı tavsiye edilebilir. Bunun yanında, bekleme olmayan farklı amaç fonksiyonuna sahip akış tipi çizelgeleme problemlerine ve diğer çizelgeleme problemlerine bu güçlendirici tekniklerin kullanılması mümkündür.
Fink ve Voß, 2003	Tavlama benzetimi, yasad arama ve deneysel karşılaştırma	n -iş, m -makine sürekli akış tipi çizelgeleme problemi	Toplam akış zamanı en küçüklenmesi	Problemnin çözümü için birkaç metasezgisel (SA, TS) yaklaşım önerilmiştir. Bunların birbirlerine göre çalışma zamanı ve çözüm kalitesi incelenmiş, ayrıca, algoritmalar uygulanma ve uyarlanmasında gerekli bilgi ve efor açısından değerlendirilmiştir.	Hesaplama sonuçları, tüm metodların tek başına baskın olmadığını göstermektedir. Shift hareketi ile belirlenen komşuluk yapısının en iyi olduğu görülmektedir.	Benzer çalışma farklı akış tipleri içinde yapılabilir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çözelleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Wang ve Zheng, 2003	Melez genetik algoritma ve Deneyisel karşılaştırma	n -iş, m -makine akış tipi çözellemesi problemi	Toplam tamamlanma zamanı en küçükleme	Probleminin çözümü için genetik algoritma yaklaşımı kullanılmıştır. Öncelikle, GA'nun performansının farklı şekillerde etkileyen başlama (NEH, SGA, NEH+SGA), çaprazlama (LOX, PMX, C1, NABEL) ve mutasyon (SWAP, INV, INS) operatörleri değerlendirilmiştir. Daha sonra yeni bir melez sezgisel önerilmiştir. Bu sezgiselin performansı, NEH ve SGA sezgiselleri ile karşılaştırılmıştır.	Probleminin çözümü için önerilen melez sezgisel en iyi çözümü vermektedir.	Daha kaliteli, zaman performansı yüksek ve sağlamlık elde etmek için, diğer üst düzey operatörler, probleme özel bilgi uygulanması genellik ve kolaylık gösterdiği için, diğer birçok probleme de uygulanabilir.
Framinan ve diğ., 2004	Sezgisel algoritma ve Deneyisel karşılaştırma	n -iş, m -makine akış tipi çözellemesi problemi	Ortalama toplam akış zamanının en küçükleme	Problemin çözümü için literatürde mevcut sezgisellerin sınıflandırması ve geniş bir karşılaştırması yapılmıştır. Ayrıca, elde edilen sonuçlara göre, problem için, iki yeni birleşik sezgisel (C1 ve C2) önerilmektedir.	Hesaplama sonuçları, FL (Framinan ve Leisten) ve B5FT (Framinan ve ark.) sezgisellerinin en iyi çözümler verdiğini göstermektedir. FL, çözüm kalitesi açısından, B5FT ise, çözüm süresi açısından en iyi olduğu görülmektedir. C2 sezgiseli tüm mevcut sezgisellerden daha iyi sonuçlar vermektedir.	-
Iyer ve Saxena, 2004	Genetik algoritma ve Deneyisel karşılaştırma	n -iş, m -makine akış tipi çözellemesi problemi	Toplam tamamlanma zamanı en küçükleme	Probleminin çözümü için genetik algoritma yaklaşımı kullanılmıştır. Amaç, bir akış tipi çözellemesi probleminin çözümünde GA'nın etkinliğinin problem yapısına uygun çeşitli GA operatörlerini biçimlendirerek güçlendirilebileceğini göstermektir. Bu amaçla standart GA ile probleme özel bilgi kullanan değiştirilmiş bir arama stratejisi karşılaştırılmıştır.	Problem yapısal bilgisi kullanılarak standart GA'nın yeniden düzenlenmesine dayanan önerilen algoritma, standart GA'ya göre daha iyi sonuçlar vermektedir.	Benzer çalışma, diğer çözellemesi problemleri için de yapılabilir.
Oğuz ve diğ., 2004	Yasak arama ve Deneyisel karşılaştırma	n -iş, her m -aşamada çok işlemciye sahip melez akış tipi çözellemesi problemi	Toplam tamamlanma zamanı en küçükleme	Öncelikle problemin karmaşıklığını ortaya konmuştur. Probleminin çözümü için TS algoritmasına dayanan bir sezgisel yaklaşım önerilmiştir.	Önerilen algoritma, en iyi çözümleri basit yasaklar listeleriyle oluşturmaktadır.	Problem için, hazırlık zamanı, transfer zamanı, farklı eniyileme kriterleri dikkate alınarak değerlendirilebilir.

Tablo A.1 Tek-Amaçlı Akış Tipi Çözme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Rajendran ve Ziegler, 2004	Karınca kolonileri eniyileme ve Deneysel karşılaştırma	n -iş, m -makine akış tipi çözme problemi	1. Toplam tamamlanma zamanı en küçükleme veya 2. Toplam akış zamanının en küçükleme	Problemi çözmek için iki karınca-koloni algoritması önerilmiştir. İlk karınca-koloni algoritması (M-MMAS), Stützle tarafından önerilen max.min karınca algoritması ile Merkle ve Middendorf (2000) tarafından geliştirilen toplama kuralı ve önerilen yeni bir yerel arama tekniği birleştirilerek elde edilmiştir. İkinci algoritma (PACO) ise, yeni oluşturulmuştur. Önerilen algoritmalar, test edilmiştir. Daha sonra, önerilen algoritmalar toplam akış zamanı gözönüne alınarak, Liu ve Reeves tarafından belirlenen en iyi sezgisel sonuçları ile karşılaştırılmıştır.	Karşılaştırmalar, önerilen karınca-koloni algoritmalarının en iyi sonucu verdiğini göstermiştir.	Farklı amaçlara göre akış tipi, atolye tipi ve hücre tipi gibi diğer çözme problemlerini çözmek için de daha etkin karınca koloni algoritmaları geliştirilebilir.
Thornton ve Hunsucker, 2004	Sezgisel algoritma ve Deneysel karşılaştırma	n -iş, her m -aşamada çok işlemciye sahip akış tipi çözme problemi	Toplam tamamlanma zamanı en küçükleme	Öncelikle her aşamada çok işlemciye sahip ve aşamalar arasında stoklama ve blokajla bulunmayan problemin çözümü için bir sezgisel algoritma önerilmiş ve bilinen sezgisellerle performans sonuçları karşılaştırılmıştır.	Önerilen sezgisel karşılaştırılan sezgisellerin de en iyisidir.	Önerilen güçlü bir algoritma, uygun problem özellikleri ile daha büyük ölçekli problemlerin çözümü için uygulanabilir.
Ying ve Liao, 2004	Karınca kolonileri eniyileme ve Deneysel karşılaştırma	n -iş, m -makine akış tipi çözme problemi ($n/m/P/C_{max}$)	Toplam tamamlanma zamanı en küçükleme	Problemin çözümü için karınca koloni sistemi önerilmiştir. Önerilen yaklaşım, diğer metasezgisellerle (GA, SA ve NS) ile karşılaştırılmıştır.	ACS, diğer metasezgisellerden daha iyi sonuç vermiştir.	1. Palmer yönteminden farklı yönlendirme kuralları kullanılabilir. 2. ACS, incelenmeye değer çeşitli opsiyon ve parametre kümeleri sunar. 3. ACS farklı çözme problemlerine de uygulanabilir. 4. Bu çözme problemi için, diğer karınca koloni eniyileme yöntemlerinde araştırma sürdürülebilir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Çalışmaları

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Diaz ve French, 1983	Literatür incelemesi	n -iş, tek-makine çok-amaçlı çizelgeleme problemi	1.Toplam tamamlanma zamanı 2.Ortalama tamamlanma zamanı 3.Pozitif gecikme zamanı 4.Ortalama pozitif gecikme zamanı en küçüklenmesi	Mevcut literatürdeki çok amaçlı çizelgeleme algoritmalarının incelenmiştir. İncelenen yaklaşımlar: 1. Van Wassenhove ve Gelders in yaklaşımı ve bununla ilgili diğer yaklaşımlar 2.Kontrol edilebilir işlem zamanlı etkin çizelgeler oluşturan algoritmalar 3.Tam sayılı programlama yaklaşımlar 4.Çok nitelikli değer fonksiyonu yaklaşımları	Tek amaçlı çizelgeleme, literatürde geniş yer kaplarken, çok amaçlı çizelgeleme daha azdır. Bunun nedeni, çok amaçlı çizelgeleme için genel bir yaklaşımın olmamasıdır. Bu çalışmada bunu konudaki çalışmaları inceleyip, genel bir yaklaşım oluşturmaktadır.	Van Wassenhove ve Gelders yaklaşımı, kontrol edilebilir işlem zamanlı etkin çizelgeler ve tam sayılı programlama yaklaşımları daha geneldir. Ancak bunlar n -boyutlu çizelgeler geliştiren yaklaşımlar gibi görülmemektedir. Çok nitelikli değer fonksiyonu yaklaşımları ise umut verici görülmektedir.
Wilson, 1989	Mixed integer goal-programming formülasyonu ve deneysel karşılaştırma	n -iş, m -makine 2-amaçlı akış tipi çizelgeleme problemi	1.Toplam tamamlanma zamanı 2.Toplam akış zamanı en küçüklenmesi	n -iş m -makinelik akış tipi çizelgeleme problemi için çok sayıda değişken içeren mixed integer goal-programming formülasyonu yapılmış ve bu daha önce tanımlanan mixed integer goal-programming formülasyonu ile karşılaştırılmıştır. Ayrıca iki amaçlı amaç programlama yaklaşımı için pratik limitler tartışılmaktadır.	1.Az sayıda makine ile küçük ölçekli problemler için önerilen formülasyon en iyi sonucu verir. 2.Büyük ölçekli problemler için Selen ve Hott' un formülasyonu en iyidir. 3.15-ış ve 4-makinelik durum dönüm noktasıdır. Bu büyüklüğün üzerinde 2. durum baskındır. 4.20-ış ve 7-makinelik problemler için gerekli hesaplama zamanı oldukça kabul edilebilirdir.	Sonuçlar önerilen formülasyonun küçük problemler için iyi çalıştığını göstermektedir. Limitler kalacak şekilde tamsayılı programlama yaklaşımı verildiğinde, büyük ölçekli modelleri uzun süreli bilgisayar zamanı kullanmaksızın çözmek zordur. Bu nedenle tamsayılı programlama çözümünün özelliklerinden üretilen seçgiseller gelecekte gerçek ölçekli problemleri çözmek için en iyi yol gibi görülmektedir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Daniels ve Chambers, 1990	Sezgisel	n -ış, 2-makine 2-amaçlı akış tipi çözelme problemi	1. Toplam tamamlanma zamanı 2. Maksimum pozitif gecikme zamanı en küçükleme	2-amaçlı, n -ış ve 2-makine akış tipi çözelme problemi için sezgisel algoritma önerilmiştir ve rasgele üretilen 30-40 ışli problem üzerinde test edilmiştir.	Önerilen süreç, ortalama düzeyde bir çözüm sağlamıştır. Büyük ölçekli problemler için, sezgisel etkin çözüme oldukça yakın çözümler sağlarlar.	Akış tipi problemde, diğer performans ölçütleri (ortalama gecikme, geciken iş sayısı vb.) C_{max} ile birlikte değerdendirilebilir. Ayrıca, yönetiminde çözelmedeki yükünü azaltarak şekilde öncelikleri değerdendirilebilir. Son olarak, çok amaçlı çözelme diğer çözelme tiplerine de (paralel makine, çok-makine atölye vb.) uygulanabilir.
Ho ve Chang, 1991	Sezgisel ve deneysel karşılaştırma	n -ış, m -makine çok-amaçlı akış tipi çözelme problemi	1. Toplam tamamlanma zamanı 2. Toplam akış zamanı 3. Makine aylak kalma zamanının en küçükleme (Önerilen algoritma literatürde bugüne kadar üretilmiş sezgisellerin çözümlerinde oluşan işlemler arası boşluğu en küçükleme amaçları)	Literatürdeki mevcut çözelme sezgiselleri tanımlanmış, yeni bir sezgisel yöntem önerilmiştir. Bu sezgisel etkinliğini toplam tamamlanma zamanı, ortalama akış zamanı ve ortalama kullanım açısından literatürde bilinen sezgisellerle karşılaştırarak şekilde bir simülasyon analizi yapılmıştır. Önerilen sezgisel CDS sezgiseli tarafından oluşturulan başlangıç çözümünü değerdendirmektedir.	Önerilen algoritma çözelgeleri mevcut işlemler arası boşluğu en küçükleyerek değerdendirir. Önerilen algoritma, CPU zamanını orta ölçüde arttırırken, iyi olduğu bilinen 5 sezgiselden 3 kritere göre daha iyi sonuçlar vermiştir.	Denemeler işlemler arası boşluğu çözümleri değerdendirebilecek bir kavram olarak kullanılabileceğini göstermektedir.
Rajendran, 1992	Dal-sınır algoritması, sezgisel ve deneysel karşılaştırma	n -ış, 2-aşama 2-amaçlı akış tipi çözelme problemi	1. Toplam tamamlanma zamanı 2. Toplam akış zamanı en küçükleme	Problem çözümleri için bir dal-sınır tekniği ve iki sezgisel tanımlanmıştır. Önerilen sezgiseller Johnson yöntemi ile karşılaştırılmıştır.	Önerilen yaklaşımlar eniyi veya yakın-eniyi çözüme oldukça kısa sürede ulaşabilmektedir.	-

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Rajendran, 1994	Sezgisel ve deneysel karşılaştırma	n -ış, m -makine çok-amaçlı akış tipi çözelgeleme problemi	1. Toplam tamamlanma zamanı 2. Toplam akış zamanı 3. Makine aylak kalma en küçüklennesi	Problemin çözümü için bir sezgisel önerilmiştir. Ayrıca sezgisel tercih ilişkisi oluşturulmuştur. Önerilen algoritmanın sonuçları literatürdeki HC sezgiseli ile karşılaştırılmıştır. Ayrıca hücre tipi akış için benzer çalışma yapılmıştır.	Önerilen algoritma amaç değerlerinin en küçüklennesinde oldukça etkindir. Aynı sonuç hücre akış tipi için de geçerlidir.	-
Rajendran, 1995	Sezgisel ve deneysel karşılaştırma	n -ış, m -makine çok-amaçlı akış tipi çözelgeleme problemi	1. Toplam tamamlanma zamanı 2. Toplam akış zamanı 3. Makine aylak kalma zamanının en küçüklennesi	Problemin çözümü için bir sezgisel algoritma önerilmiştir. Bu algoritma, öncelikle toplam tamamlanma zamanı ve toplam akış zamanını en küçükleyecek şekilde geliştirilmiş, daha sonra makine aylak zamanı için genişletilmiştir. Algoritmanın sonuçları HC sezgiseli tarafından üretilen sonuçlarla karşılaştırılmıştır.	Önerilen algoritma, toplam tamamlanma zamanı ve toplam akış zamanının en küçükleyecek iki kriterli problemin çözümünde çok etkindir. Makine aylak zaman değeri de dikkate alınarak yapılan çok amaçlı çalışmada algoritma karşılaştırılan sezgiselden daha etkindir.	Çok amaçlı çözelgelerin geliştirilme çalışmalarına bir başlangıç oluşturmaya açınsından önemlidir.
Murata ve diğ., 1996	Genetik algoritma	n -ış, m -makine tek-amaçlı ve iki-amaçlı işlem stüresinin aralık şeklinde tanımlandığı akış tipi çözelgeleme problemi.	1. Toplam tamamlanma zamanı 2. Toplam akış zamanı 3. Maksimum aylak kalma 4. Toplam aylak kalma zamanının en küçüklennesi	Problemin öncelikle tek amaç durumu formüle edilmiş ve küçük ölçekli problemler için uygulanmıştır. Daha sonra iki amaç için genetik algoritma yaklaşımı önerilmiştir.	İki amaç için, oluşturulan çoklu baskın çözümlerden son çözümün seçimi, karar verici tarafından yapılabilir.	-

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Nagar ve diğ., 1996	Dal-sınır, genetik algoritma ve deneysel karşılaştırma	n -ış, 2-makine (iki aşama) 2-amaçlı akış tipi çizelgeleme problemi	1. Toplam tamamlanma zamanı 2. Ortalama akış zamanının en küçüklenmesi	Problemin çözümü için dal sınır yöntemi ve genetik algoritma tabanlı birleştirilmiş bir sezgisel algoritma (metasezgisel) ve iki modifikasyonu önerilmiştir. Bu algoritmaların performansı üç algoritma ile karşılaştırılmıştır. 1. Tam GA 2. Tam dal sınır 3. Sezgisel	Metasezgisel ve iki modifikasyonu, diğer algoritmalara göre daha iyidir. CPU zamanı ve çözüm kalitesi açısından metasezgisel ve ilk modifikasyonu 50'den fazla iş için kullanılamazken ikinci modifikasyonu büyük ölçekli problemler içinde uygundur.	Metasezgisel ve iki modifikasyonu ile her iki yöntemin de avantajları birleştirilmiştir.
Neppalli ve diğ., 1996	Genetik algoritma ve deneysel karşılaştırma	n -ış, 2-makine (iki aşama) 2-amaçlı akış tipi çizelgeleme problemi	1. Toplam tamamlanma zamanı 2. Toplam akış zamanının en küçüklenmesi	Problemin çözümü için bir genetik algoritma tabanlı iki yaklaşım önerilmiştir: 1. Vektör değeri yaklaşımı (Vector evaluation approach) 2. Ağırlıklar toplamı yaklaşımı (Weighted-sum approach) Her iki yaklaşımın en iyi çalışma şartları (başlangıç nüfusu, etki faktörü, ölçümleme mekanizması, bitiş kriteri) tespit edilmiştir. Daha sonra her iki yaklaşımın tespit edilen şartlardaki performansı ile Rajendran'ın algoritması tarafından üretilen sonuçlarla karşılaştırılmıştır. En son olarak her iki yaklaşım birbirleri ile karşılaştırılmıştır.	Önerilen algoritma, Rajendran'ın algoritması tarafından üretilen sonuçlardan daha iyi sonuçlar vermiştir. İki yaklaşım birbiri ile karşılaştırıldığında vektör değeri yaklaşımı, ağırlıklar toplamı yaklaşımından daha etkilidir.	Genel anlamda, genetik algoritma tabanlı iki yaklaşımın çeşitli parametrelili durumları, birçok iki amaçlı çizelgeleme problemine uygulanabilir. Operatör karşın koşullarında spesifik bilgiler problemin içine katılarak iki yaklaşımın performansı güçlendirilebilir. Ayrıca her iki amaç da eşit şekilde gözönüne alınarak çalışmalar yapılabilir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelgeme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Sridhar ve Rajendran, 1996	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -makine çok-amaçlı akış tipi çözelgeme problemi	1. Toplam tamamlanma zamanı 2. Toplam Akış zamanı 3. Makine aylak kalma zamanının en küçüklmesi	Genetik algoritmaya dayanan bir sezgisel geliştirilmiştir. Algoritmanın performansı, HC sezgiseli ile karşılaştırılmıştır. Ayrıca bu formülasyon hücre tipi üretim sistemine uyarlanmıştır.	Önerilen algoritmanın performansı, HC algoritmasından daha iyi sonuçlar vermiştir.	Çok amaçlı çözelgeme üzerine karşılaştırmalı bir çalışma olması açısından önemlidir.
Murata ve Ishibuchi, 1997	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -aşamalı çok-amaçlı akış tipi çözelgeme problemi	1. Toplam tamamlanma zamanı 2. Maksimum pozitif gecikmenin 3. Toplam akış zamanı	Üç amaçlı çözelgeme probleminin çözümü için önerilmiş çok amaçlı genetik algoritmaların performansları karşılaştırılmıştır.	Çok amaçlı genetik yerel arama en kısa CPU zamanı ile en iyi performansı göstermiştir.	-

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Cavalieri ve Gaiardelli, 1998	Melez genetik algoritma ve deneysel karşılaştırma	n -ış, m -makine 2-amaçlı akış tipi çözelleme problemi	1. Toplam tamamlanma zamanı 2. Ortalama pozitif gecikmenin en küçüklenmesi	İstisacı firmasındaki akış hatında bir uygulama yapılmıştır. Problemin çözümü için iki genetik algoritma tabanlı yaklaşım önerilmiştir: 1. GA1; sadece GA dir ve hatta işlerin sıralamasını yapmakta, 2. GA2; GA, EDD kuralı ile birleştirilmekte ve yine hatta işleri sıralamaktadır. Her iki algoritma, dinamik doğuş oranlı, dinamik nüfus büyüklüğü, çok işlemli değerdendirme kriteri, adaptif çaprazlama ve değişim oranları olacak şekilde(adaptif GA) tekrar tanınlanmıştır. Öncelikle, adaptive GA1 ve GA2, statik durumları ile karşılaştırılmıştır. Daha sonra adaptif GA1 ve GA2, dal-sınır ile karşılaştırılmıştır. En son olarak her iki yaklaşımın haftalık üretim karışımının değışimine karşı gösterdiği sağlamlık test edimiştir.	1. Adaptive GA1 ve GA2, statik durumlarına göre daha iyi sonuçlar vermiştir. 2. GA1 ve GA2, dal-sınır ile karşılaştırıldığında, küçük parti büyüklüğü için dal-sınır ile aynı sonuçları vermekte iken, sayı artıkça, eniyi çözüm olmasına rağmen, dal-sınır aksine, kabul edilebilir hesaplama zamanına sahiptir. 3. Haftalık üretim karışımının değışimine karşı GA1, GA2' den daha sağlamdır.	1. GA1 performans ile hesaplama zamanı azasında iyi bir takas sağlanmaktadır ve farklı şartlarda da çalışabilmektedir. 2. GA olasılığa bağı bir yöntem olduğu için performansını değerdendirmek için çok sayıda deneme yapmak gerekir.
Ishibuchi ve Murata, 1998	Genetik algoritma ve deneysel karşılaştırma	n -ış, m -makine çok-amaçlı akış tipi çözelleme problemi	1. Toplam tamamlanma zamanı 2. Maksimum pozitif gecikme 3. Toplam akış zamanı en küçüklenmesi	Problemin çözümü için bir melez genetik algoritma önerilmiştir. Yaklaşım, genetik operatörler tarafından bulunan her çözüme yerel arama uygular. Öncelikle iki ve üç amaç için örnek problemler çözülmüş, daha sonra önerilen algoritma, diğer çok amaçlı GA' larla ve rasgele örnekleme karşılaştırılmıştır.	Çok amaçlı problemler için, diğer algoritmalarla karşılaştırıldığında, birçok çözümden, önerilen algoritmanın yüksek performans gösterdiği görülmektedir. Hesaplama zamanı diğer algoritmalarla hemen hemen aynıdır.	Çok amaçlı melez algoritma (GA ve komşuluk arama) uygulamast olması açısından önemlidir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Sivrikaya-Serifoğlu ve Ulusoy, 1998	Dal-sınır algoritması ve deneysel karşılaştırma	n -ış, 2-makine 2-amaçlı akış tipi çözelme problemi	1. Toplam tamamlanma zamanı 2. Ortalama akış zamanının en küçükleme	Problemin çözümü için üç dal-sınır tekniği tanımlanmıştır. Her yaklaşım için alt sınırlar oluşturulmuştur. Üst sınır oluşturmak için iki akış tipi sezgisel önerilmektedir. Amaç değerlerinin ağırlıklı toplamı (w) en küçüklemiştir.	Üst sınır, oldukça iyi iken, alt sınır hala zayıf kalmaktadır. Alt sınır eniyi çözümlerle karşılaştırıldığında, alt sınırın w değeri oldukça kötüleştiği görülmektedir.	Alt sınır performansını arttırmak için, akış zamanı alt sınırı üzerinde çalışma yapılması gerekir.
Tagami ve Kawabe, 1998	Genetik algoritma ve deneysel karşılaştırma	n -ış, m -makine 2-amaçlı akış tipi çözelme problemi.	1. Toplam tamamlanma zamanı 2. Maksimum gecikme zamanı en küçükleme	Çok amaçlı eniyileme için Pareto komşuluk aramaya dayanan bir genetik algoritma yaklaşımı (GA1) önerilmiştir. Yaklaşımın amacı, takas alanı komşuğuna dağılımı baskın çözümlerin bir kümesini oluşturmaktır. Yaklaşımın etkinliğini değerlendirmek için, çok amaçlı sıralamalı GA yaklaşımı (GA2) ve çok amaçlı paylaşılmalı GA (GA3) yaklaşımları ile karşılaştırılmıştır.	Önerilen GA yaklaşımı, diğer standart GA yaklaşımları ile karşılaştırıldığında, dağıtılmış çözümleri bulmayı oldukça başarılı.	Çeşitli çok amaçlı eniyileme problemlerinin genel etkinliğini ortaya koymak için sunulan işlemlerin uygulaması yapılabilir.
Chakravarty ve Rajendran, 1999	Sezgisel ve deneysel karşılaştırma	n -ış, m -makine 2-amaçlı akış tipi çözelme problemi	1. Toplam tamamlanma zamanı 2. Maksimum pozitif gecikme zamanının en küçükleme	Problemi için sezgisel algoritma önerilmiştir. Sezgisel başlangıç çözümü için 3 yöntem kullanmakta ve bu çözümü tavlama benzetimi ile güçlendirmektedir. Algoritmanın performansı, diğer bir sezgisel ile karşılaştırılmıştır.	Önerilen algoritmanın performansı, mevcut algoritmadan daha iyi sonuç vermiştir.	-
Chen ve Smith, 1999	Genetik algoritma ve deneysel karşılaştırma	n -ış, 2-makine 2-amaçlı akış tipi çözelme problemi.	Erken ve gecikme maliyeti	Problemin çözümü için genetik algoritma yaklaşımı önerilmiştir. Öncelikle iki çaprazlama operatörünün (öncelik koruyucu çaprazlama ve düzgün sıralama-tabanlı çaprazlama) etkinlikleri karşılaştırılmış, daha sonra probleme öncelik kısıtları ilave edilerek araştırma alanının azaltılması sağlanmıştır.	Öncelik kısıtları ilave edildiğinde öncelik koruyucu çaprazlama, düzgün sıralama-tabanlı çaprazlamaya göre daha iyi performans göstermiştir. Öncelik koruyucu çaprazlama, düzgün sıralama-tabanlı çaprazlamadan daha hızlı bir noktada birleşmiştir.	Akış tipi problemleri için araştırma alanını azaltmak için öncelik kısıtları kullanılır. Kısıtlara hizmet eden bir çaprazlama operatörü (PPX) daha faydalı olmaktadır.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelgeme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Gupta ve diğ., 1999	Yasak arama ve deneysel karşılaştırma	n -ış, 2-makine 2-amaçlı akış tipi çözelgeme problemi	1. Toplam tamamlanma zamanı 2. Toplam akış zamanının en küçüklenmesi	Problem çözümleri için bir yasak arama algoritması önerilmiştir. TS sezgiselinin üzerinde dört faktörün etkisinin deneysel analizi yapılmıştır: başlangıç çözümü, hareket tipi, komşuluk büyüklüğü ve liste büyüklüğü. Daha sonra en iyi kombinasyon belirlenmiştir. Önerilen algoritmanın etkinliği Rajendran tarafından önerilen sezgiselle karşılaştırılmıştır.	Önerilen algoritmanın etkinliği Rajendran tarafından önerilen sezgiselden ve böylece Neppalli tarafından önerilen genetik algoritmadan daha iyidir.	1. m -aşamalı akış tipini çözmek için de TS sezgiseli kullanılabilir. 2. Birincil ve ikincil kriterler için en iyi TS faktör kombinasyonu kullanılarak, çözelgeme problemlerinin çözümü için TS kullanımını genişletilebilir. 3. TS sezgiselinin kullanımını, sınırlandırılmış çözelgeme problemlerinde gelişim sağlamak için araştırılabilir. 4. İki kriterli çözelgeme problemlerini çözmek için yerel arama tekniklerini kullanmak faydalı ve ilginç olabilir.
Sayın ve Karabati, 1999	Dal-sınır algoritması ve deneysel karşılaştırma	n -ış, 2-makine 2-amaçlı akış tipi çözelgeme problemi	1. Toplam tamamlanma zamanı 2. Toplam bitiş zamanının en küçüklenmesi	Problem çözümleri için etkin çözümler kütmesi tam olarak belirlenene dek tek amaçlı çözelgeme problemini iteratif olarak çözen bir dal-sınır tekniği tanımlanmıştır. Genel durumlar için önerilen yeni bir alt sınır oluşturulmuştur ve çift dal-sınır aramayı önlemek için bir koordinasyon süreci geliştirilmiştir.	Hesaplama sonuçları, önerilen algoritma ile uygun ölçüdeki problemler uygun bir sürede çözülebilmektedir.	Bu çözüm algoritması çok makine problemi adapte edilebilir. Ancak hesaplama açısından bu tatin edici olmayacağı için, algoritmanın sezgisel versiyonları tavsiye edilebilir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Tagami ve Kawabe, 1999	Genetik algoritma ve deneysel karşılaştırma	n iş, m -makine 2-amaçlı akış tipi çözelme problemi.	1.Toplam tamamlanma zamanı en küçüklenmesi 2.Maksimum gecikme zamanı en küçüklenmesi	Problem çözümü eniyileme için pareto partitioning yöntemini kullanan bir genetik algoritma yaklaşımı (GA1) önerilmiştir. Yaklaşımın amacı, takas alanı komşuluğuna dağılmış baskın çözümlerin bir kümesini oluşturmaktır. Yaklaşımın etkinliğini değerlendirmek için, çok amaçlı sıralanmış GA yaklaşımı (GA2) ve çok amaçlı ağırlıklandırılmış toplam GA (GA3) yaklaşımları ile karşılaştırılmıştır.	Önerilen GA yaklaşımı, diğer standart GA yaklaşımları ile karşılaştırıldığında, dağılmış çözümleri bulmayı oldukça başarılı.	-
Tamura ve diğ., 1999	Sezgisel ve deneysel karşılaştırma	n -iş, 2-makine (A ve B) çok-amaçlı akış tipi çözelme problemi	1.Toplam tamamlanma zamanı 2.A işleminin hazırlık zamanları sayısının 3.İşlerin değişim oranının toplamının 4.Sürekli üretim kısıtını bozma cezasının en küçüklenmesi	Çok amaçlı birleşik eniyileme problemlerini çözmek için uygun takas yöntemlerini sağlayan etkin bir metasezgisel yaklaşım önerilmiştir. Öncelikle, çok amaçlı birleşik eniyileme problemleri için, aile seçicilik kavramı ile bir genetik algoritma kullanılarak pareto eniyi çözümler oluşturulmuştur. Daha sonra, uygun takas yöntemine dayanan bir metasezgiselle karar vericinin tercih ettiği bir çözüm bulmaya çalışılır. Yaklaşımın aile seçicilik kavramı kullanılıp kullanılmaması stratejileri karşılaştırılmıştır.	Çok amaçlı birleşik eniyileme problemlerini çözmek için uygun takas yöntemlerini sağlayan etkin bir metasezgiseldir. Yaklaşımında, aile seçicilik kavramı kullanıldığında, daha iyi sonuçlar vermiştir.	Üretim yönetimindeki birçok performans değerlendirme problemi çok amaçlı olduğu için, önerilen yöntem birçok gerçek üretim yönetim sistemi için kullanılabilir. Yöntemi güçlendirmek için, aspiration düzeylerin nasıl hazırlanacağı, bireylerin Pareto eniyisinin nasıl değerlendirileceği, birçok gerçek probleme uygulanarak araştırılabilir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Yeh, 1999	Dal-sınır algoritması ve deneysel karşılaştırma	n -ış, 2-makine 2-amaçlı akış tipi çözelleme problemi ($n/2$ /flowshop/ $\alpha F + \beta C_{max}$)	1. Akış zamanı 2. Toplam tamamlanma zamanı en küçüklenmesi	Problemin çözümü için bir dal-sınır algoritması (YGAXB) (iki alt-sınır ve bir üst-sınır) önerilmiştir. Yaklaşım, üst-sınır için pairwise exchange algoritması ile aktif bir sezgisel (YGAX) kullanır. Üst sınır için, aktif bir sezgisel kısmı ve tam değerler (pairwise exchange algoritması kısmı ve aktif bir sezgisel kısmı), Nagar'ın aktif bir sezgisel kısmı ve tam değerleri (NGX) ile karşılaştırılmıştır. İlk alt-sınır Nagar'ın, ikinci alt-sınır ise, Johnson algoritması kullanılarak bulunan birinci alt-sınıra ağırlıklı ayıak kalma zamanının eklenmesi ile oluşturulur. Önerilen algoritma sonuçları yine Nagar'ın değerleri (NGAXB) ile karşılaştırılmıştır.	Üst sınır için, YGAX aktif bir sezgisel kısmı ile bulunmuş üst-sınır, Nagar'ın tam değerlerinden daha iyidir. Çalışma süreleri açısından da üstünlüğü vardır. İş sayısı arttıkça bu iyileşme farkı büyür. YGAXB algoritması da Nagar'ın algoritmasından (NGAXB) oldukça iyidir. Çalışma süreleri açısından da üstünlüğü vardır. İş sayısı arttıkça bu iyileşme farkı büyür.	İşlem verileri (ayıak kalma gibi) küçükse, YGAX üst-sınırı ve YGAXB' in her iki alt sınır değeri arasındaki fark küçüktür. Ancak, fark büyükse, problemin çözümü zordur. Bu nedenle, daha sıkı alt ve üst sınır değerleri bulmak gerekir.
Ishibashi ve diğ., 2000	Genetik algoritma ve deneysel karşılaştırma	n -ış, m -makine çok-amaçlı akış tipi çözelleme problemi	1. Toplam tamamlanma zamanı 2. Toplam akış zamanı 3. Toplam pozitif gecikmenin en küçüklenmesi	Problemin çözümü için daha önce tek amaçlı eniyileme problemleri için hızlı ve sağlam aramalar sağlayan güçlü bir GA önerilmiştir. Önerilen algoritmanın iki operatörü akış tipi problemleri için uygun hale getirilmiştir. Algoritmanın sonuçları kanonik GA ile karşılaştırılmıştır.	Önerilen algoritmanın çok amaçlı akış tipi çözelleme problemleri için, çözüm kümesinin kalitesi ve pareto-eniyi çözümleri açısından kanonik GA' dan daha iyidir. Nesil seçimi, arama hızını ivmelenmeye, kullanılan operatör güvenilir ve sağlam arama performansına yol açar.	Önerilen algoritma diğer çok amaçlı eniyileme problemlerine uygulanabilir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Mabed ve diğ., 2000	Genetik algoritma ve deneysel karşılaştırma	n -iş, m -aşamalı 2-amaçlı akış tipi çözelme problemi	1. Toplam tamamlanma zamanı 2. Toplam pozitif gecikmenin en küçüklenmesi	Problem için çözüm için genetik algoritma önerilmiştir. Öncelikle çeşitli seçim stratejileri (ağırlıklı toplam, paralel, NSGA, NDS, WAR, Elitizm) tanımlanmış ve performansları karşılaştırılmıştır. Değişim için birkaç yaklaşım (genotypic, phenotypic ve melez) gösterilmiş ve çözüme etkileri gösterilmiştir. Daha sonra, GA ile yerel arama algoritması melezleştirilmiştir.	Non-pareto stratejileri (ağırlıklı toplam, paralel), çok amaçlı durum için uygun değildir. NSGA, NDS, WAR stratejileri, çok amaçlı probleme oldukça uygun görülmektedir. Elitizm yaklaşımı ise oldukça güçlüdür. Phenotypic değişim ve Genotypic değişim yaklaşımının birleşimi ise aramayı dikkate değer şekilde güçlendirir. Melez yaklaşım büyük ölçüde problemler için çözümü güçlendirebilir.	-
TKindt ve diğ., 2000	Karınca koloni leri eniyileme ve deneysel karşılaştırma	n -iş, 2-makine 2-amaçlı akış tipi çözelme problemi	1. Toplam tamamlanma zamanı 2. Toplam akış zamanının en küçüklenmesi	Problem için çözüm için, bir KKE sezgisel (SACO) önerilmiştir. Önerilen algoritma, literatürde bu problem için önerilmiş diğer sezgiseller ile karşılaştırılmıştır.	SACO' nun mevcut algoritmalarından daha iyi sonuçlar verdiği gösterilmiştir.	-
Gupta ve diğ., 2001	Sezgisel ve deneysel karşılaştırma	n -iş, 2-makine 2-amaçlı akış tipi çözelme problemi	1. Toplam tamamlanma zamanı (birincil amaç) 2. Toplam akış zamanı (ikincil amaç) en küçüklenmesi	Öncelikle problemi çözen yaklaşımlar incelenmiş, eniyi çözüm için oluşturulan baskınlık koşulları tanımlanmıştır. Bu baskınlık kuralları, polinomsal olarak çözülebilen durumları tanımlamak için kullanılmıştır. Daha sonra, eniyi veya yakın-eniyi çizelgeler için iki polinomsal sezgisel algoritma önerilmiştir. Bu sezgiseller mevcut sezgisellerle karşılaştırılmıştır.	Insert algoritması, uygun çözüm süresi ile diğer sezgisellerden daha iyi sonuç vermektedir.	Diğer iki amaçlı problemler için de önerilen ilk üç algoritmanın etkinliği değerlendirilebilir. Ayrıca etkin baskınlık koşulları ve bunların yerel arama algoritmalarında kullanımı üzerine çalışma yapılabilir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Talbi, Rahoual ve diğ., 2001	Genetik algoritma ve deneysel karşılaştırma	n -ış, m -makine 2-amaçlı akış tipi çözelleme problemi	1. Toplam tamamlanma zamanı 2. Toplam pozitif gecikmenin en küçükleme	Problemin çözümü için genetik algoritma önerilmiştir. Özellikle çeşitli seçim operatörleri (WSGA, VEGA, NSGA, NDS, WAR, Elitism) tanımlanmış ve performansları karşılaştırılmıştır. "A" parametresinin (elitism pressure) etkisi tartışılmıştır. Değişim oluşturmada üç paylaştırma yaklaşımı (genotypic, phenotypic ve combined) uygulanmış ve çözüme etkileri gösterilmiştir. Daha sonra, GA ile yerel arama algoritması melezleştirilmiştir. Bu yaklaşım için paralel bir model önerilmiştir ve etkisi ortaya konmuştur.	NSGA, NDS, WAR stratejileri, çok amaçlı probleme oldukça uygun görülmektedir. Dışık değeri "A" parametresi, yoğunlaşmayı azaltırken, yüksek değeri "A" parametresi bozulma riskini arttırır. Phenotypic değişim yaklaşımı daha iyi ve büyük Pareto sınırları oluşturmaktadır. Genotypic değişim yaklaşımı da iyi sonuçlar oluşturmaktadır. Her iki yaklaşımın birleşimi ise aramayı dikkate değer şekilde güçlendirir. Melez yaklaşım büyük ölçütlü problemler için çözümlü güçlendirilebilir. Paralel bir model ise, ntifus büyüklüğünü ve nesil sayısını arttırmakta ve böylece daha iyi sonuçlar elde etmektedir.	Problem sadece iki kriterli problem için çözülmüştür. Çok kriter için de kolaylıkla uygulanabilir. Bundan başka, diğer melez çizelgelerle değeriendirilmelidir. GA'nın, daha ileri arama teknikleri (yasak arama gibi) ile birleşimi daha iyi sonuçlar verebilir.
Zhao ve diğ., 2001	Genetik algoritma ve deneysel karşılaştırma	n -ış, m -makine çok-amaçlı akış tipi çözelleme problemi	1. Toplam tamamlanma zamanı 2. Ortalama pozitif gecikmenin 3. Ortalama akış zamanı en küçükleme	Probleminin çözümü için, bir genetik algoritmaya kısmi enumeration seçim yöntemi adapte edilmiştir. Yaklaşımın sonuçları diğer iki genetik algoritma ile karşılaştırılmıştır.	Kısmi enumeration seçim yöntemine dayalı genetik algoritma, diğer yaklaşımlardan daha iyi sonuç vermiştir.	-

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Basseur ve diğ., 2002	Genetik algoritma ve deneysel karşılaştırma	n -ış, m -makine 2-amaçlı akış tipi çizelgeleme problemi	1. Toplam tamamlanma zamanı 2. Toplam pozitif gecikmenin en küçüklenmesi	Çok amaçlı evrimsel algoritmaların dizaynında karşı karşıya kalınan problemleri çözebilecek bir yöntem olarak bir uyumlu pareto genetik algoritma sunulmaktadır ve bu mekanizma akış tipi çizelgeleme problemlerine uygulanmıştır. Burada; 1. Farklı genetik operatörler kullanılmaktadır. 2. Birleştirilmiş bir paylaşım, karar alanı gibi, amaç alanına uygulanır. 3. Amaç alanındaki uygun yerlerin büyüklüğü otomatik olarak hesaplanır. 4. Pareto GA ile bir memetik algoritma yaklaşımı birleşimi ile melez bir yaklaşım uygulanmıştır. Ayrıca, çok amaçlı evrimsel algoritmaların performansını değerlendirmek için iki indikatör oluşturulmuştur. Algoritmanın performansı daha önce bu araştırma grubu tarafından önerilen algoritma sonuçları ile karşılaştırılmıştır.	Önerilen GA yaklaşımının indikatörleri daha önce geliştirilen algoritmaya göre daha iyi sonuç vermiştir.	-
Chang ve diğ., 2002	Genetik algoritma ve deneysel karşılaştırma	n -ış, m -makine çok-amaçlı akış tipi çizelgeleme problemi	1. Toplam tamamlanma zamanı 2. Toplam akışı 3. Toplam pozitif gecikme 4. Maksimum pozitif gecikme zamanı en küçüklenmesi	Probleminin çözümü için, Pareto eniyi çözümleri araştırarak bir yaklaşım önerilmiştir. Yaklaşımın etkinliği, değişken ağırlıklıdırma yaklaşımı ile karşılaştırılmıştır. Bu ağırlıklıdırma yaklaşımını, genetik algoritma ve genetik yerel melez algoritma çözüm ortamlarında test edilmiştir.	Önerilen ağırlıklıdırma yaklaşımı, değişken ağırlıklıdırma yaklaşımına göre, araştırılan ortalama Pareto eniyi çözümsayısı, ortalama non-dominated çözüm sayısı, non-dominated çözüme oranı ve CPU zamanı açısından daha üstündür.	Farklı amaçlara sahip çizelgeleme problemlerini çözmek için önerilen yaklaşım ile bulanık mantık birarada kullanılabilir. Ayrıca, diğer arama algoritmaları (yasadak arama, tav'lama benzetimi, ordinal eniyileme vb.) ile güçlendirilebilir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Gupta, Hennig ve diğ., 2002	Tavlama benzetimi, threshold accepting (TA), çok aşamalı arama, genetik algoritma ve dencyssel karşılaştırma	n -ış, 2-aşama 2-amaçlı akış tipi çizelgeleme problemi (F2/Fh(C2/C _{max}))	1. Toplam tamamlanma zamanı (birincil amaç) 2. Toplam akışı (ikincil amaç) 3. Toplam ağırlıklı akışı (ikincil amaç) 4. Toplam ağırlıklı pozitif gecikmenin (ikincil amaç) en küçüklükmesi	Problemin çözümü için tavlama benzetimi, eşik kabulü (threshold accepting), yavaş arama, çok aşamalı arama, genetik algoritma yaklaşımları incelenmiştir. Bu sezgisellerin parametrelerinin etkisi (pairwise interchange ve shift komşulukları, şema düzenleri vb.) ve başlangıç çözümleri karşılaştırılmıştır. Ayrıca önerilen yerel araştırma yaklaşımı değerlendirilmiş ve performansı diğer algoritmalarla karşılaştırılmıştır.	1. F2/Fh(C2/C_{max}) problemi için, pairwise interchange komşuluğu, shift komşuluğuna göre daha iyi sonuçlar verir. 2. Eşik kabulü için, değişken eşik şeması, azalan eşik şemasından daha üstündür. SA için, soğutma şeması amaç değerinde bir artış sağlamak için kullanılır. TS için, pozisyon niteliği küçük ölçekli yavaşlar listesi ile birlikte en iyi sonucu verir. 3. Çok aşamalı arama için, yüksek aşama komşuluğu seçimi çözüm kalitesini etkiler, pairwise interchange ve shift komşulukları ise en iyi çalışan komşuluklardır. 4. C2=ΣCi ve C2=ΣwiCi için, çok aşamalı arama en iyi sonucu vermiştir. 5. C2=ΣwiCi için, SA, en iyi sonucu vermiştir. 6. Genetik algoritma yaklaşımları oldukça yavaş ve diğer yerel sezgisellerle birlikte değerlendirilmedikçe zayıftır.	1. m-aşamalı iki amaçlı akış tipi çizelgeleme problemlerini çözmek için önerilen yaklaşımın kullanımını genişletilebilir. 2. Çeşitli polinomsal sınırlı çizelgeleme problemlerinin geliştirme potansiyelini bulmak için, yerel arama sezgisellerinin kullanımını araştırılabilir. 3. Diğer teslim zamanıyla ilgili kriterler (gecikme ve erken cezaları gibi) 4. Farklı birincil amaçlar kullanılabilir. 5. Tek bir algoritma içine birkaç stratejinin birleştirilmesi araştırılabilir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Gupta, Kruger ve diğ., 2002	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine kontrol edilebilen işlem zamanına sahip ve çeşitli makine merkezlerinde çok sayıda aynı tip paralel makinelerin olduğu serbest zamanlı melez 2- amaçlı akış tipi çözelleme problemi.	Erken ve gecikme cezalarının, ağırlıklandırılmış tamamlanma zamanının ve teslim zamanı atamalarındaki maliyetlerin toplamının en küçüklenmesi	Bu problemin çözümü için, klasik problemlerin sezgisel çözümleri için bilinen yaklaşımlar oluşturulmuş ve iş ekleme tekniğine dayanan yardımcı algoritmalar ve yerel aramaya dayanan iteratif algoritmalar önerilmiştir. En iyi çözümü belirlemek için uygun komşuluklar belirlenmiş ve bu sezgisellerin uygun kontrol mekanizmalarının belirlenmesi üzerinde durulmuştur.	1. Teslim zamanına dayanan eniyileme kriteri için, tüm sıralama pozisyonları, eklenecek iş için aday pozisyonlar olarak ele alınmasına gerek yoktur. 2. İş teslim zamanlarının hesaplanmasında başlangıç sıklık değeri ekleme algoritmasının çözüm kalitesini etkiler. 3. Eklmeye dayanan algoritma, öncelik kuralına dayanan algoritmalarından daha uygun yardımcı algoritmadır. 4. Önerilen iteratif algoritmaların performansı, parametrelerin sayısına bağlı olmasına rağmen, bazı parametrelerin değişimi konusunda oldukça sağlamdır.	1. Bu çalışmada gözönlüne alınan komşuluklar daha da düzenlenebilir. Örneğin, bir merkezdeki makinalara işlerin dağılımı vb. 2. Birleşik komşuluk kuralları (shift ve pairwise) oluşturulabilir. 3. Ekleme algoritmasında, çözelleme yapısı içinde güçlendirici süreçler aranabilir. 4. Daha başka kısıtlar gözönlüne alınabilir.
Pugazhendhi ve diğ., 2002	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine 2-amaçlı permütasyon olmayan akış tipi çözelleme problemi	1. Toplam tamamlanma zamanı 2. Toplam akış zamanının en küçüklenmesi	Problemin çözümü için permütasyon olmayan çözelleme oluşturmak için basit bir sezgisel önerilmiştir. Önerilen algoritmanın diğer kurallara üzerindeki etkisi gösterilmiş ve çözüm stresi NEH ve Ishibuchi algoritmaları ile karşılaştırılmıştır.	Önerilen yaklaşım, permütasyon olmayan çözelleme oluşturmada etkindir ve çözelleme performans kriterlerini güçlendirmektedir. Ayrıca bunu kısa bir hesaplama stresinde yapmaktadır.	-
T'K'indt ve diğ., 2002	Karınca koloni leri eniyileme, tavlama deneysel karşılaştırma	n -iş, 2-makine 2-amaçlı akış tipi çözelleme problemi	1. Toplam tamamlanma zamanı 2. Toplam bitiş zamanının en küçüklenmesi	Probleminin çözümü için, yerel arama ile tavlama benzetimi yaklaşımlarını birleştiren bir KKE sezgiseli (SACO) önerilmiştir. Önerilen algoritma, literatürde bu problem için önerilmiş diğer sezgiseller ile karşılaştırılmıştır.	SACO' nun mevcut algoritmalarından daha iyi sonuçlar verdiği gösterilmiştir.	-

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Yeh, 2002	Memetic algoritma ve deneysel karşılaştırma	n -iş, 2-makine 2-amaçlı akış tipi çizelgeleme problemi ($n/2/\text{flowshop}/\alpha F+\beta C_{\max}$)	1. Akış zamanı 2. Toplam tamamlanma zamanı en küçüklenmesi	Problem çözümü için standart GA, melez yerel arama metodu ve hırslı sezgisel metot ile birleştirilen bir memetik algoritma önerilmiştir. Önerilen algoritma sonuçları dal-sınır algoritması, en-iyi bilinen bir sezgisel ve standart GA ile karşılaştırılmıştır.	Önerilen MA algoritması, uygun çözüm stresi ile iyi sonuçlar vermektedir.	-
Allahverdi, 2003	Teorem-ispat, sezgisel ve deneysel karşılaştırma	n -iş, 2- ve m -makine iki kriterli akış tipi çizelgeleme problemi	1. Toplam tamamlanma zamanı 2. Ortalama akış zamanının en küçüklenmesi	Öncelikle problemin çözümü sağlayan dal-sınır algoritması için 2 baskınlık ilişkisinin teoremi ve ispatı yapılmıştır. Daha sonra bu problem için literatürde mevcut tüm sezgisellerin karşılaştırılması yapılmış ve önerilen ekleme stratejileri farklı 3 yeni sezgisel algoritmanın (AAH1, AAH2 ve AAH3) performansı bunlarla karşılaştırılmıştır. Bu	Öncelikle, iki-makine problem için önerilen sezgiseller ve diğer tüm sezgiseller karşılaştırıldığında, AAH1, AAH2 ve AAH3 algoritmaları en iyi sonucu vermiştir. AAH3, AAH1' den ve AAH1, AAH2' den daha iyidir. m -makine problem için önerilen sezgiseller ve diğer tüm sezgisellerin en iyisi olan AAH2 ve AAH3 algoritmaları en iyi sezgiseller ile karşılaştırıldığında, AAH1, AAH2 den ve AAH1, AAH3' den daha iyidir. Son olarak önerilen 2 baskınlık ilişkisinin etkin olduğu gösterilmiştir.	Önerilen ve mevcut tüm sezgiseller, hazırlık zamanlarının ihmal edildiği durumlar için geliştirilmiştir. Bu zamanlar da gözönüne alınarak da denemeler yapılabilir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çizelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Ishibuchi ve diğ., 2003	Genetik algoritma ve deneysel karşılaştırma	n -ış, m -makine çok-amaçlı akış tipi çizelgeleme problemi	1. Toplam tamamlanma zamanı 2. Maksimum pozitif gecikmenin 3. Toplam akış zamanı en küçüklenmesi	Probleminin çözümü için daha önce tanımlanmış yerel arama için başlangıç çözümlere iyi bireyler seçen, her başlangıç çözümü için uygun yerel arama yöntemi belirleyen çok amaçlı bir genetik arama algoritması önerilmiştir. Daha sonra, hesaplamalarda genetik algoritma ile yerel arama arasında dengenin önemi gösterilmiştir. Önerilen algoritmanın performansı, diğer algoritmaların performansları karşılaştırılmıştır. Son olarak, yerel aramanın, çok amaçlı memetik algoritmaların oluşturulmasında evrimsel çok amaçlı eniyileme algoritmaları ile kolaylıkla birleştirilebileceği gösterilmiştir.	Her üç algoritma benzer çözüm stirelerine sahiptir. Diğer algoritmaların performansları yerel arama ile melezleştirildiğinde güçlenmektedir.	1. Yerel arama ve GA arasındaki denge için kullanılan üç parametrenin dinamik kontrolü için çalışma yapılabilir. 2. Çok amaçlı genetik yerel arama, diğer çok amaçlı algoritmalarla karşılaştırması yapılabilir. 3. Yerel arama ve GA arasındaki dengeyi sağlayan çok amaçlı bir memetik algoritma oluşturulabilir.
TKindt ve diğ., 2003	Dal-sınır algoritması, sezgisel ve deneysel karşılaştırma	n -ış, 2-makine 2-amaçlı akış tipi çizelgeleme problemi	1. Toplam tamamlanma zamanı 2. Toplam bitiş zamanının en küçüklenmesi	Problemin çözümü için bir dal-sınır tekniği ve önerilen dal-sınır tekniğinin üst sınır değerlerini kullanan bir sezgisel (UB2) tanımlanmıştır. Önerilen dal-sınır algoritması, problemin esnekliğine göre birkaç alt sınır kullanır. Ağaç yapısının büyüklüğünü sınırlandırmak için birkaç baskınlık kuralı tanımlanmıştır. UB2, insert ve HRaj sezgiselleri ile karşılaştırılmıştır.	UB2, insert algoritmasından daha iyidir. Problem büyüklüğü arttıkça, UB2 daha etkili olmaktadır. HRaj en iyi sonucu vermektedir. Önerilen dal-sınır tekniklerinden BB2 daha üstündür.	1. Daha büyük boyutlu problemleri çözmek için güçlü alt sınırlar bulunabilir. 2. Birincil ve ikincil amaçlar yer değiştirilebilir. 3. Toplam geciken iş sayısı veya toplam erken ve pozitif gecikme cezaları ikincil kriter olarak kullanılabilir. 4. Çok-aşamalı veya paralel makine uygulamaları için benzer çalışmalar yapılabilir.

Tablo B.1 İki- ve Çok-Amaçlı Akış Tipi Çözelgeleme Çalışmaları (Devam)

Yazar(lar)	Dizayn	Problem Tipi	Amaç	Uygulama	Sonuç	Tartışma
Allahverdi, 2004	Sezgisel ve deneysel karşılaştırma	n -iş, m -makine 2-amaçlı akış tipi çözelgeleme problemi	1.Toplam tamamlanma zamanı 2.Maksimum pozitif gecikme zamanının en küçüklenmesi	Probleminin çözümü için bir sezgisel yaklaşım önerilmiştir. Literatürdeki mevcut sezgisellerle performansları karşılaştırılmıştır. Ayrıca, 3-makinelili durum için bir baskınlık ilişkisi ve alt sınır geliştirilmiştir.	Önerilen yaklaşım, diğer algoritmalarından daha iyi sonuç vermektedir. 3-makinelili durum için oluşturulan baskınlık ilişkisinin de etkin olduğu görülmektedir.	Hazırlık sürelerinin de gözönüne alındığı durum için çalışma yapılabilir.
Allahverdi ve Aldowaisan, 2004	Sezgisel algoritma ve deneysel karşılaştırma	n -iş, m -makine 2-amaçlı akış tipi çözelgeleme problemi	1.Toplam tamamlanma zamanı 2.Maksimum gecikme zamanı en küçüklenmesi	Probleminin çözümü için bir melez SA ve GA sezgiselleri önerilmiştir. Ayrıca, bir baskınlık ilişkisi ve dal-sınır algoritması geliştirilmiştir. Literatürdeki mevcut sezgisellerle performansları karşılaştırılmıştır.	Önerilen melez SA ve GA algoritmaları, klasik SA ve GA algoritmalarından daha iyi sonuç vermektedir. İki algoritmanın performansı benzerdir. Önerilen sezgisel tek kriter kadar iki kriter için de iyi sonuçlar vermektedir. Diğer sezgisellerden de daha iyi sonuç vermektedir. Ayrıca baskınlık ilişkisinin ve dal-sınır algoritmasının da etkin olduğu görülmektedir.	-

Ek C

Tablo C.1 İlk Çözüm için Algoritma Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1322	1322	1322	1322	1322	1322	1322	1322	1322	1322
20x5	ta002	1368	1373	1377	1371	1373	1371	1371	1377	1371	1368
20x5	ta003	1128	1114	1114	1114	1114	1114	1114	1114	1114	1122
20x5	ta004	1330	1336	1337	1336	1329	1329	1333	1329	1328	1337
20x5	ta005	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338
20x5	ta006	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224
20x5	ta007	1317	1317	1317	1317	1317	1317	1310	1322	1317	1317
20x5	ta008	1223	1234	1223	1232	1231	1223	1234	1223	1236	1226
20x5	ta009	1307	1298	1304	1298	1304	1302	1304	1308	1303	1307
20x5	ta010	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183
20x10	ta011	1661	1667	1659	1665	1656	1662	1669	1665	1664	1673
20x10	ta012	1802	1853	1862	1853	1851	1860	1840	1858	1840	1853
20x10	ta013	1544	1546	1544	1541	1534	1544	1544	1554	1547	1546
20x10	ta014	1466	1466	1467	1467	1467	1467	1478	1470	1476	1469
20x10	ta015	1565	1549	1559	1555	1539	1550	1554	1552	1550	1552
20x10	ta016	1450	1438	1450	1459	1448	1448	1444	1450	1449	1445
20x10	ta017	1633	1621	1614	1597	1623	1607	1629	1598	1619	1633
20x10	ta018	1582	1596	1603	1589	1606	1599	1592	1590	1593	1586
20x10	ta019	1786	1734	1765	1775	1786	1765	1753	1752	1779	1753
20x10	ta020	1736	1744	1734	1742	1745	1747	1738	1739	1723	1746
20x20	ta021	2460	2466	2475	2490	2503	2484	2456	2510	2484	2480
20x20	ta022	2185	2182	2188	2190	2192	2190	2193	2187	2184	2181
20x20	ta023	2502	2512	2477	2471	2456	2504	2480	2499	2503	2508
20x20	ta024	2296	2313	2291	2313	2319	2310	2302	2315	2305	2312
20x20	ta025	2393	2400	2403	2396	2399	2369	2404	2380	2384	2399
20x20	ta026	2337	2331	2333	2334	2322	2320	2336	2339	2305	2339
20x20	ta027	2381	2370	2385	2381	2379	2376	2386	2367	2369	2388
20x20	ta028	2305	2306	2283	2305	2306	2294	2298	2287	2309	2303
20x20	ta029	2375	2383	2394	2407	2387	2383	2375	2379	2394	2384
20x20	ta030	2324	2330	2330	2323	2335	2278	2325	2295	2338	2287

Tablo C.2 İlk Çözüm için Algoritma Performans Sonuçları

$n \times m$	Problem No	Δ_{enlyi}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}	İş Sıralaması
20x5	ta001	3.44	3.44	3.44	0.00	8,9,17,15,11,19,13,1,14,2,6,16,3,7,4,5,12,18,10,20
20x5	ta002	0.66	0.96	1.32	3.13	6,20,7,12,3,15,9,1,10,17,13,19,4,5,18,2,11,14,16,8
20x5	ta003	3.05	3.26	4.35	4.85	3,16,19,14,18,20,7,12,10,15,1,4,11,5,13,8,17,9,6,2
20x5	ta004	2.71	3.05	3.40	3.78	13,9,16,11,7,10,1,19,17,15,2,18,20,14,5,4,8,12,6,3
20x5	ta005	8.34	8.34	8.34	0.00	4,9,17,12,10,13,3,15,6,16,5,18,20,19,1,7,8,2,14,11
20x5	ta006	2.43	2.43	2.43	0.00	14,11,6,17,16,13,12,1,7,5,20,2,18,10,15,4,9,8,19,3
20x5	ta007	5.73	6.28	6.70	2.86	10,13,11,20,5,15,3,8,2,6,16,1,9,4,19,17,18,7,14,12
20x5	ta008	1.41	1.87	2.49	5.40	17,16,9,2,14,12,6,4,10,18,19,5,1,3,7,8,20,13,15,11
20x5	ta009	5.53	5.98	6.34	3.47	4,17,1,8,20,18,2,16,7,6,3,10,12,11,15,14,19,9,5,13
20x5	ta010	6.77	6.77	6.77	0.00	11,7,6,16,10,8,5,13,1,12,19,18,2,3,17,4,9,15,14,20
20x10	ta011	4.68	5.19	5.75	4.93	18,5,12,17,9,15,3,2,20,10,4,13,11,14,7,6,19,8,16,1
20x10	ta012	8.62	11.34	12.24	17.52	19,5,17,9,15,12,7,2,1,13,11,20,10,8,4,6,14,3,18,16
20x10	ta013	2.54	3.24	3.88	4.99	7,9,13,16,15,5,1,20,10,6,4,3,19,14,17,11,2,12,18,8
20x10	ta014	6.39	6.63	7.26	4.27	18,20,3,6,10,12,16,2,11,15,7,13,9,4,1,14,5,8,19,17
20x10	ta015	8.46	9.41	10.29	6.79	16,8,13,18,6,14,15,1,12,2,20,4,5,17,3,19,7,10,9,11
20x10	ta016	2.93	3.66	4.44	5.36	19,6,8,7,20,3,18,13,14,16,17,11,9,4,10,15,12,5,2,1
20x10	ta017	7.61	8.99	10.04	13.25	10,19,7,1,6,4,18,9,17,20,12,5,3,16,13,2,11,15,8,14
20x10	ta018	2.86	3.62	4.42	7.50	17,7,19,15,4,18,14,5,9,16,8,20,10,3,2,1,13,6,11,12
20x10	ta019	8.85	10.78	12.12	16.98	11,12,20,17,2,4,8,14,18,6,16,7,19,1,5,9,10,15,13,3
20x10	ta020	8.30	9.33	9.81	7.24	12,17,13,5,4,19,16,6,7,2,8,10,15,14,18,20,9,3,1,11
20x20	ta021	6.92	8.00	9.27	17.49	8,16,10,14,13,15,20,18,5,3,12,9,11,6,7,19,1,2,4,17
20x20	ta022	3.86	4.15	4.43	4.13	18,11,6,3,10,20,16,4,1,12,14,13,7,19,8,15,5,17,9,2
20x20	ta023	5.59	7.10	8.00	18.77	19,5,18,2,13,4,16,15,1,9,10,14,20,3,17,12,6,11,7,8
20x20	ta024	3.06	3.81	4.32	8.92	11,14,8,13,18,2,6,4,20,15,12,5,19,16,1,7,3,10,9,17
20x20	ta025	3.40	4.44	4.93	11.43	14,15,18,2,9,1,17,10,19,3,20,16,4,5,12,7,11,13,8,6
20x20	ta026	3.55	4.65	5.08	10.85	6,13,2,5,15,18,8,16,9,1,17,10,12,11,20,14,7,3,4,19
20x20	ta027	4.14	4.63	5.06	7.47	5,12,14,10,17,6,16,4,19,2,18,9,8,11,20,15,1,3,7,13
20x20	ta028	3.77	4.53	4.95	8.87	4,10,2,16,14,17,7,20,5,8,18,13,6,11,3,15,1,19,9,12
20x20	ta029	6.17	6.67	7.60	9.90	1,3,2,14,6,13,17,7,18,15,20,11,4,8,10,12,16,19,9,5
20x20	ta030	4.59	6.36	7.35	21.49	3,7,15,17,19,12,6,13,18,1,5,14,8,2,9,4,10,11,16,20
Ortalama=		4.88	5.63	6.23	7.72	

Ek D

Tablo D.1.1 α ve β Parametreleri Eğilim Belirleme Deneme Sonuçları

Problem No	Tekrar No					Tekrar No				
	1	2	3	4	5	1	2	3	4	5
	$\alpha=0$					$\beta=0$				
ta001	1325	1325	1325	1325	1325	1359	1385	1390	1381	1402
ta002	1429	1429	1429	1429	1429	1515	1481	1481	1489	1439
ta003	1162	1162	1162	1162	1162	1321	1246	1322	1246	1322
ta004	1421	1421	1421	1421	1421	1470	1512	1492	1470	1470
ta005	1360	1360	1360	1360	1360	1366	1366	1366	1366	1371
ta006	1305	1305	1305	1305	1305	1330	1342	1424	1414	1330
ta007	1360	1360	1360	1360	1360	1344	1280	1345	1364	1355
ta008	1285	1285	1285	1285	1285	1343	1354	1315	1365	1331
ta009	1403	1403	1403	1403	1403	1354	1405	1397	1405	1354
ta010	1194	1194	1194	1194	1194	1199	1323	1323	1323	1255
	$\alpha=0,5$					$\beta=0,5$				
ta001	1324	1324	1324	1324	1324	1297	1297	1297	1297	1297
ta002	1403	1403	1397	1403	1397	1368	1368	1377	1368	1368
ta003	1132	1128	1128	1128	1128	1120	1115	1114	1116	1116
ta004	1343	1350	1350	1345	1350	1359	1373	1364	1350	1342
ta005	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338
ta006	1234	1234	1224	1234	1224	1210	1210	1210	1210	1210
ta007	1360	1360	1360	1360	1360	1284	1284	1284	1271	1271
ta008	1237	1237	1239	1237	1239	1240	1223	1230	1223	1243
ta009	1350	1350	1307	1350	1350	1298	1309	1278	1298	1301
ta010	1183	1183	1183	1183	1183	1161	1161	1161	1161	1161
	$\alpha=1$					$\beta=1$				
ta001	1322	1322	1322	1322	1322	1297	1297	1297	1297	1297
ta002	1371	1371	1371	1369	1371	1371	1368	1368	1371	1371
ta003	1114	1127	1114	1127	1119	1114	1114	1114	1114	1114
ta004	1338	1330	1337	1337	1337	1331	1328	1332	1335	1329
ta005	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338
ta006	1234	1224	1224	1231	1224	1224	1224	1224	1224	1224

Tablo D.1.1 α ve β Parametreleri Eğilim Belirleme Deneme Sonuçları (Devam)

Problem No	Tekrar No					Tekrar No				
	1	2	3	4	5	1	2	3	4	5
ta007	1317	1314	1284	1309	1317	1284	1272	1271	1283	1271
ta008	1217	1231	1224	1220	1225	1218	1216	1222	1215	1220
ta009	1304	1316	1304	1313	1304	1298	1300	1306	1298	1304
ta010	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183
	$\alpha=2$					$\beta=2$				
ta001	1297	1297	1297	1297	1297	1322	1322	1322	1322	1322
ta002	1368	1368	1370	1368	1368	1371	1381	1377	1377	1373
ta003	1114	1114	1119	1114	1114	1114	1114	1114	1114	1127
ta004	1315	1329	1330	1338	1333	1337	1335	1340	1336	1343
ta005	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338
ta006	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224
ta007	1284	1284	1284	1284	1272	1317	1309	1317	1317	1317
ta008	1214	1212	1220	1223	1218	1225	1223	1233	1229	1234
ta009	1298	1281	1298	1298	1298	1306	1311	1304	1306	1304
ta010	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183
	$\alpha=5$					$\beta=5$				
ta001	1297	1297	1297	1297	1297	1324	1324	1324	1324	1324
ta002	1367	1372	1370	1370	1377	1403	1403	1403	1403	1403
ta003	1116	1120	1132	1121	1117	1132	1132	1132	1132	1132
ta004	1362	1333	1359	1357	1352	1350	1350	1350	1350	1350
ta005	1338	1338	1338	1338	1338	1360	1338	1338	1338	1338
ta006	1210	1210	1210	1220	1211	1288	1288	1288	1234	1262
ta007	1284	1284	1259	1282	1284	1360	1360	1360	1360	1360
ta008	1243	1243	1228	1254	1240	1239	1237	1239	1239	1239
ta009	1284	1298	1278	1281	1298	1350	1350	1350	1350	1350
ta010	1161	1161	1161	1161	1161	1183	1183	1183	1183	1183

Tablo D.1.2 pl , pg ve q_0 Parametreleri Eğilim Belirleme Deneme Sonuçları

Problem No	Tekrar No					Tekrar No					Tekrar No				
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
	$pl=0$					$pg=0$					$q_0=0$				
ta001	1325	1325	1324	1324	1324	1325	1325	1325	1325	1325	1322	1322	1322	1322	1322
ta002	1397	1400	1400	1400	1412	1429	1429	1429	1429	1429	1371	1383	1381	1381	1371
ta003	1162	1162	1162	1162	1162	1162	1162	1162	1162	1162	1114	1114	1114	1114	1114
ta004	1421	1421	1410	1458	1435	1421	1421	1421	1421	1421	1334	1330	1336	1337	1337
ta005	1338	1338	1338	1360	1338	1360	1360	1360	1360	1360	1338	1338	1338	1338	1338
ta006	1305	1263	1305	1301	1305	1305	1305	1305	1305	1305	1224	1224	1224	1224	1224
ta007	1326	1357	1360	1349	1344	1360	1360	1360	1360	1360	1317	1317	1317	1317	1317
ta008	1285	1290	1285	1285	1300	1285	1285	1285	1285	1285	1232	1223	1232	1224	1233
ta009	1378	1378	1358	1387	1392	1403	1403	1403	1403	1403	1304	1306	1304	1306	1302
ta010	1205	1194	1194	1194	1194	1194	1194	1194	1194	1194	1183	1183	1183	1183	1183
	$pl=0,1$					$pg=0,1$					$q_0=0,25$				
ta001	1322	1322	1322	1322	1322	1322	1322	1322	1322	1322	1322	1322	1322	1322	1322
ta002	1371	1383	1381	1381	1371	1371	1383	1381	1381	1371	1381	1371	1376	1390	1385
ta003	1114	1126	1114	1114	1125	1114	1126	1114	1114	1125	1119	1114	1123	1127	1114
ta004	1336	1336	1337	1328	1337	1336	1336	1337	1328	1337	1328	1334	1328	1335	1337
ta005	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338
ta006	1224	1224	1224	1224	1227	1224	1224	1224	1224	1227	1224	1224	1234	1224	1224
ta007	1317	1321	1309	1310	1309	1317	1321	1309	1310	1309	1292	1317	1317	1317	1299
ta008	1225	1223	1231	1223	1223	1225	1223	1231	1223	1223	1225	1220	1235	1235	1236
ta009	1315	1298	1337	1298	1302	1315	1298	1337	1298	1302	1304	1314	1307	1304	1307
ta010	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183
	$pl=0,2$					$pg=0,2$					$q_0=0,5$				
ta001	1324	1324	1324	1324	1322	1322	1297	1322	1322	1322	1322	1322	1322	1322	1322
ta002	1371	1371	1368	1371	1393	1368	1371	1368	1368	1371	1371	1381	1370	1370	1377
ta003	1132	1132	1132	1132	1132	1114	1132	1114	1119	1132	1114	1128	1114	1114	1114
ta004	1331	1338	1339	1343	1338	1334	1328	1334	1345	1323	1337	1334	1330	1337	1336
ta005	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338
ta006	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224

Tablo D.1.2 pl , pg ve q_0 Parametreleri Eğilim Belirleme Deneme Sonuçları (Devam)

Problem No	Tekrar No					Tekrar No					Tekrar No				
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
ta007	1317	1317	1317	1317	1317	1284	1284	1284	1284	1284	1317	1309	1317	1292	1317
ta008	1225	1242	1233	1244	1237	1222	1219	1225	1217	1220	1223	1228	1229	1223	1225
ta009	1322	1335	1335	1314	1304	1298	1298	1298	1298	1306	1298	1302	1304	1304	1303
ta010	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183
	$pl=0,3$					$pg=0,3$					$q_0=0,75$				
ta001	1324	1324	1324	1324	1324	1297	1297	1297	1297	1297	1322	1322	1322	1322	1322
ta002	1371	1371	1371	1371	1371	1368	1368	1368	1368	1373	1383	1383	1377	1381	1371
ta003	1136	1142	1136	1132	1132	1114	1114	1114	1132	1109	1119	1118	1127	1114	1114
ta004	1346	1346	1353	1336	1350	1334	1335	1341	1329	1334	1336	1333	1336	1337	1337
ta005	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338
ta006	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224
ta007	1322	1317	1317	1317	1317	1284	1284	1284	1284	1284	1317	1309	1292	1317	1292
ta008	1264	1233	1254	1264	1261	1223	1222	1216	1225	1212	1232	1216	1223	1228	1227
ta009	1335	1335	1335	1346	1335	1297	1298	1312	1284	1298	1307	1308	1304	1304	1313
ta010	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183
	$pl=0,5$					$pg=0,5$					$q_0=0,9$				
ta001	1324	1324	1324	1324	1324	1297	1297	1297	1297	1297	1322	1322	1322	1322	1322
ta002	1371	1371	1368	1368	1371	1367	1372	1368	1377	1372	1371	1373	1373	1377	1371
ta003	1144	1136	1143	1136	1136	1111	1104	1124	1132	1132	1114	1114	1114	1114	1114
ta004	1359	1354	1362	1350	1350	1360	1361	1364	1372	1346	1331	1328	1330	1336	1329
ta005	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338	1338
ta006	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224	1224
ta007	1322	1317	1317	1317	1322	1284	1284	1271	1284	1284	1317	1283	1317	1317	1317
ta008	1267	1273	1267	1273	1276	1219	1217	1219	1222	1223	1235	1232	1229	1235	1225
ta009	1335	1335	1350	1335	1335	1296	1303	1296	1298	1298	1304	1304	1298	1307	1321
ta010	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183

Tablo D.2.1 α Parametresi Eğilim Belirleme Bağlı Hata Sonuçları

Δ_{eniyi}	α				
Problem No	0	0.5	1	2	5
ta001	3.68	3.60	3.44	1.49	1.49
ta002	5.15	2.80	0.74	0.66	0.59
ta003	7.49	4.35	3.05	3.05	3.24
ta004	9.90	3.87	2.86	1.70	3.09
ta005	10.12	8.34	8.34	8.34	8.34
ta006	9.21	2.43	2.43	2.43	1.26
ta007	10.21	10.21	4.05	3.08	2.03
ta008	6.55	2.57	0.91	0.50	1.82
ta009	14.07	6.26	6.02	4.15	3.90
ta010	7.76	6.77	6.77	6.77	4.78
Ortalama=	8.41	5.12	3.86	3.22	3.05
Δ_{ort}	α				
Problem No	0	0,5	1	2	5
ta001	3.68	3.60	3.44	1.49	1.49
ta002	5.15	3.06	0.85	0.69	0.90
ta003	7.49	4.42	3.63	3.15	3.72
ta004	9.90	4.22	3.31	2.78	4.61
ta005	10.12	8.34	8.34	8.34	8.34
ta006	9.21	2.93	2.71	2.43	1.44
ta007	10.21	10.21	6.01	3.86	3.61
ta008	6.55	2.64	1.44	0.95	2.95
ta009	14.07	9.06	6.36	5.25	4.70
ta010	7.76	6.77	6.77	6.77	4.78
Ortalama=	8.41	5.52	4.29	3.57	3.65

Tablo D.2.2 β Parametresi Eğilim Belirleme Bağlı Hata Sonuçları

Δ_{eniyl}	β				
Problem No	0	0.5	1	2	5
ta001	6.338	1.487	1.487	3.443	3.599
ta002	5.887	0.662	0.662	0.883	3.238
ta003	15.264	3.053	3.053	3.053	4.718
ta004	13.689	3.790	2.707	3.248	4.408
ta005	10.607	8.340	8.340	8.340	8.340
ta006	11.297	1.255	2.427	2.427	3.264
ta007	3.728	2.998	2.998	6.078	10.211
ta008	9.038	1.410	0.746	1.410	2.570
ta009	10.081	3.902	5.528	6.016	9.756
ta010	8.213	4.783	6.769	6.769	6.769
Ortalama=	9.414	3.168	3.472	4.167	5.687
Δ_{ort}	β				
Problem No	0	0,5	1	2	5
ta001	8.247	1.487	1.487	3.443	3.599
ta002	8.977	0.795	0.795	1.236	3.238
ta003	19.463	3.256	3.053	3.293	4.718
ta004	14.679	4.996	2.939	3.496	4.408
ta005	10.688	8.340	8.340	8.340	8.696
ta006	14.477	1.255	2.427	2.427	6.444
ta007	8.395	3.630	3.420	6.596	10.211
ta008	11.244	2.139	1.012	1.891	2.703
ta009	12.439	5.431	5.789	6.195	9.756
ta010	15.939	4.783	6.769	6.769	6.769
Ortalama=	12.455	3.611	3.603	4.369	6.054

Tablo D.2.3 pl Parametresi Eğilim Belirleme Bağlı Hata Sonuçları

Δ_{eniyi}	pl				
Problem No	0	0.1	0.2	0.3	0.5
<i>ta001</i>	3.599	3.443	3.443	3.599	3.599
<i>ta002</i>	2.796	0.883	0.662	0.883	0.662
<i>ta003</i>	7.493	3.053	4.718	4.718	5.088
<i>ta004</i>	9.049	2.707	2.939	3.326	4.408
<i>ta005</i>	8.340	8.340	8.340	8.340	8.340
<i>ta006</i>	5.690	2.427	2.427	2.427	2.427
<i>ta007</i>	7.455	6.078	6.726	6.726	6.726
<i>ta008</i>	6.551	1.410	1.575	2.239	5.058
<i>ta009</i>	10.407	5.528	6.016	8.537	8.537
<i>ta010</i>	7.762	6.769	6.769	6.769	6.769
Ortalama=	6.914	4.064	4.362	4.756	5.161
Δ_{ort}	pl				
Problem No	0	0,1	0,2	0,3	0,5
<i>ta001</i>	3.631	3.443	3.568	3.599	3.599
<i>ta002</i>	3.149	1.354	1.163	0.883	0.795
<i>ta003</i>	7.493	3.478	4.718	5.051	5.365
<i>ta004</i>	10.518	3.233	3.465	4.114	4.795
<i>ta005</i>	8.696	8.340	8.340	8.340	8.340
<i>ta006</i>	8.435	2.477	2.427	2.427	2.427
<i>ta007</i>	9.173	6.418	6.726	6.807	6.888
<i>ta008</i>	6.882	1.575	2.504	4.080	5.406
<i>ta009</i>	12.081	6.504	7.480	8.715	8.780
<i>ta010</i>	7.960	6.769	6.769	6.769	6.769
Ortalama=	7.802	4.359	4.716	5.079	5.317

Tablo D.2.4 pg Parametresi Eğilim Belirleme Bağlı Hata Sonuçları

Δ_{eniyi}	pg				
Problem No	0	0.1	0.2	0.3	0.5
$ta001$	3.678	3.443	1.487	1.487	1.487
$ta002$	5.151	0.883	0.662	0.662	0.589
$ta003$	7.493	3.053	3.053	2.590	2.128
$ta004$	9.899	2.707	2.320	2.784	4.099
$ta005$	10.121	8.340	8.340	8.340	8.340
$ta006$	9.205	2.427	2.427	2.427	2.427
$ta007$	10.211	6.078	4.052	4.052	2.998
$ta008$	6.551	1.410	0.912	0.498	0.912
$ta009$	14.065	5.528	5.528	4.390	5.366
$ta010$	7.762	6.769	6.769	6.769	6.769
Ortalama=	8.414	4.064	3.555	3.400	3.511
Δ_{ort}	pg				
Problem No	0	0,1	0,2	0,3	0,5
$ta001$	3.678	3.443	3.052	1.487	1.487
$ta002$	5.151	1.354	0.751	0.736	0.898
$ta003$	7.493	3.478	3.811	3.293	3.663
$ta004$	9.899	3.233	3.078	3.217	5.228
$ta005$	10.121	8.340	8.340	8.340	8.340
$ta006$	9.205	2.477	2.427	2.427	2.427
$ta007$	10.211	6.418	4.052	4.052	3.841
$ta008$	6.551	1.575	1.211	1.128	1.161
$ta009$	14.065	6.504	5.659	5.512	5.545
$ta010$	7.762	6.769	6.769	6.769	6.769
Ortalama=	8.414	4.359	3.915	3.696	3.936

Tablo D.2.5 q_0 Parametresi Eğilim Belirleme Bağlı Hata Sonuçları

Δ_{eni}	q_0				
Problem No	0	0.25	0.5	0.75	0.9
<i>ta001</i>	3.443	3.443	3.443	3.443	3.443
<i>ta002</i>	0.883	0.883	0.809	0.883	0.883
<i>ta003</i>	3.053	3.053	3.053	3.053	3.053
<i>ta004</i>	2.862	2.707	2.862	3.094	2.707
<i>ta005</i>	8.340	8.340	8.340	8.340	8.340
<i>ta006</i>	2.427	2.427	2.427	2.427	2.427
<i>ta007</i>	6.726	4.700	4.700	4.700	3.971
<i>ta008</i>	1.410	1.161	1.410	0.829	1.575
<i>ta009</i>	5.854	6.016	5.528	6.016	5.528
<i>ta010</i>	6.769	6.769	6.769	6.769	6.769
Ortalama=	<i>4.177</i>	<i>3.950</i>	<i>3.934</i>	<i>3.955</i>	<i>3.870</i>
Δ_{ort}	q_0				
Problem No	0	0,25	0,5	0,75	0,9
<i>ta001</i>	3.443	3.443	3.443	3.443	3.443
<i>ta002</i>	1.354	1.589	1.089	1.472	1.030
<i>ta003</i>	3.053	3.552	3.312	3.460	3.053
<i>ta004</i>	3.233	3.047	3.233	3.310	2.923
<i>ta005</i>	8.340	8.340	8.340	8.340	8.340
<i>ta006</i>	2.427	2.594	2.427	2.427	2.427
<i>ta007</i>	6.726	6.029	6.191	5.786	6.175
<i>ta008</i>	1.891	2.007	1.625	1.592	2.090
<i>ta009</i>	6.049	6.276	5.870	6.276	6.244
<i>ta010</i>	6.769	6.769	6.769	6.769	6.769
Ortalama=	<i>4.328</i>	<i>4.365</i>	<i>4.230</i>	<i>4.287</i>	<i>4.249</i>

Tablo D.3.1 α Parametresi Değerini Belirleme Deneme Sonuçları

Problem No	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
	$\alpha=0$									
ta011	1738	1738	1738	1738	1738	1738	1738	1738	1738	1738
ta012	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
ta013	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
ta014	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
	$\alpha=0,5$									
ta011	1661	1654	1662	1649	1660	1673	1675	1663	1646	1665
ta012	1860	1845	1828	1853	1846	1852	1805	1864	1859	1803
ta013	1554	1548	1544	1539	1552	1539	1544	1549	1534	1548
ta014	1478	1466	1467	1472	1467	1476	1469	1482	1472	1467
	$\alpha=1$									
ta011	1643	1644	1646	1634	1640	1646	1655	1639	1634	1636
ta012	1777	1785	1771	1765	1771	1771	1789	1785	1782	1787
ta013	1555	1554	1534	1551	1552	1547	1556	1546	1554	1547
ta014	1467	1466	1466	1467	1478	1466	1461	1472	1467	1456
	$\alpha=2$									
ta011	1660	1634	1651	1660	1656	1648	1652	1634	1657	1660
ta012	1782	1776	1779	1758	1785	1753	1777	1771	1776	1780
ta013	1552	1554	1559	1562	1575	1560	1563	1559	1545	1553
ta014	1479	1472	1469	1449	1468	1469	1465	1448	1494	1479
	$\alpha=5$									
ta011	1650	1668	1686	1693	1671	1677	1675	1668	1699	1704
ta012	1777	1781	1763	1765	1760	1763	1755	1757	1721	1757
ta013	1596	1635	1584	1640	1632	1544	1636	1614	1627	1620
ta014	1452	1433	1444	1441	1449	1442	1456	1460	1455	1459

Tablo D.3.2 β Parametresi Değerini Belirleme Deneme Sonuçları

Problem No	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
	$\beta=0$									
ta011	1879	1816	1812	1801	1816	1791	1879	1818	1844	1780
ta012	1921	1867	1884	1901	1889	1960	1865	1922	1904	1909
ta013	1682	1757	1654	1700	1665	1709	1674	1667	1734	1660
ta014	1650	1585	1632	1632	1626	1637	1593	1626	1599	1650
	$\beta=0,5$									
ta011	1647	1646	1640	1630	1667	1654	1657	1641	1648	1657
ta012	1780	1767	1772	1763	1778	1769	1768	1758	1773	1781
ta013	1594	1541	1575	1549	1559	1548	1601	1554	1540	1557
ta014	1476	1473	1445	1473	1465	1467	1477	1469	1475	1448
	$\beta=1$									
ta011	1643	1644	1646	1634	1640	1646	1655	1639	1634	1636
ta012	1777	1785	1771	1765	1771	1771	1789	1785	1782	1787
ta013	1555	1554	1534	1551	1552	1547	1556	1546	1554	1547
ta014	1467	1466	1466	1467	1478	1466	1461	1472	1467	1456
	$\beta=2$									
ta011	1664	1651	1662	1652	1665	1661	1662	1667	1640	1656
ta012	1803	1869	1859	1798	1804	1848	1838	1848	1856	1801
ta013	1534	1539	1539	1544	1544	1539	1537	1551	1548	1544
ta014	1467	1474	1467	1467	1467	1461	1482	1467	1479	1467
	$\beta=5$									
ta011	1689	1694	1686	1696	1692	1697	1689	1688	1689	1689
ta012	1878	1889	1888	1897	1877	1881	1881	1881	1889	1881
ta013	1573	1576	1568	1573	1570	1577	1577	1558	1564	1579
ta014	1489	1474	1474	1482	1474	1477	1474	1487	1474	1474

Tablo D.3.3 pl Parametresi Değerini Belirleme Deneme Sonuçları

Problem No	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
	$pl=0$									
<i>ta011</i>	1720	1710	1720	1710	1720	1720	1753	1720	1720	1710
<i>ta012</i>	1881	1930	1897	1911	1931	1911	1881	1881	1930	1881
<i>ta013</i>	1704	1689	1695	1677	1677	1677	1695	1695	1695	1695
<i>ta014</i>	1585	1585	1531	1526	1585	1585	1531	1582	1582	1585
	$pl=0,1$									
<i>ta011</i>	1648	1641	1638	1638	1645	1641	1642	1646	1634	1648
<i>ta012</i>	1772	1775	1781	1777	1787	1768	1778	1784	1774	1766
<i>ta013</i>	1554	1552	1537	1544	1552	1556	1547	1550	1544	1546
<i>ta014</i>	1455	1466	1467	1466	1467	1470	1470	1472	1472	1470
	$pl=0,2$									
<i>ta011</i>	1639	1648	1657	1657	1639	1645	1648	1648	1648	1656
<i>ta012</i>	1782	1793	1794	1786	1782	1777	1791	1800	1797	1789
<i>ta013</i>	1548	1544	1545	1544	1556	1556	1544	1544	1550	1556
<i>ta014</i>	1473	1467	1468	1464	1469	1472	1482	1466	1470	1471
	$pl=0,3$									
<i>ta011</i>	1669	1663	1673	1673	1673	1673	1665	1656	1648	1673
<i>ta012</i>	1791	1807	1807	1797	1807	1800	1807	1783	1805	1798
<i>ta013</i>	1548	1553	1554	1565	1548	1554	1544	1558	1561	1565
<i>ta014</i>	1480	1478	1476	1481	1474	1475	1470	1472	1474	1477
	$pl=0,5$									
<i>ta011</i>	1673	1673	1673	1673	1673	1673	1673	1673	1673	1673
<i>ta012</i>	1807	1807	1807	1811	1807	1818	1798	1805	1798	1800
<i>ta013</i>	1557	1557	1566	1575	1572	1574	1572	1569	1559	1570
<i>ta014</i>	1489	1478	1478	1491	1489	1478	1489	1486	1478	1478

Tablo D.3.4 pg Parametresi Değerini Belirleme Deneme Sonuçları

Problem No	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
	$pg=0$									
ta011	1738	1738	1738	1738	1738	1738	1738	1738	1738	1738
ta012	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
ta013	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
ta014	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
	$pg=0,1$									
ta011	1648	1641	1638	1638	1645	1641	1642	1646	1634	1648
ta012	1772	1775	1781	1777	1787	1768	1778	1784	1774	1766
ta013	1554	1552	1537	1544	1552	1556	1547	1550	1544	1546
ta014	1455	1466	1467	1466	1467	1470	1470	1472	1472	1470
	$pg=0,2$									
ta011	1652	1665	1645	1652	1634	1642	1657	1646	1648	1638
ta012	1774	1778	1784	1779	1773	1776	1737	1794	1775	1770
ta013	1576	1576	1558	1537	1557	1568	1564	1555	1561	1548
ta014	1465	1452	1473	1478	1477	1472	1458	1481	1467	1470
	$pg=0,3$									
ta011	1646	1643	1668	1656	1634	1652	1646	1653	1648	1638
ta012	1765	1783	1775	1784	1770	1781	1782	1778	1781	1767
ta013	1558	1580	1576	1574	1574	1574	1598	1575	1569	1569
ta014	1496	1489	1486	1463	1459	1453	1465	1471	1452	1471
	$pg=0,5$									
ta011	1652	1666	1670	1664	1647	1680	1680	1668	1704	1658
ta012	1780	1786	1778	1784	1776	1771	1800	1763	1762	1778
ta013	1608	1573	1590	1607	1608	1593	1637	1569	1595	1592
ta014	1464	1461	1472	1489	1468	1479	1483	1467	1492	1488

Tablo D.3.5 q_0 Parametresi Değerini Belirleme Deneme Sonuçları

Problem No	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
	$q_0=0$									
ta011	1634	1641	1620	1656	1641	1651	1640	1640	1646	1642
ta012	1784	1775	1786	1771	1784	1783	1779	1779	1778	1777
ta013	1550	1555	1554	1548	1552	1553	1555	1537	1540	1557
ta014	1475	1467	1467	1469	1466	1485	1472	1466	1467	1467
	$q_0=0,25$									
ta011	1634	1649	1638	1634	1643	1628	1634	1634	1653	1609
ta012	1765	1783	1783	1785	1774	1760	1770	1774	1774	1777
ta013	1555	1546	1555	1543	1549	1552	1555	1554	1544	1563
ta014	1473	1467	1467	1467	1456	1468	1489	1455	1467	1468
	$q_0=0,5$									
ta011	1632	1632	1647	1636	1635	1634	1629	1648	1647	1652
ta012	1782	1776	1779	1782	1780	1783	1777	1777	1780	1787
ta013	1551	1556	1537	1544	1538	1544	1545	1550	1544	1539
ta014	1472	1477	1470	1471	1473	1467	1472	1469	1466	1470
	$q_0=0,75$									
ta011	1658	1640	1646	1646	1634	1625	1632	1648	1646	1651
ta012	1775	1780	1789	1770	1764	1781	1787	1780	1786	1781
ta013	1556	1549	1537	1544	1554	1545	1544	1543	1547	1544
ta014	1472	1469	1470	1467	1472	1479	1469	1469	1467	1469
	$q_0=0,9$									
ta011	1648	1641	1638	1638	1645	1641	1642	1646	1634	1648
ta012	1772	1775	1781	1777	1787	1768	1778	1784	1774	1766
ta013	1554	1552	1537	1544	1552	1556	1547	1550	1544	1546
ta014	1455	1466	1467	1466	1467	1470	1470	1472	1472	1470

Tablo D.4.1 α Parametresi Değerini Belirleme Bağlı Hata Sonuçları

Δ_{eniyi}	α				
Problem No	0	0.5	1	2	5
<i>ta011</i>	9.86	4.05	3.29	3.29	4.30
<i>ta012</i>	15.91	8.68	6.39	5.67	3.74
<i>ta013</i>	13.30	2.54	2.54	3.28	3.21
<i>ta014</i>	15.02	6.39	5.66	5.08	3.99
Ortalama=	<i>13.52</i>	<i>5.41</i>	<i>4.47</i>	<i>4.33</i>	<i>3.81</i>
Δ_{ort}	α				
Problem No	0	0.5	1	2	5
<i>ta011</i>	9.86	4.98	3.77	4.37	6.14
<i>ta012</i>	15.91	11.00	7.19	6.91	6.08
<i>ta013</i>	13.30	3.28	3.58	4.16	7.81
<i>ta014</i>	15.02	6.79	6.43	6.62	5.16
Ortalama=	<i>13.52</i>	<i>6.51</i>	<i>5.24</i>	<i>5.52</i>	<i>6.30</i>

Tablo D.4.2 β Parametresi Değerini Belirleme Bağlı Hata Sonuçları

Δ_{eniyi}	β				
Problem No	0	0.5	1	2	5
<i>ta011</i>	12.52	3.03	3.29	3.67	6.57
<i>ta012</i>	12.42	5.97	6.39	8.38	13.14
<i>ta013</i>	10.56	2.94	2.54	2.54	4.14
<i>ta014</i>	15.02	4.86	5.66	6.02	6.97
Ortalama=	<i>12.63</i>	<i>4.20</i>	<i>4.47</i>	<i>5.15</i>	<i>7.71</i>
Δ_{ort}	β				
Problem No	0	0.5	1	2	5
<i>ta011</i>	15.27	4.22	3.77	4.80	6.88
<i>ta012</i>	14.66	6.75	7.19	10.45	13.57
<i>ta013</i>	12.98	4.40	3.58	3.07	5.05
<i>ta014</i>	17.78	6.44	6.43	6.66	7.25
Ortalama=	<i>15.17</i>	<i>5.45</i>	<i>5.24</i>	<i>6.25</i>	<i>8.19</i>

Tablo D.4.3 pl Parametresi Değerini Belirleme Bağlı Hata Sonuçları

Δ_{eniyi}	pl				
Problem No	0	0.1	0.2	0.3	0.5
$ta011$	8.09	3.29	3.60	5.35	5.75
$ta012$	13.38	6.45	7.11	8.51	8.85
$ta013$	12.10	2.74	3.21	3.94	4.75
$ta014$	10.74	5.59	6.24	7.09	7.65
Ortalama=	11.19	4.16	4.64	5.93	6.45
Δ_{ort}	pl				
Problem No	0	0.1	0.2	0.3	0.5
$ta011$	8.74	3.80	4.20	5.35	5.75
$ta012$	14.73	7.06	7.84	8.51	8.85
$ta013$	12.96	3.49	3.52	3.94	4.75
$ta014$	13.77	6.49	6.69	7.09	7.65
Ortalama=	12.15	4.78	5.19	5.93	6.45

Tablo D.4.4 pg Parametresi Değerini Belirleme Bağlı Hata Sonuçları

Δ_{eniyi}	pg				
Problem No	0	0.1	0.2	0.3	0.5
$ta011$	9.86	3.29	3.29	3.29	4.11
$ta012$	15.91	6.45	4.70	6.39	6.21
$ta013$	13.30	2.74	2.74	4.14	4.88
$ta014$	15.02	5.59	5.37	5.37	6.02
Ortalama=	13.03	4.16	3.58	4.61	5.07
Δ_{ort}	pg				
Problem No	0	0.1	0.2	0.3	0.5
$ta011$	9.86	3.80	4.17	4.20	5.49
$ta012$	15.91	7.06	6.93	7.09	7.16
$ta013$	13.30	3.49	4.28	5.26	6.76
$ta014$	15.02	6.49	6.63	6.71	7.13
Ortalama=	13.03	4.78	5.13	5.52	6.47

Tablo D.4.5 q_0 Parametresi Değerini Belirleme Bağlı Hata Sonuçları

Δ_{eniyi}	q_0				
Problem No	0	0.25	0.5	0.75	0.9
<i>ta011</i>	2.40	1.71	2.97	2.72	3.29
<i>ta012</i>	6.75	6.09	7.05	6.33	6.45
<i>ta013</i>	2.74	3.14	2.74	2.74	2.74
<i>ta014</i>	6.39	5.59	6.39	6.46	5.59
Ortalama=	3.96	3.65	4.25	3.93	4.16
Δ_{ort}	q_0				
Problem No	0	0.25	0.5	0.75	0.9
<i>ta011</i>	3.74	3.39	3.62	3.83	3.80
<i>ta012</i>	7.27	6.96	7.31	7.25	7.06
<i>ta013</i>	3.62	3.72	3.26	3.36	3.49
<i>ta014</i>	6.68	6.51	6.73	6.70	6.49
Ortalama=	4.87	4.69	4.73	4.81	4.78

Tablo D.5 Taguchi Analizi Deneme Sonuçları

Deneme No	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
Problem No=	ta011									
1	1640	1657	1646	1652	1621	1661	1646	1634	1654	1660
2	1665	1677	1678	1638	1651	1667	1656	1681	1673	1669
3	1657	1646	1657	1657	1649	1648	1647	1657	1648	1652
4	1640	1657	1648	1654	1656	1647	1652	1631	1649	1652
5	1648	1650	1647	1648	1645	1634	1645	1648	1648	1648
6	1646	1636	1648	1645	1648	1647	1635	1646	1632	1634
7	1649	1648	1657	1645	1648	1646	1636	1658	1647	1631
8	1696	1650	1661	1667	1673	1651	1667	1667	1706	1653
Problem No=	ta012									
1	1768	1767	1770	1760	1779	1748	1765	1775	1775	1756
2	1780	1770	1775	1776	1774	1769	1775	1784	1777	1761
3	1783	1804	1798	1792	1799	1798	1807	1805	1795	1793
4	1786	1791	1767	1790	1779	1773	1771	1783	1780	1794
5	1780	1771	1782	1786	1774	1754	1772	1777	1786	1785
6	1747	1773	1767	1781	1763	1754	1757	1743	1774	1773
7	1768	1767	1766	1772	1763	1767	1771	1780	1775	1767
8	1780	1777	1777	1780	1769	1770	1799	1785	1782	1780
Problem No=	ta013									
1	1537	1556	1559	1585	1572	1564	1573	1554	1557	1566
2	1593	1628	1598	1620	1586	1603	1593	1610	1605	1611
3	1552	1556	1557	1552	1551	1548	1539	1547	1558	1539
4	1547	1544	1551	1554	1554	1544	1554	1554	1540	1544
5	1555	1539	1544	1547	1553	1539	1544	1544	1544	1544
6	1544	1537	1546	1542	1550	1544	1547	1552	1552	1543
7	1555	1619	1600	1558	1575	1570	1562	1559	1578	1561
8	1588	1614	1636	1607	1586	1620	1620	1644	1591	1611

Tablo D.5 Taguchi Analizi Deneme Sonuçları (Devam)

Deneme No	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
Problem No=	ta014									
1	1487	1467	1473	1459	1455	1474	1465	1457	1475	1467
2	1469	1441	1456	1446	1459	1467	1445	1481	1478	1447
3	1472	1472	1473	1470	1472	1472	1467	1467	1472	1470
4	1466	1472	1466	1466	1472	1473	1467	1474	1467	1461
5	1452	1457	1446	1460	1464	1462	1454	1456	1465	1462
6	1452	1435	1450	1440	1438	1453	1440	1440	1445	1437
7	1481	1483	1451	1456	1475	1468	1463	1462	1441	1446
8	1440	1461	1471	1448	1468	1458	1484	1465	1465	1465
Problem No=	ta015									
1	1493	1512	1535	1531	1521	1509	1522	1532	1524	1524
2	1529	1517	1564	1531	1516	1500	1545	1518	1517	1548
3	1553	1503	1534	1548	1565	1544	1555	1554	1564	1558
4	1550	1538	1547	1544	1544	1559	1528	1526	1545	1545
5	1530	1540	1535	1524	1542	1538	1524	1520	1550	1544
6	1531	1530	1543	1524	1522	1526	1521	1542	1534	1521
7	1515	1529	1535	1540	1519	1532	1509	1520	1524	1500
8	1541	1543	1541	1522	1537	1529	1553	1523	1572	1528
Problem No=	ta016									
1	1444	1448	1453	1444	1442	1450	1450	1458	1442	1436
2	1467	1455	1469	1446	1451	1475	1464	1448	1443	1433
3	1454	1450	1450	1458	1449	1449	1446	1457	1450	1450
4	1450	1450	1447	1444	1448	1448	1449	1447	1433	1452
5	1440	1444	1444	1444	1446	1438	1445	1451	1447	1446
6	1447	1437	1444	1447	1444	1454	1428	1438	1435	1443
7	1446	1444	1444	1427	1440	1449	1457	1438	1446	1452
8	1466	1462	1448	1466	1485	1479	1456	1441	1461	1450

Tablo D.5 Taguchi Analizi Deneme Sonuçları (Devam)

Deneme No	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
Problem No=	<i>ta017</i>									
1	1581	1564	1558	1584	1558	1582	1582	1575	1577	1582
2	1582	1553	1564	1560	1568	1563	1564	1573	1561	1554
3	1592	1590	1592	1580	1593	1587	1581	1592	1593	1592
4	1592	1586	1578	1576	1587	1579	1588	1588	1568	1588
5	1576	1531	1572	1566	1575	1571	1574	1576	1565	1535
6	1548	1539	1531	1537	1548	1548	1530	1549	1531	1531
7	1564	1582	1570	1583	1572	1587	1576	1575	1581	1576
8	1567	1567	1566	1569	1582	1558	1546	1565	1584	1584
Problem No=	<i>ta018</i>									
1	1609	1588	1601	1602	1597	1602	1593	1610	1601	1601
2	1667	1678	1647	1678	1659	1660	1661	1677	1675	1678
3	1601	1588	1598	1604	1600	1594	1593	1604	1599	1598
4	1590	1607	1582	1595	1589	1586	1586	1582	1598	1591
5	1601	1599	1588	1588	1571	1601	1582	1586	1582	1585
6	1586	1583	1588	1586	1590	1584	1582	1580	1593	1577
7	1605	1601	1602	1582	1578	1593	1598	1580	1595	1594
8	1661	1656	1696	1682	1671	1677	1678	1663	1644	1677
Problem No=	<i>ta019</i>									
1	1710	1710	1712	1713	1710	1703	1709	1717	1708	1708
2	1724	1714	1717	1714	1701	1704	1713	1716	1707	1716
3	1739	1740	1734	1741	1736	1736	1743	1741	1739	1736
4	1734	1716	1734	1731	1708	1721	1734	1734	1733	1729
5	1714	1708	1710	1711	1693	1696	1716	1729	1710	1699
6	1675	1706	1692	1701	1696	1699	1702	1694	1689	1688
7	1703	1708	1718	1715	1714	1710	1715	1715	1695	1714
8	1709	1703	1696	1722	1712	1707	1714	1718	1727	1712

Tablo D.5 Taguchi Analizi Deneme Sonuçları (Devam)

Deneme No	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
Problem No=	ta020									
1	1661	1672	1672	1672	1681	1676	1662	1658	1668	1673
2	1646	1671	1669	1675	1663	1667	1661	1675	1667	1675
3	1703	1692	1675	1702	1692	1698	1687	1699	1696	1698
4	1673	1682	1675	1686	1679	1688	1672	1678	1677	1703
5	1690	1681	1683	1680	1677	1681	1681	1687	1671	1669
6	1675	1677	1679	1686	1671	1688	1667	1678	1681	1684
7	1673	1681	1668	1676	1672	1679	1666	1663	1683	1685
8	1661	1675	1674	1681	1663	1660	1670	1659	1674	1665

Tablo D.6.1 Parmetre Analizi Sonrası Algoritma Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1297	1297	1297	1297	1297	1297	1297	1297	1297	1297
20x5	ta002	1368	1368	1368	1368	1368	1368	1371	1368	1368	1368
20x5	ta003	1127	1118	1119	1115	1116	1114	1119	1116	1114	1114
20x5	ta004	1336	1348	1329	1334	1322	1328	1341	1334	1337	1341
20x5	ta005	1280	1277	1292	1277	1276	1305	1271	1310	1256	1267
20x5	ta006	1210	1210	1219	1214	1220	1219	1216	1219	1219	1219
20x5	ta007	1259	1259	1251	1266	1251	1273	1269	1259	1259	1259
20x5	ta008	1220	1216	1215	1218	1219	1207	1218	1220	1221	1219
20x5	ta009	1298	1296	1297	1297	1297	1294	1290	1294	1290	1276
20x5	ta010	1161	1160	1155	1161	1141	1140	1161	1161	1161	1151
20x10	ta011	1646	1636	1648	1645	1648	1647	1635	1646	1630	1634
20x10	ta012	1747	1735	1767	1781	1763	1754	1757	1743	1774	1773
20x10	ta013	1544	1537	1546	1542	1550	1544	1547	1552	1552	1543
20x10	ta014	1452	1435	1450	1440	1438	1453	1428	1440	1445	1437
20x10	ta015	1531	1530	1543	1524	1522	1526	1521	1542	1534	1516
20x10	ta016	1447	1437	1444	1447	1444	1454	1435	1438	1435	1443
20x10	ta017	1548	1539	1531	1537	1548	1548	1531	1549	1531	1531
20x10	ta018	1586	1583	1588	1586	1590	1584	1582	1580	1593	1570
20x10	ta019	1675	1706	1692	1701	1696	1699	1702	1694	1689	1688
20x10	ta020	1675	1677	1679	1686	1671	1688	1669	1678	1681	1684
20x20	ta021	2462	2457	2433	2448	2429	2451	2449	2454	2449	2453
20x20	ta022	2172	2167	2203	2179	2142	2187	2162	2188	2175	2185
20x20	ta023	2422	2413	2433	2441	2416	2421	2422	2400	2427	2446
20x20	ta024	2309	2303	2306	2286	2318	2316	2305	2311	2287	2291
20x20	ta025	2390	2395	2408	2392	2359	2379	2400	2384	2400	2383
20x20	ta026	2317	2320	2315	2304	2290	2311	2306	2300	2323	2317
20x20	ta027	2363	2364	2362	2364	2366	2360	2362	2355	2367	2354
20x20	ta028	2285	2293	2287	2278	2287	2283	2279	2301	2279	2290
20x20	ta029	2378	2368	2349	2372	2379	2384	2368	2367	2384	2365
20x20	ta030	2288	2276	2308	2287	2287	2274	2280	2267	2271	2266

Tablo D.6.2 Parametre Analizi Sonrası Algoritma Performans Sonuçları

$n \times m$	Problem No	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}	İş Sıralaması
20x5	ta001	1.487	1.487	1.487	0.000	6,9,8,11,19,4,16,3,14,17,15,1,5,7,10,18,12,2,13,20
20x5	ta002	0.662	0.684	0.883	0.949	6,20,7,15,9,1,12,3,10,13,17,19,5,11,14,2,4,18,16,8
20x5	ta003	3.053	3.349	4.255	3.967	3,16,19,14,20,18,12,4,1,7,15,10,11,5,8,9,13,17,6,2
20x5	ta004	2.243	3.248	4.254	7.468	13,9,16,11,17,15,2,20,7,19,10,1,5,8,14,4,12,6,3,18
20x5	ta005	1.700	3.733	6.073	16.723	12,10,13,9,17,19,3,4,15,6,16,7,2,14,11,1,5,18,8,20
20x5	ta006	1.255	1.799	2.092	3.866	14,11,6,16,17,13,5,7,20,12,4,18,15,10,8,9,1,19,2,3
20x5	ta007	0.969	1.735	2.744	7.075	5,11,20,8,15,2,3,1,9,4,6,19,17,18,7,14,12,16,13,10
20x5	ta008	0.083	0.937	1.244	4.057	17,16,9,2,14,5,19,12,18,10,8,3,1,20,7,13,4,15,6,11
20x5	ta009	3.740	5.114	5.528	6.590	4,2,10,12,17,18,1,3,8,20,7,6,11,15,14,19,9,5,13,16
20x5	ta010	2.888	4.260	4.783	8.443	11,7,6,10,16,19,12,8,1,2,3,18,13,5,9,15,14,20,17,4
20x10	ta011	3.034	3.761	4.172	6.900	5,12,15,17,20,3,4,2,9,10,11,14,7,6,19,8,13,18,16,1
20x10	ta012	4.581	6.052	7.354	14.864	17,12,15,5,19,6,13,9,11,20,2,7,1,4,10,8,3,16,14,18
20x10	ta013	2.741	3.322	3.743	4.739	4,3,9,13,5,1,16,15,10,20,6,7,17,11,2,12,19,14,18,8
20x10	ta014	3.628	4.630	5.443	8.080	18,20,3,12,16,6,10,2,11,7,1,15,9,13,4,8,14,5,17,19
20x10	ta015	6.836	7.745	8.739	8.888	16,8,13,14,18,4,6,15,1,2,20,12,17,3,19,7,5,10,9,11
20x10	ta016	2.720	3.250	4.080	6.150	19,8,18,20,3,9,6,13,7,4,17,16,11,14,10,15,2,12,5,1
20x10	ta017	3.167	3.726	4.380	8.179	19,6,4,7,1,9,20,12,17,3,5,13,16,18,11,2,10,14,8,15
20x10	ta018	2.081	3.004	3.576	6.303	8,20,7,4,18,14,9,17,3,5,10,16,19,15,2,1,13,6,11,12
20x10	ta019	5.148	6.353	7.094	8.892	8,17,4,2,20,14,16,11,18,7,19,1,3,5,9,15,10,12,13,6
20x10	ta020	4.903	5.519	6.097	6.179	5,4,13,7,16,8,2,6,14,18,10,20,17,15,1,19,12,9,3,11
20x20	ta021	5.747	6.596	7.183	10.179	7,16,14,13,8,19,3,9,11,20,5,12,10,15,18,6,4,2,17,1
20x20	ta022	2.000	3.619	4.905	16.780	18,11,20,3,10,13,4,16,1,14,15,6,12,7,8,19,5,17,9,2
20x20	ta023	3.181	4.218	5.159	13.503	8,5,19,13,4,16,15,1,9,18,10,14,17,3,12,20,11,6,2,7
20x20	ta024	2.834	3.608	4.274	11.526	14,18,3,13,8,12,6,4,20,15,1,7,5,19,2,11,16,10,9,17
20x20	ta025	2.968	4.278	5.107	13.784	9,2,19,14,1,17,10,3,20,16,4,12,5,7,13,18,15,11,8,6
20x20	ta026	2.875	3.787	4.358	10.242	6,13,2,5,15,18,1,8,16,7,9,17,10,12,11,20,3,4,14,19
20x20	ta027	3.564	3.902	4.136	4.296	10,6,12,14,5,18,16,11,20,15,4,19,9,2,17,8,1,7,3,13
20x20	ta028	3.545	3.918	4.591	7.177	4,10,2,20,5,16,7,13,14,17,6,11,3,15,19,8,18,9,1,12
20x20	ta029	5.007	6.008	6.571	10.564	1,8,3,2,14,6,7,13,17,18,15,16,10,12,4,19,9,11,20,5
20x20	ta030	4.040	4.702	5.969	12.660	3,7,15,17,19,12,6,13,18,9,10,4,1,8,2,5,11,16,14,20
Ortalama=		3.089	3.945	4.676	8.301	

Tablo D.7 α ve β Parametreleri Etkisini Belirleme Deneme Sonuçları

Deneme No	Parametreler		Tekrar No									
	α	β	1	2	3	4	5	6	7	8	9	10
Problem No=			ta011									
1	0	0	1937	1937	1937	1937	1937	1937	1937	1937	1937	1937
2	0	0.5	1738	1738	1738	1738	1738	1738	1738	1738	1738	1738
3	0	1	1738	1738	1738	1738	1738	1738	1738	1738	1738	1738
4	0	2	1738	1738	1738	1738	1738	1738	1738	1738	1738	1738
5	0	5	1738	1738	1738	1738	1738	1738	1738	1738	1738	1738
6	0.5	0	1879	1838	1825	1816	1822	1841	1879	1879	1861	1861
7	0.5	0.5	1631	1639	1639	1625	1631	1624	1634	1642	1634	1646
8	0.5	1	1661	1654	1662	1649	1660	1673	1675	1663	1646	1665
9	0.5	2	1685	1686	1658	1674	1664	1676	1688	1686	1679	1677
10	0.5	5	1710	1711	1716	1713	1716	1716	1716	1710	1736	1716
11	1	0	1879	1816	1812	1801	1816	1791	1879	1818	1844	1780
12	1	0.5	1647	1646	1640	1630	1667	1654	1657	1641	1648	1657
13	1	1	1643	1644	1646	1634	1640	1646	1655	1639	1634	1636
14	1	2	1664	1651	1662	1652	1665	1661	1662	1667	1640	1656
15	1	5	1689	1694	1686	1696	1692	1697	1689	1688	1689	1689
16	2	0	1879	1744	1859	1861	1876	1870	1766	1842	1870	1877
17	2	0.5	1658	1660	1652	1707	1681	1668	1682	1670	1673	1651
18	2	1	1660	1634	1651	1660	1656	1648	1652	1634	1657	1660
19	2	2	1623	1645	1649	1646	1653	1627	1640	1623	1613	1623
20	2	5	1665	1679	1660	1672	1649	1640	1672	1676	1667	1649
21	5	0	1845	1879	1802	1818	1798	1859	1820	1859	1827	1802
22	5	0.5	1702	1681	1667	1672	1689	1659	1690	1674	1671	1694
23	5	1	1650	1668	1686	1693	1671	1677	1675	1668	1699	1704
24	5	2	1644	1658	1651	1657	1656	1665	1661	1642	1653	1634
25	5	5	1638	1643	1619	1634	1648	1637	1640	1649	1629	1638

Tablo D.7 α ve β Parametreleri Etkisini Belirleme Deneme Sonuçları (Devam)

Deneme No	Parametreler		Tekrar No									
	α	β	1	2	3	4	5	6	7	8	9	10
Problem No=			ta012									
1	0	0	2037	2037	2037	2037	2037	2037	2037	2037	2037	2037
2	0	0.5	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
3	0	1	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
4	0	2	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
5	0	5	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
6	0.5	0	1865	1894	1912	1909	1886	1939	1854	1904	1921	1849
7	0.5	0.5	1774	1778	1780	1775	1787	1776	1780	1777	1773	1766
8	0.5	1	1860	1845	1828	1853	1846	1852	1805	1864	1859	1803
9	0.5	2	1881	1877	1881	1877	1881	1881	1878	1873	1881	1877
10	0.5	5	1918	1918	1918	1918	1915	1918	1918	1918	1918	1915
11	1	0	1921	1867	1884	1901	1889	1960	1865	1922	1904	1909
12	1	0.5	1780	1767	1772	1763	1778	1769	1768	1758	1773	1781
13	1	1	1777	1785	1771	1765	1771	1771	1789	1785	1782	1787
14	1	2	1803	1869	1859	1798	1804	1848	1838	1848	1856	1801
15	1	5	1878	1889	1888	1897	1877	1881	1881	1881	1889	1881
16	2	0	1868	1953	1904	1953	1871	1874	1922	1909	1848	1953
17	2	0.5	1729	1767	1761	1802	1746	1806	1787	1776	1754	1767
18	2	1	1782	1776	1779	1758	1785	1753	1777	1771	1776	1780
19	2	2	1793	1759	1778	1757	1785	1769	1775	1768	1768	1780
20	2	5	1862	1874	1866	1858	1850	1795	1859	1869	1865	1865
21	5	0	1904	1945	1901	1873	1881	1909	1950	1944	1828	1873
22	5	0.5	1794	1792	1777	1805	1765	1799	1813	1774	1803	1770
23	5	1	1777	1781	1763	1765	1760	1763	1755	1757	1721	1757
24	5	2	1765	1769	1767	1771	1764	1782	1786	1781	1774	1786
25	5	5	1775	1773	1781	1782	1779	1784	1776	1789	1781	1774

Tablo D.7 α ve β Parametreleri Etkisini Belirleme Deneme Sonuçları (Devam)

Deneme No	Parametreler		Tekrar No									
	α	β	1	2	3	4	5	6	7	8	9	10
Problem No=			ta013									
1	0	0	1757	1757	1757	1757	1757	1757	1757	1757	1757	1757
2	0	0.5	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
3	0	1	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
4	0	2	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
5	0	5	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
6	0.5	0	1710	1682	1698	1677	1664	1677	1674	1707	1745	1691
7	0.5	0.5	1548	1544	1553	1553	1549	1549	1539	1556	1554	1547
8	0.5	1	1554	1548	1544	1539	1552	1539	1544	1549	1534	1548
9	0.5	2	1552	1544	1570	1570	1563	1560	1576	1572	1555	1576
10	0.5	5	1587	1586	1589	1587	1587	1587	1587	1587	1586	1579
11	1	0	1682	1757	1654	1700	1665	1709	1674	1667	1734	1660
12	1	0.5	1594	1541	1575	1549	1559	1548	1601	1554	1540	1557
13	1	1	1555	1554	1534	1551	1552	1547	1556	1546	1554	1547
14	1	2	1534	1539	1539	1544	1544	1539	1537	1551	1548	1544
15	1	5	1573	1576	1568	1573	1570	1577	1577	1558	1564	1579
16	2	0	1686	1685	1707	1735	1672	1668	1704	1734	1680	1734
17	2	0.5	1615	1568	1604	1612	1670	1625	1625	1585	1636	1595
18	2	1	1552	1554	1559	1562	1575	1560	1563	1559	1545	1553
19	2	2	1549	1554	1556	1554	1537	1561	1543	1554	1555	1553
20	2	5	1554	1554	1566	1549	1544	1547	1544	1546	1555	1551
21	5	0	1673	1683	1690	1665	1701	1679	1658	1696	1709	1659
22	5	0.5	1596	1642	1625	1572	1590	1584	1611	1621	1640	1603
23	5	1	1596	1635	1584	1640	1632	1544	1636	1614	1627	1620
24	5	2	1561	1591	1584	1570	1561	1601	1571	1588	1581	1597
25	5	5	1545	1556	1544	1549	1534	1543	1540	1546	1540	1546

Tablo D.7 α ve β Parametreleri Etkisini Belirleme Deneme Sonuçları (Devam)

Deneme No	Parametreler		Tekrar No									
	α	β	1	2	3	4	5	6	7	8	9	10
Problem No=			ta014									
1	0	0	1655	1655	1655	1655	1655	1655	1655	1655	1655	1655
2	0	0.5	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
3	0	1	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
4	0	2	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
5	0	5	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
6	0.5	0	1593	1573	1605	1585	1610	1650	1628	1637	1637	1594
7	0.5	0.5	1466	1468	1474	1472	1470	1468	1466	1463	1472	1478
8	0.5	1	1478	1466	1467	1472	1467	1476	1469	1482	1472	1467
9	0.5	2	1482	1476	1474	1476	1474	1474	1474	1474	1489	1489
10	0.5	5	1510	1510	1510	1516	1517	1528	1526	1504	1501	1516
11	1	0	1650	1585	1632	1632	1626	1637	1593	1626	1599	1650
12	1	0.5	1476	1473	1445	1473	1465	1467	1477	1469	1475	1448
13	1	1	1467	1466	1466	1467	1478	1466	1461	1472	1467	1456
14	1	2	1467	1474	1467	1467	1467	1461	1482	1467	1479	1467
15	1	5	1489	1474	1474	1482	1474	1477	1474	1487	1474	1474
16	2	0	1650	1626	1637	1601	1577	1650	1585	1590	1594	1626
17	2	0.5	1431	1441	1487	1476	1481	1467	1437	1467	1457	1459
18	2	1	1479	1472	1469	1449	1468	1469	1465	1448	1494	1479
19	2	2	1467	1478	1467	1473	1472	1465	1469	1473	1467	1473
20	2	5	1480	1467	1469	1474	1474	1472	1466	1474	1474	1466
21	5	0	1621	1604	1608	1637	1637	1637	1570	1588	1637	1637
22	5	0.5	1480	1461	1463	1478	1435	1473	1455	1456	1476	1447
23	5	1	1452	1433	1444	1441	1449	1442	1456	1460	1455	1459
24	5	2	1462	1460	1466	1464	1472	1460	1435	1472	1480	1463
25	5	5	1467	1470	1484	1458	1467	1459	1470	1472	1474	1466

Tablo D.8 pl ve pg Parametreleri Etkisini Belirleme Deneme Sonuçları

Deneme No	Parametreler		Tekrar No									
	pl	pg	1	2	3	4	5	6	7	8	9	10
Problem No=			ta011									
1	0	0	1738	1738	1738	1738	1738	1738	1738	1738	1738	1738
2	0	0.1	1720	1710	1720	1710	1720	1720	1753	1720	1720	1710
3	0	0.2	1720	1742	1720	1720	1720	1742	1742	1720	1720	1742
4	0	0.3	1742	1720	1720	1742	1710	1720	1742	1720	1710	1720
5	0	0.5	1720	1720	1742	1742	1720	1720	1720	1742	1720	1742
6	0.1	0	1738	1738	1738	1738	1738	1738	1738	1738	1738	1738
7	0.1	0.1	1648	1641	1638	1638	1645	1641	1642	1646	1634	1648
8	0.1	0.2	1652	1665	1645	1652	1634	1642	1657	1646	1648	1638
9	0.1	0.3	1646	1643	1668	1656	1634	1652	1646	1653	1648	1638
10	0.1	0.5	1652	1666	1670	1664	1647	1680	1680	1668	1704	1658
11	0.2	0	1738	1738	1738	1738	1738	1738	1738	1738	1738	1738
12	0.2	0.1	1639	1648	1657	1657	1639	1645	1648	1648	1648	1656
13	0.2	0.2	1651	1647	1657	1648	1648	1646	1650	1652	1655	1648
14	0.2	0.3	1643	1639	1648	1635	1648	1634	1649	1655	1648	1651
15	0.2	0.5	1634	1631	1648	1639	1636	1632	1641	1648	1639	1651
16	0.3	0	1738	1738	1738	1738	1738	1738	1738	1738	1738	1738
17	0.3	0.1	1669	1663	1673	1673	1673	1673	1665	1656	1648	1673
18	0.3	0.2	1657	1648	1663	1663	1648	1663	1657	1663	1657	1663
19	0.3	0.3	1657	1654	1648	1657	1648	1650	1656	1653	1657	1657
20	0.3	0.5	1657	1657	1648	1634	1648	1648	1648	1657	1648	1648
21	0.5	0	1738	1738	1738	1738	1738	1738	1738	1738	1738	1738
22	0.5	0.1	1673	1673	1673	1673	1673	1673	1673	1673	1673	1673
23	0.5	0.2	1673	1673	1673	1673	1673	1673	1673	1673	1673	1673
24	0.5	0.3	1673	1673	1673	1673	1673	1671	1673	1673	1673	1673
25	0.5	0.5	1673	1669	1669	1669	1669	1669	1673	1669	1669	1673

Tablo D.8 pl ve pg Parametreleri Etkisini Belirleme Deneme Sonuçları (Devam)

Deneme No	Parametreler		Tekrar No									
	pl	pg	1	2	3	4	5	6	7	8	9	10
Problem No=			ta012									
1	0	0	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
2	0	0.1	1881	1930	1897	1911	1931	1911	1881	1881	1930	1881
3	0	0.2	1899	1930	1881	1881	1881	1917	1881	1899	1948	1881
4	0	0.3	1911	1878	1899	1881	1930	1930	1899	1917	1934	1924
5	0	0.5	1881	1931	1934	1934	1931	1881	1899	1881	1899	1931
6	0.1	0	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
7	0.1	0.1	1772	1775	1781	1777	1787	1768	1778	1784	1774	1766
8	0.1	0.2	1774	1778	1784	1779	1773	1776	1737	1794	1775	1770
9	0.1	0.3	1765	1783	1775	1784	1770	1781	1782	1778	1781	1767
10	0.1	0.5	1780	1786	1778	1784	1776	1771	1800	1763	1762	1778
11	0.2	0	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
12	0.2	0.1	1782	1793	1794	1786	1782	1777	1791	1800	1797	1789
13	0.2	0.2	1790	1783	1782	1777	1785	1775	1793	1786	1770	1789
14	0.2	0.3	1778	1769	1781	1777	1779	1779	1782	1788	1784	1783
15	0.2	0.5	1784	1784	1777	1784	1775	1773	1780	1776	1770	1772
16	0.3	0	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
17	0.3	0.1	1791	1807	1807	1797	1807	1800	1807	1783	1805	1798
18	0.3	0.2	1797	1784	1798	1788	1777	1791	1800	1784	1786	1791
19	0.3	0.3	1797	1781	1797	1776	1787	1803	1791	1789	1795	1783
20	0.3	0.5	1794	1794	1783	1795	1792	1778	1789	1793	1782	1772
21	0.5	0	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
22	0.5	0.1	1807	1807	1807	1811	1807	1818	1798	1805	1798	1800
23	0.5	0.2	1801	1811	1807	1805	1807	1805	1810	1783	1807	1814
24	0.5	0.3	1803	1788	1788	1813	1789	1801	1799	1788	1799	1797
25	0.5	0.5	1802	1798	1797	1797	1798	1786	1788	1792	1803	1807

Tablo D.8 *pl* ve *pg* Parametreleri Etkisini Belirleme Deneme Sonuçları (Devam)

Deneme No	Parametreler		Tekrar No									
	<i>pl</i>	<i>pg</i>	1	2	3	4	5	6	7	8	9	10
Problem No=			ta013									
1	0	0	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
2	0	0.1	1704	1689	1695	1677	1677	1677	1695	1695	1695	1695
3	0	0.2	1677	1695	1695	1695	1677	1695	1677	1704	1695	1677
4	0	0.3	1695	1677	1695	1695	1704	1677	1677	1695	1695	1695
5	0	0.5	1695	1695	1704	1695	1695	1695	1695	1677	1695	1695
6	0.1	0	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
7	0.1	0.1	1554	1552	1537	1544	1552	1556	1547	1550	1544	1546
8	0.1	0.2	1576	1576	1558	1537	1557	1568	1564	1555	1561	1548
9	0.1	0.3	1558	1580	1576	1574	1574	1574	1598	1575	1569	1569
10	0.1	0.5	1608	1573	1590	1607	1608	1593	1637	1569	1595	1592
11	0.2	0	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
12	0.2	0.1	1548	1544	1545	1544	1556	1556	1544	1544	1550	1556
13	0.2	0.2	1548	1544	1544	1540	1554	1551	1554	1539	1539	1539
14	0.2	0.3	1546	1546	1537	1539	1544	1534	1544	1539	1554	1539
15	0.2	0.5	1543	1547	1539	1544	1544	1544	1544	1539	1537	1539
16	0.3	0	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
17	0.3	0.1	1548	1553	1554	1565	1548	1554	1544	1558	1561	1565
18	0.3	0.2	1570	1558	1547	1552	1544	1560	1557	1557	1554	1562
19	0.3	0.3	1559	1550	1547	1554	1554	1554	1552	1544	1544	1548
20	0.3	0.5	1547	1544	1544	1547	1554	1557	1547	1554	1554	1544
21	0.5	0	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
22	0.5	0.1	1557	1557	1566	1575	1572	1574	1572	1569	1559	1570
23	0.5	0.2	1562	1554	1570	1579	1570	1578	1570	1559	1578	1568
24	0.5	0.3	1544	1555	1572	1575	1559	1548	1548	1566	1574	1567
25	0.5	0.5	1570	1553	1579	1544	1555	1547	1570	1563	1566	1570

Tablo D.8 *pl* ve *pg* Parametreleri Etkisini Belirleme Deneme Sonuçları (Devam)

Deneme No	Parametreler		Tekrar No									
	<i>pl</i>	<i>pg</i>	1	2	3	4	5	6	7	8	9	10
Problem No=			ta014									
1	0	0	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
2	0	0.1	1585	1585	1531	1526	1585	1585	1531	1582	1582	1585
3	0	0.2	1585	1531	1585	1585	1585	1585	1585	1585	1585	1585
4	0	0.3	1572	1585	1585	1608	1585	1575	1585	1585	1531	1608
5	0	0.5	1585	1557	1585	1554	1585	1585	1585	1585	1585	1572
6	0.1	0	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
7	0.1	0.1	1455	1466	1467	1466	1467	1470	1470	1472	1472	1470
8	0.1	0.2	1465	1452	1473	1478	1477	1472	1458	1481	1467	1470
9	0.1	0.3	1496	1489	1486	1463	1459	1453	1465	1471	1452	1471
10	0.1	0.5	1464	1461	1472	1489	1468	1479	1483	1467	1492	1488
11	0.2	0	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
12	0.2	0.1	1473	1467	1468	1464	1469	1472	1482	1466	1470	1471
13	0.2	0.2	1468	1473	1470	1467	1474	1472	1468	1466	1467	1466
14	0.2	0.3	1469	1471	1472	1472	1467	1465	1467	1466	1467	1473
15	0.2	0.5	1467	1468	1467	1469	1472	1470	1470	1466	1462	1472
16	0.3	0	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
17	0.3	0.1	1480	1478	1476	1481	1474	1475	1470	1472	1474	1477
18	0.3	0.2	1479	1463	1472	1467	1472	1478	1472	1472	1470	1470
19	0.3	0.3	1464	1479	1464	1472	1466	1470	1467	1470	1480	1468
20	0.3	0.5	1464	1467	1466	1472	1463	1461	1470	1480	1463	1475
21	0.5	0	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
22	0.5	0.1	1489	1478	1478	1491	1489	1478	1489	1486	1478	1478
23	0.5	0.2	1478	1485	1482	1466	1484	1485	1474	1478	1478	1478
24	0.5	0.3	1482	1472	1477	1478	1487	1487	1489	1481	1482	1478
25	0.5	0.5	1481	1477	1470	1463	1483	1474	1487	1487	1474	1487

Tablo D.9 İterasyon Sayısı Parametresi Etkisini Belirleme Deneme Sonuçları

$t_{\max}$	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
Problem No=	ta011									
1	1753	1760	1760	1738	1790	1753	1753	1760	1760	1738
2	1738	1790	1742	1720	1738	1738	1753	1738	1720	1738
3	1738	1738	1738	1742	1720	1753	1738	1738	1724	1742
5	1738	1742	1720	1720	1720	1720	1729	1720	1720	1720
8	1735	1720	1745	1720	1720	1720	1720	1720	1720	1720
10	1720	1720	1720	1695	1706	1706	1731	1720	1720	1724
20	1720	1720	1718	1704	1720	1705	1718	1720	1720	1688
50	1683	1712	1673	1682	1720	1720	1699	1684	1694	1673
75	1686	1715	1667	1676	1673	1699	1700	1705	1705	1695
100	1691	1715	1685	1699	1715	1703	1689	1668	1657	1681
200	1673	1669	1681	1672	1650	1659	1689	1690	1685	1670
500	1666	1673	1685	1673	1676	1654	1667	1647	1684	1671
750	1648	1664	1663	1658	1660	1676	1671	1661	1658	1657
1000	1673	1651	1650	1661	1648	1641	1651	1667	1655	1657
1250	1648	1664	1669	1636	1660	1646	1657	1659	1659	1650
1500	1669	1670	1631	1663	1653	1658	1655	1663	1656	1657
1750	1637	1662	1665	1647	1648	1663	1654	1658	1659	1651
2000	1657	1656	1646	1661	1658	1646	1649	1658	1660	1654
2250	1657	1643	1651	1648	1657	1656	1640	1657	1646	1637
2500	1654	1639	1642	1648	1655	1648	1648	1643	1659	1639
2750	1637	1653	1634	1644	1666	1665	1649	1636	1646	1643
3000	1645	1655	1636	1634	1626	1658	1650	1644	1654	1650
3500	1634	1654	1652	1657	1657	1640	1638	1652	1644	1643
4000	1636	1625	1646	1651	1652	1652	1642	1646	1648	1646
4500	1632	1627	1646	1647	1635	1647	1648	1641	1646	1643
5000	1631	1635	1635	1631	1646	1634	1634	1638	1643	1648
6000	1632	1643	1634	1642	1652	1655	1644	1623	1645	1648

Tablo D.9 İterasyon Sayısı Parametre Etkisini Belirleme Deneme Sonuçları (Devam)

$t_{\max}$	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
Problem No=	ta012									
1	1923	1923	1923	1948	1934	1931	1931	1934	1923	1934
2	1931	1907	1930	1923	1931	1912	1881	1881	1881	1881
3	1919	1914	1931	1912	1881	1931	1881	1934	1874	1881
5	1874	1930	1859	1881	1881	1881	1899	1863	1862	1881
8	1881	1894	1881	1830	1830	1912	1881	1931	1815	1881
10	1881	1881	1880	1881	1930	1909	1857	1831	1907	1880
20	1827	1893	1819	1811	1825	1832	1819	1848	1813	1879
50	1825	1824	1809	1812	1845	1776	1786	1822	1820	1802
75	1770	1800	1831	1810	1837	1848	1833	1811	1802	1832
100	1802	1816	1813	1833	1783	1784	1799	1802	1801	1812
200	1804	1774	1799	1787	1810	1794	1815	1799	1789	1791
500	1788	1779	1788	1772	1803	1803	1786	1784	1786	1778
750	1795	1793	1786	1772	1768	1791	1782	1780	1764	1796
1000	1795	1787	1800	1786	1787	1783	1781	1799	1800	1784
1250	1787	1771	1792	1761	1790	1782	1773	1787	1787	1793
1500	1798	1775	1774	1775	1767	1764	1774	1791	1783	1801
1750	1777	1784	1765	1768	1762	1790	1773	1789	1782	1781
2000	1782	1772	1773	1780	1784	1783	1769	1789	1750	1782
2250	1760	1767	1795	1781	1778	1769	1770	1755	1761	1784
2500	1788	1779	1779	1781	1774	1756	1780	1785	1760	1773
2750	1783	1754	1787	1769	1771	1749	1764	1786	1777	1776
3000	1771	1788	1775	1768	1779	1767	1772	1757	1785	1790
3500	1762	1782	1782	1759	1782	1756	1756	1768	1787	1776
4000	1742	1747	1778	1762	1749	1772	1769	1776	1760	1757
4500	1766	1783	1780	1770	1774	1775	1778	1775	1737	1739
5000	1747	1735	1767	1781	1763	1754	1757	1743	1774	1773
6000	1776	1745	1752	1759	1761	1773	1733	1767	1766	1774

Tablo D.9 İterasyon Sayısı Parametre Etkisini Belirleme Deneme Sonuçları (Devam)

$t_{\max}$	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
Problem No=	ta013									
1	1713	1713	1695	1713	1695	1695	1695	1695	1695	1695
2	1695	1677	1713	1695	1695	1693	1695	1695	1695	1713
3	1677	1695	1695	1695	1677	1695	1695	1695	1677	1677
5	1688	1677	1677	1677	1677	1649	1677	1677	1695	1664
8	1639	1651	1677	1682	1695	1677	1653	1660	1640	1639
10	1677	1631	1677	1621	1677	1668	1627	1677	1677	1677
20	1660	1659	1654	1637	1649	1649	1635	1635	1649	1644
50	1613	1572	1609	1598	1616	1608	1597	1651	1645	1610
75	1585	1614	1621	1631	1622	1597	1607	1635	1575	1612
100	1636	1606	1613	1611	1609	1595	1575	1585	1555	1618
200	1559	1588	1598	1571	1593	1580	1585	1579	1583	1573
500	1585	1578	1572	1553	1588	1566	1564	1573	1571	1581
750	1561	1550	1560	1561	1572	1561	1577	1564	1560	1549
1000	1560	1564	1550	1587	1572	1555	1537	1564	1556	1571
1250	1553	1565	1564	1580	1570	1578	1574	1546	1562	1553
1500	1555	1566	1564	1561	1561	1558	1556	1544	1551	1569
1750	1555	1568	1538	1559	1566	1553	1549	1554	1554	1549
2000	1556	1564	1560	1552	1557	1554	1549	1548	1557	1570
2250	1537	1556	1543	1556	1558	1550	1552	1548	1545	1552
2500	1555	1556	1544	1544	1551	1554	1551	1556	1556	1537
2750	1559	1555	1559	1544	1544	1555	1565	1537	1547	1570
3000	1551	1555	1554	1548	1563	1546	1548	1557	1553	1554
3500	1554	1551	1555	1548	1544	1556	1555	1553	1548	1554
4000	1544	1570	1548	1544	1537	1556	1551	1553	1555	1544
4500	1554	1549	1544	1557	1556	1555	1548	1541	1547	1546
5000	1545	1550	1554	1538	1546	1553	1537	1544	1554	1544
6000	1545	1544	1545	1547	1553	1554	1545	1544	1554	1537

Tablo D.9 İterasyon Sayısı Parametre Etkisini Belirleme Deneme Sonuçları (Devam)

$t_{\max}$	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
Problem No=	ta014									
1	1585	1598	1585	1585	1608	1585	1585	1585	1585	1585
2	1566	1585	1585	1585	1579	1598	1585	1549	1559	1566
3	1566	1549	1566	1579	1579	1585	1574	1585	1579	1583
5	1559	1548	1551	1561	1519	1549	1554	1531	1574	1526
8	1550	1542	1562	1544	1540	1544	1544	1517	1549	1553
10	1554	1507	1517	1574	1550	1559	1547	1549	1522	1544
20	1537	1526	1544	1541	1507	1521	1533	1526	1550	1516
50	1539	1512	1469	1527	1514	1541	1497	1498	1514	1518
75	1502	1442	1511	1526	1514	1527	1529	1497	1475	1493
100	1480	1510	1517	1485	1486	1459	1496	1521	1502	1498
200	1516	1455	1483	1477	1510	1450	1484	1486	1460	1530
500	1473	1463	1471	1474	1439	1468	1454	1453	1493	1487
750	1471	1460	1457	1485	1457	1474	1468	1440	1476	1466
1000	1461	1468	1465	1472	1458	1462	1453	1468	1449	1475
1250	1460	1439	1460	1456	1467	1490	1467	1475	1458	1463
1500	1468	1455	1449	1468	1455	1425	1445	1476	1466	1459
1750	1465	1469	1472	1467	1465	1444	1436	1474	1417	1456
2000	1442	1428	1444	1468	1443	1462	1449	1440	1464	1478
2250	1446	1445	1463	1459	1429	1441	1449	1438	1441	1467
2500	1425	1454	1473	1445	1456	1473	1447	1463	1441	1453
2750	1448	1450	1464	1456	1437	1458	1449	1439	1437	1469
3000	1461	1467	1463	1447	1439	1425	1450	1454	1468	1428
3500	1414	1443	1457	1444	1455	1448	1442	1459	1453	1427
4000	1451	1441	1455	1436	1430	1440	1453	1446	1438	1438
4500	1458	1438	1418	1435	1443	1445	1458	1440	1465	1453
5000	1429	1442	1443	1441	1454	1458	1449	1425	1440	1442
6000	1447	1453	1454	1443	1444	1425	1442	1456	1447	1448

Tablo D.10 Karınca Sayısı Parametre Etkisini Belirleme Deneme Sonuçları

$k_{\max}$	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
Problem No=	ta011									
1	1790	1802	1753	1773	1785	1790	1790	1760	1790	1730
2	1711	1709	1790	1669	1718	1725	1690	1687	1680	1712
4	1656	1685	1670	1699	1700	1687	1674	1677	1694	1684
8	1670	1664	1681	1655	1656	1667	1665	1666	1670	1653
12	1673	1643	1666	1659	1655	1645	1659	1650	1643	1664
14	1663	1635	1641	1663	1655	1670	1648	1668	1647	1649
16	1646	1639	1630	1653	1638	1652	1632	1654	1654	1657
20	1646	1648	1633	1636	1640	1640	1647	1632	1634	1640
24	1643	1648	1649	1649	1654	1634	1646	1651	1647	1640
36	1666	1637	1673	1657	1657	1662	1669	1663	1655	1657
48	1673	1661	1673	1669	1690	1673	1669	1673	1673	1673
Problem No=	ta012									
1	1907	1899	1946	1888	1946	1942	1948	1881	1932	1921
2	1762	1798	1813	1781	1776	1786	1792	1776	1800	1797
4	1777	1767	1756	1783	1760	1748	1780	1786	1744	1782
8	1773	1750	1744	1764	1752	1758	1775	1738	1765	1771
12	1761	1737	1735	1754	1756	1765	1747	1742	1746	1758
14	1758	1744	1760	1758	1757	1755	1764	1746	1741	1748
16	1746	1761	1758	1754	1778	1746	1744	1743	1762	1757
20	1772	1764	1773	1765	1766	1761	1770	1770	1771	1772
24	1775	1776	1770	1779	1771	1769	1773	1769	1783	1781
36	1772	1779	1778	1776	1771	1767	1787	1778	1780	1775
48	1767	1777	1762	1768	1785	1781	1760	1773	1780	1766

Tablo D.10 Karınca Sayısı Parametre Etkisini Belirleme Deneme Sonuçları (Devam)

$k_{\max}$	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
Problem No=	ta013									
1	1747	1717	1695	1729	1736	1717	1704	1772	1731	1695
2	1619	1627	1620	1620	1616	1642	1631	1729	1632	1642
4	1593	1584	1610	1617	1613	1630	1571	1610	1614	1629
8	1592	1586	1588	1582	1594	1577	1597	1595	1597	1593
12	1572	1599	1596	1595	1562	1598	1589	1595	1602	1569
14	1581	1557	1579	1578	1575	1561	1563	1564	1557	1597
16	1561	1544	1557	1556	1558	1557	1567	1558	1554	1570
20	1553	1552	1548	1544	1558	1551	1548	1555	1539	1556
24	1539	1544	1554	1544	1549	1551	1556	1548	1552	1549
36	1554	1555	1564	1553	1555	1555	1547	1544	1544	1556
48	1548	1558	1551	1554	1556	1556	1556	1544	1544	1564
Problem No=	ta014									
1	1547	1585	1583	1622	1579	1605	1608	1585	1597	1572
2	1507	1509	1489	1514	1485	1480	1515	1516	1469	1510
4	1478	1488	1486	1500	1464	1484	1484	1482	1474	1480
8	1476	1449	1458	1459	1464	1437	1462	1467	1466	1468
12	1459	1451	1448	1471	1467	1468	1444	1473	1469	1443
14	1466	1452	1444	1440	1446	1458	1438	1444	1455	1453
16	1445	1448	1457	1460	1447	1462	1444	1442	1442	1462
20	1448	1428	1444	1435	1438	1441	1455	1445	1450	1431
24	1454	1435	1436	1463	1437	1443	1440	1461	1440	1428
36	1437	1445	1465	1455	1449	1449	1429	1452	1440	1466
48	1452	1472	1466	1425	1462	1469	1458	1456	1448	1459

Ek E**Tablo E.1 Karıncalar Arası İşbirliği Etkisini Belirleme Deneme Sonuçları**

Problem No	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
$\alpha=0$										
<i>ta011</i>	1721	1721	1748	1749	1749	1753	1754	1766	1742	1743
<i>ta012</i>	1923	1923	1923	1923	1923	1923	1923	1923	1923	1923
<i>ta013</i>	1695	1695	1695	1695	1695	1695	1695	1695	1695	1695
<i>ta014</i>	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
$\alpha=1$										
<i>ta011</i>	1637	1648	1649	1647	1647	1628	1636	1650	1657	1648
<i>ta012</i>	1776	1778	1772	1777	1765	1784	1780	1777	1780	1775
<i>ta013</i>	1544	1547	1537	1537	1551	1542	1544	1550	1546	1555
<i>ta014</i>	1462	1473	1452	1459	1456	1472	1470	1446	1464	1444

EK F.

Tablo F.1.1 R-Prob Geçiş Kuralı Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1297	1297	1297	1297	1297	1297	1297	1297	1297	1297
20x5	ta002	1368	1368	1368	1368	1368	1368	1368	1368	1368	1368
20x5	ta003	1114	1114	1114	1114	1114	1114	1114	1114	1114	1114
20x5	ta004	1337	1333	1336	1334	1333	1332	1334	1340	1331	1333
20x5	ta005	1261	1311	1277	1313	1271	1271	1282	1291	1311	1286
20x5	ta006	1219	1220	1210	1219	1219	1219	1219	1220	1210	1216
20x5	ta007	1262	1266	1251	1272	1259	1264	1256	1259	1263	1274
20x5	ta008	1217	1219	1219	1212	1225	1223	1223	1225	1214	1219
20x5	ta009	1287	1296	1281	1289	1294	1288	1284	1290	1287	1296
20x5	ta010	1153	1159	1161	1149	1161	1140	1138	1152	1154	1161
20x10	ta011	1638	1646	1646	1639	1648	1645	1634	1629	1652	1648
20x10	ta012	1745	1770	1772	1771	1772	1776	1761	1774	1749	1755
20x10	ta013	1539	1547	1535	1547	1542	1561	1546	1554	1556	1554
20x10	ta014	1438	1452	1453	1456	1428	1452	1438	1434	1438	1441
20x10	ta015	1541	1539	1526	1540	1543	1535	1519	1534	1521	1542
20x10	ta016	1442	1439	1444	1450	1444	1444	1445	1451	1444	1452
20x10	ta017	1531	1559	1563	1548	1550	1560	1540	1566	1558	1538
20x10	ta018	1572	1570	1590	1588	1586	1592	1579	1587	1586	1575
20x10	ta019	1704	1686	1699	1695	1694	1685	1697	1698	1711	1701
20x10	ta020	1675	1676	1664	1686	1664	1676	1682	1677	1670	1665
20x20	ta021	2429	2464	2478	2450	2454	2476	2447	2468	2425	2460
20x20	ta022	2194	2186	2195	2182	2178	2185	2195	2183	2183	2176
20x20	ta023	2436	2440	2425	2420	2424	2401	2429	2414	2426	2422
20x20	ta024	2301	2319	2309	2310	2318	2304	2311	2313	2317	2314
20x20	ta025	2399	2390	2389	2386	2384	2382	2371	2398	2386	2382
20x20	ta026	2317	2324	2307	2303	2309	2321	2314	2310	2306	2301
20x20	ta027	2362	2366	2366	2368	2348	2368	2366	2366	2369	2366
20x20	ta028	2287	2278	2295	2279	2282	2291	2285	2294	2285	2277
20x20	ta029	2379	2345	2361	2384	2388	2377	2369	2353	2367	2368
20x20	ta030	2276	2278	2273	2273	2281	2267	2282	2278	2269	2277

Tablo F.1.2 PR-Prob Geçiş Kuralı Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1297	1297	1297	1297	1297	1297	1297	1297	1297	1297
20x5	ta002	1368	1368	1368	1368	1368	1368	1371	1368	1368	1368
20x5	ta003	1127	1118	1119	1115	1116	1114	1119	1116	1114	1114
20x5	ta004	1336	1348	1329	1334	1322	1328	1341	1334	1337	1341
20x5	ta005	1280	1277	1292	1277	1276	1305	1271	1310	1256	1267
20x5	ta006	1210	1210	1219	1214	1220	1219	1216	1219	1219	1219
20x5	ta007	1259	1259	1251	1266	1251	1273	1269	1259	1259	1259
20x5	ta008	1220	1216	1215	1218	1219	1207	1218	1220	1221	1219
20x5	ta009	1298	1296	1297	1297	1297	1294	1290	1294	1290	1276
20x5	ta010	1161	1160	1155	1161	1141	1140	1161	1161	1161	1151
20x10	ta011	1646	1636	1648	1645	1648	1647	1635	1646	1630	1634
20x10	ta012	1747	1735	1767	1781	1763	1754	1757	1743	1774	1773
20x10	ta013	1544	1537	1546	1542	1550	1544	1547	1552	1552	1543
20x10	ta014	1452	1435	1450	1440	1438	1453	1428	1440	1445	1437
20x10	ta015	1531	1530	1543	1524	1522	1526	1521	1542	1534	1516
20x10	ta016	1447	1437	1444	1447	1444	1454	1435	1438	1435	1443
20x10	ta017	1548	1539	1531	1537	1548	1548	1531	1549	1531	1531
20x10	ta018	1586	1583	1588	1586	1590	1584	1582	1580	1593	1570
20x10	ta019	1675	1706	1692	1701	1696	1699	1702	1694	1689	1688
20x10	ta020	1675	1677	1679	1686	1671	1688	1669	1678	1681	1684
20x20	ta021	2462	2457	2433	2448	2429	2451	2449	2454	2449	2453
20x20	ta022	2172	2167	2203	2179	2142	2187	2162	2188	2175	2185
20x20	ta023	2422	2413	2433	2441	2416	2421	2422	2400	2427	2446
20x20	ta024	2309	2303	2306	2286	2318	2316	2305	2311	2287	2291
20x20	ta025	2390	2395	2408	2392	2359	2379	2400	2384	2400	2383
20x20	ta026	2317	2320	2315	2304	2290	2311	2306	2300	2323	2317
20x20	ta027	2363	2364	2362	2364	2366	2360	2362	2355	2367	2354
20x20	ta028	2285	2293	2287	2278	2287	2283	2279	2301	2279	2290
20x20	ta029	2378	2368	2349	2372	2379	2384	2368	2367	2384	2365
20x20	ta030	2288	2276	2308	2287	2287	2274	2280	2267	2271	2266

Tablo F.1.3 Psum Geçiş Kuralı Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1324	1324	1324	1324	1324	1324	1324	1322	1324	1324
20x5	ta002	1367	1368	1367	1374	1368	1367	1368	1367	1368	1371
20x5	ta003	1123	1124	1132	1128	1119	1120	1128	1120	1112	1112
20x5	ta004	1361	1338	1361	1359	1362	1362	1358	1353	1338	1338
20x5	ta005	1360	1360	1360	1360	1360	1360	1360	1360	1360	1360
20x5	ta006	1305	1305	1305	1305	1305	1305	1305	1305	1305	1305
20x5	ta007	1315	1315	1315	1315	1315	1335	1303	1314	1335	1335
20x5	ta008	1283	1274	1283	1283	1268	1270	1268	1274	1274	1276
20x5	ta009	1290	1287	1317	1304	1314	1300	1300	1299	1287	1289
20x5	ta010	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183
20x10	ta011	1726	1715	1712	1725	1730	1726	1726	1730	1712	1722
20x10	ta012	1796	1825	1859	1807	1826	1796	1811	1816	1807	1773
20x10	ta013	1661	1663	1619	1658	1655	1624	1629	1636	1635	1641
20x10	ta014	1498	1500	1480	1549	1485	1487	1492	1500	1499	1500
20x10	ta015	1603	1603	1603	1603	1603	1603	1603	1603	1603	1603
20x10	ta016	1454	1464	1463	1464	1463	1454	1463	1464	1471	1463
20x10	ta017	1621	1641	1641	1660	1653	1653	1641	1641	1629	1641
20x10	ta018	1644	1698	1653	1688	1682	1677	1654	1674	1651	1687
20x10	ta019	1720	1741	1722	1722	1726	1731	1733	1731	1719	1739
20x10	ta020	1707	1713	1708	1715	1743	1724	1708	1672	1713	1708
20x20	ta021	2494	2447	2460	2453	2449	2453	2460	2471	2452	2439
20x20	ta022	2235	2168	2212	2226	2186	2212	2186	2212	2186	2180
20x20	ta023	2473	2478	2499	2463	2463	2484	2477	2473	2465	2444
20x20	ta024	2350	2365	2331	2386	2406	2323	2382	2329	2334	2390
20x20	ta025	2502	2488	2471	2457	2434	2486	2473	2422	2497	2496
20x20	ta026	2373	2387	2370	2340	2391	2375	2347	2370	2379	2369
20x20	ta027	2378	2384	2358	2386	2370	2395	2355	2369	2370	2384
20x20	ta028	2346	2350	2341	2360	2322	2339	2351	2360	2352	2337
20x20	ta029	2414	2418	2434	2421	2463	2450	2373	2390	2432	2434
20x20	ta030	2379	2351	2382	2414	2378	2378	2481	2383	2380	2396

Tablo F.1.4 R-Eval Geçiş Kuralı Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1324	1324	1324	1324	1324	1324	1324	1324	1324	1324
20x5	ta002	1368	1368	1367	1367	1368	1367	1368	1368	1368	1368
20x5	ta003	1132	1132	1148	1112	1132	1114	1131	1133	1132	1129
20x5	ta004	1345	1363	1364	1354	1370	1349	1357	1356	1343	1345
20x5	ta005	1360	1360	1360	1360	1360	1360	1360	1360	1360	1360
20x5	ta006	1240	1240	1240	1240	1240	1240	1240	1240	1240	1240
20x5	ta007	1303	1339	1303	1303	1298	1335	1295	1295	1312	1298
20x5	ta008	1267	1266	1266	1267	1280	1263	1263	1263	1263	1267
20x5	ta009	1302	1307	1306	1297	1297	1290	1298	1292	1298	1304
20x5	ta010	1183	1183	1183	1183	1183	1183	1183	1183	1183	1183
20x10	ta011	1661	1667	1656	1668	1696	1687	1667	1666	1685	1703
20x10	ta012	1804	1807	1769	1770	1783	1778	1797	1775	1779	1794
20x10	ta013	1637	1616	1603	1600	1585	1610	1623	1622	1621	1629
20x10	ta014	1472	1481	1471	1471	1492	1461	1471	1463	1461	1474
20x10	ta015	1565	1608	1575	1575	1577	1603	1614	1575	1573	1575
20x10	ta016	1449	1469	1454	1450	1458	1449	1457	1460	1446	1453
20x10	ta017	1618	1601	1616	1599	1612	1645	1645	1645	1618	1646
20x10	ta018	1659	1667	1681	1691	1680	1671	1657	1645	1643	1660
20x10	ta019	1734	1717	1717	1716	1717	1717	1712	1717	1719	1717
20x10	ta020	1673	1714	1705	1689	1697	1705	1725	1721	1678	1727
20x20	ta021	2430	2436	2424	2431	2451	2436	2440	2442	2431	2442
20x20	ta022	2199	2207	2210	2209	2194	2201	2204	2204	2217	2206
20x20	ta023	2428	2411	2403	2418	2427	2416	2394	2420	2416	2415
20x20	ta024	2309	2320	2296	2311	2324	2335	2318	2279	2269	2284
20x20	ta025	2393	2405	2374	2383	2370	2421	2368	2435	2417	2383
20x20	ta026	2364	2349	2361	2335	2359	2358	2359	2371	2370	2352
20x20	ta027	2378	2391	2377	2390	2360	2365	2360	2360	2391	2358
20x20	ta028	2317	2346	2322	2331	2323	2312	2349	2333	2334	2299
20x20	ta029	2360	2403	2368	2367	2408	2385	2384	2389	2414	2383
20x20	ta030	2363	2306	2323	2313	2321	2370	2359	2344	2340	2345

Tablo F.2 Geçiş Kuralları Bağlı Hata Sonuçları

$n \times m$	Problem No	R-Prop				PR-Prop				Psum				R-Eval			
		Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}
20x5	ta001	1.49	1.49	1.49	0.00	1.49	1.49	1.49	0.00	3.44	3.58	3.60	0.63	3.60	3.60	3.60	0.00
20x5	ta002	0.66	0.66	0.66	0.00	0.66	0.68	0.88	0.95	0.59	0.70	1.10	2.27	0.59	0.64	0.66	0.48
20x5	ta003	3.05	3.05	3.05	0.00	3.05	3.35	4.26	3.97	2.87	3.77	4.72	6.61	2.87	4.49	6.20	10.16
20x5	ta004	2.94	3.19	3.63	2.67	2.24	3.25	4.25	7.47	3.48	4.64	5.34	10.68	3.87	4.76	5.96	9.16
20x5	ta005	2.11	4.24	6.32	18.73	1.70	3.73	6.07	16.72	10.12	10.12	10.12	0.00	10.12	10.12	10.12	0.00
20x5	ta006	1.26	1.85	2.09	3.90	1.26	1.80	2.09	3.87	9.21	9.21	9.21	0.00	3.77	3.77	3.77	0.00
20x5	ta007	0.97	1.90	2.82	6.96	0.97	1.74	2.74	7.07	5.17	6.51	7.75	11.18	4.52	5.58	8.07	16.05
20x5	ta008	0.50	1.13	1.58	4.45	0.08	0.94	1.24	4.06	5.14	5.75	6.38	5.95	4.73	5.02	6.14	5.08
20x5	ta009	4.15	4.81	5.37	4.96	3.74	5.11	5.53	6.59	4.63	5.59	7.07	10.77	4.88	5.62	6.26	5.65
20x5	ta010	2.71	4.04	4.78	8.43	2.89	4.26	4.78	8.44	6.77	6.77	6.77	0.00	6.77	6.77	6.77	0.00
20x10	ta011	2.97	3.82	4.42	7.21	3.03	3.76	4.17	6.90	8.22	8.87	9.36	6.93	4.68	5.92	7.65	15.92
20x10	ta012	5.18	6.36	7.05	11.21	4.58	6.05	7.35	14.86	6.87	9.20	12.06	22.78	6.63	7.63	8.92	13.89
20x10	ta013	2.61	3.48	4.34	8.14	2.74	3.32	3.74	4.74	8.22	9.77	11.16	16.11	5.95	7.93	9.43	15.36
20x10	ta014	3.63	4.72	5.66	9.52	3.63	4.63	5.44	8.08	7.40	8.78	12.41	19.01	6.02	6.80	8.27	9.49
20x10	ta015	7.05	8.10	8.74	8.91	6.84	7.74	8.74	8.89	12.97	12.97	12.97	0.00	10.29	11.63	13.74	17.29
20x10	ta016	3.01	3.47	3.94	4.17	2.72	3.25	4.08	6.15	4.08	4.67	5.30	4.99	3.51	4.12	5.15	6.79
20x10	ta017	3.17	4.54	5.53	11.84	3.17	3.73	4.38	8.18	9.23	10.65	11.86	11.45	7.75	9.47	10.92	18.97
20x10	ta018	2.08	2.89	3.51	7.86	2.08	3.00	3.58	6.30	6.89	8.63	10.40	18.80	6.83	8.28	9.95	15.65
20x10	ta019	5.78	6.53	7.41	7.77	5.15	6.35	7.09	8.89	7.91	8.50	9.29	7.83	7.47	7.87	8.85	5.79
20x10	ta020	4.59	5.19	5.97	7.60	4.90	5.52	6.10	6.18	5.09	7.55	9.55	17.58	5.15	7.06	8.55	19.06
20x20	ta021	5.57	6.88	7.88	17.98	5.75	6.60	7.18	10.18	6.18	7.00	8.58	15.35	5.53	6.06	6.70	7.76
20x20	ta022	3.62	4.08	4.52	6.86	2.00	3.62	4.90	16.78	3.24	4.78	6.43	21.97	4.48	5.00	5.57	6.37
20x20	ta023	3.22	4.20	4.90	10.94	3.18	4.22	5.16	13.50	5.07	6.27	7.44	14.65	2.92	3.82	4.39	10.27
20x20	ta024	3.51	3.99	4.32	5.89	2.83	3.61	4.27	11.53	4.50	6.14	8.23	30.09	2.07	3.67	5.04	21.60
20x20	ta025	3.49	4.18	4.71	8.12	2.97	4.28	5.11	13.78	5.72	7.93	9.21	27.33	3.36	4.54	6.29	23.44
20x20	ta026	3.37	3.83	4.40	7.63	2.88	3.79	4.36	10.24	5.12	6.47	7.41	15.90	4.90	5.92	6.51	10.57
20x20	ta027	3.30	4.03	4.22	6.10	3.56	3.90	4.14	4.30	3.61	4.48	5.37	12.77	3.74	4.40	5.19	14.04
20x20	ta028	3.50	3.88	4.32	6.48	3.55	3.92	4.59	7.18	5.55	6.63	7.27	11.55	4.50	5.75	6.77	15.25
20x20	ta029	4.83	5.91	6.75	13.54	5.01	6.01	6.57	10.56	6.08	8.31	10.10	26.56	5.50	6.67	7.91	18.08
20x20	ta030	4.09	4.47	4.78	4.88	4.04	4.70	5.97	12.66	7.94	9.83	13.91	34.94	5.88	7.36	8.82	21.96
Ortalama=		3.28	4.03	4.64	7.43	3.09	3.94	4.68	8.30	6.04	7.14	8.35	12.82	5.10	6.01	7.07	11.14

Ek G

Tablo G.1.1 AS İz Güncelleme Kuralı Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1325	1325	1325	1359	1365	1343	1343	1325	1359	1325
20x5	ta002	1429	1429	1429	1429	1439	1429	1429	1439	1439	1429
20x5	ta003	1162	1162	1162	1162	1162	1162	1162	1162	1162	1162
20x5	ta004	1421	1421	1421	1421	1421	1421	1421	1421	1435	1421
20x5	ta005	1360	1360	1360	1360	1360	1360	1360	1360	1360	1360
20x5	ta006	1335	1335	1305	1335	1305	1305	1344	1335	1305	1335
20x5	ta007	1365	1360	1360	1360	1360	1365	1360	1360	1360	1365
20x5	ta008	1285	1290	1285	1300	1285	1285	1285	1285	1290	1290
20x5	ta009	1403	1403	1403	1426	1403	1407	1403	1403	1426	1403
20x5	ta010	1194	1208	1208	1208	1194	1194	1194	1194	1194	1194
20x10	ta011	1753	1738	1738	1760	1738	1760	1753	1760	1738	1790
20x10	ta012	1923	1923	1923	1931	1931	1931	1923	1923	1923	1923
20x10	ta013	1695	1695	1717	1713	1695	1713	1695	1695	1695	1717
20x10	ta014	1585	1585	1585	1585	1585	1585	1585	1585	1585	1585
20x10	ta015	1648	1648	1648	1648	1648	1648	1648	1648	1648	1648
20x10	ta016	1499	1499	1499	1527	1499	1499	1499	1499	1527	1499
20x10	ta017	1660	1660	1735	1660	1713	1660	1660	1713	1713	1713
20x10	ta018	1731	1752	1731	1731	1752	1731	1752	1731	1752	1731
20x10	ta019	1836	1805	1805	1805	1805	1805	1805	1824	1805	1805
20x10	ta020	1825	1843	1825	1843	1825	1825	1843	1825	1859	1843
20x20	ta021	2632	2610	2632	2610	2632	2632	2610	2632	2632	2610
20x20	ta022	2261	2306	2306	2261	2306	2261	2261	2261	2306	2261
20x20	ta023	2657	2619	2643	2619	2643	2619	2643	2657	2643	2643
20x20	ta024	2564	2523	2523	2535	2523	2564	2523	2523	2564	2564
20x20	ta025	2596	2596	2665	2652	2652	2652	2596	2652	2596	2652
20x20	ta026	2511	2511	2520	2511	2511	2511	2511	2511	2550	2511
20x20	ta027	2417	2437	2417	2414	2417	2417	2414	2437	2417	2437
20x20	ta028	2435	2404	2404	2404	2404	2404	2404	2404	2406	2404
20x20	ta029	2655	2655	2655	2655	2655	2655	2655	2655	2655	2655
20x20	ta030	2577	2577	2577	2614	2577	2603	2567	2603	2567	2577

Tablo G.1.2 AS_{elite} İz Güncelleme Kuralı Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1325	1359	1325	1343	1343	1325	1325	1325	1325	1325
20x5	ta002	1429	1429	1439	1429	1429	1439	1429	1439	1439	1429
20x5	ta003	1162	1162	1162	1162	1162	1162	1162	1162	1162	1162
20x5	ta004	1421	1435	1421	1435	1435	1421	1421	1421	1435	1421
20x5	ta005	1360	1360	1360	1360	1360	1360	1360	1360	1360	1333
20x5	ta006	1305	1305	1305	1305	1305	1330	1344	1344	1330	1305
20x5	ta007	1356	1356	1356	1356	1356	1342	1356	1356	1356	1356
20x5	ta008	1285	1290	1285	1285	1290	1290	1285	1285	1285	1285
20x5	ta009	1352	1352	1352	1360	1352	1360	1352	1352	1360	1360
20x5	ta010	1208	1208	1194	1219	1219	1194	1208	1194	1219	1208
20x10	ta011	1738	1738	1738	1738	1738	1753	1738	1753	1738	1738
20x10	ta012	1923	1923	1934	1923	1923	1931	1931	1931	1923	1923
20x10	ta013	1704	1695	1711	1704	1695	1695	1695	1695	1695	1711
20x10	ta014	1561	1585	1585	1585	1585	1602	1585	1585	1585	1585
20x10	ta015	1648	1648	1648	1648	1648	1651	1648	1648	1648	1651
20x10	ta016	1531	1531	1499	1499	1499	1499	1527	1499	1527	1527
20x10	ta017	1713	1660	1685	1713	1735	1713	1660	1685	1660	1713
20x10	ta018	1731	1758	1731	1731	1731	1731	1731	1731	1731	1763
20x10	ta019	1770	1770	1770	1824	1824	1770	1836	1770	1770	1770
20x10	ta020	1843	1825	1825	1843	1825	1843	1825	1825	1825	1825
20x20	ta021	2617	2620	2617	2617	2617	2617	2649	2617	2617	2656
20x20	ta022	2261	2261	2261	2261	2306	2261	2261	2331	2306	2261
20x20	ta023	2606	2606	2554	2574	2554	2621	2574	2621	2554	2554
20x20	ta024	2564	2523	2523	2564	2523	2564	2564	2523	2523	2535
20x20	ta025	2596	2596	2652	2665	2596	2596	2652	2596	2596	2596
20x20	ta026	2511	2511	2492	2511	2492	2511	2533	2511	2492	2511
20x20	ta027	2414	2417	2414	2414	2417	2437	2414	2414	2414	2414
20x20	ta028	2421	2404	2404	2404	2406	2404	2435	2421	2404	2404
20x20	ta029	2658	2687	2658	2658	2658	2658	2658	2658	2658	2658
20x20	ta030	2561	2530	2530	2561	2561	2530	2561	2530	2516	2561

Tablo G.1.3 ACS_{ib} İz Güncelleme Kuralı Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1295	1297	1286	1297	1297	1297	1297	1297	1297	1297
20x5	ta002	1367	1367	1368	1368	1367	1368	1368	1367	1368	1367
20x5	ta003	1119	1119	1114	1122	1102	1114	1123	1116	1120	1114
20x5	ta004	1336	1334	1331	1329	1329	1328	1329	1334	1324	1337
20x5	ta005	1277	1298	1291	1313	1273	1311	1261	1313	1314	1277
20x5	ta006	1210	1222	1212	1210	1215	1220	1215	1211	1220	1219
20x5	ta007	1260	1251	1268	1268	1251	1272	1262	1260	1259	1272
20x5	ta008	1222	1223	1218	1219	1224	1225	1225	1225	1216	1223
20x5	ta009	1287	1298	1298	1289	1294	1287	1298	1290	1293	1298
20x5	ta010	1161	1154	1142	1161	1161	1122	1155	1161	1138	1161
20x10	ta011	1645	1637	1638	1651	1646	1650	1634	1645	1644	1648
20x10	ta012	1769	1750	1754	1783	1766	1767	1783	1770	1770	1782
20x10	ta013	1554	1537	1549	1554	1553	1556	1554	1545	1557	1534
20x10	ta014	1440	1425	1441	1445	1452	1449	1462	1456	1430	1427
20x10	ta015	1527	1533	1526	1501	1522	1538	1521	1540	1523	1539
20x10	ta016	1445	1444	1450	1454	1435	1436	1437	1447	1436	1435
20x10	ta017	1544	1531	1566	1538	1542	1553	1548	1546	1551	1542
20x10	ta018	1591	1578	1587	1592	1596	1583	1582	1582	1587	1586
20x10	ta019	1710	1678	1705	1696	1712	1702	1695	1699	1687	1699
20x10	ta020	1669	1679	1685	1675	1673	1679	1681	1668	1691	1671
20x20	ta021	2450	2441	2459	2450	2419	2453	2442	2432	2435	2452
20x20	ta022	2185	2187	2176	2174	2183	2174	2169	2178	2180	2181
20x20	ta023	2432	2424	2412	2421	2418	2424	2441	2417	2430	2434
20x20	ta024	2294	2316	2323	2316	2295	2326	2308	2309	2331	2314
20x20	ta025	2394	2389	2373	2403	2386	2384	2381	2374	2403	2387
20x20	ta026	2302	2315	2301	2313	2308	2302	2303	2302	2302	2311
20x20	ta027	2366	2366	2360	2351	2363	2366	2366	2366	2366	2368
20x20	ta028	2298	2297	2260	2287	2283	2292	2291	2282	2296	2280
20x20	ta029	2367	2375	2335	2380	2353	2385	2385	2377	2383	2363
20x20	ta030	2273	2274	2278	2280	2256	2287	2276	2272	2276	2287

Tablo G.1.4 ACS_{gb} İz Güncelleme Kuralı Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1324	1314	1324	1322	1299	1313	1324	1297	1298	1316
20x5	ta002	1391	1389	1397	1397	1397	1399	1389	1397	1389	1397
20x5	ta003	1162	1162	1162	1162	1162	1162	1162	1162	1162	1162
20x5	ta004	1366	1421	1421	1421	1421	1421	1421	1421	1382	1421
20x5	ta005	1323	1314	1335	1323	1340	1327	1338	1338	1333	1338
20x5	ta006	1305	1305	1305	1305	1305	1305	1305	1305	1234	1305
20x5	ta007	1345	1345	1342	1345	1353	1345	1345	1342	1323	1345
20x5	ta008	1274	1274	1274	1255	1274	1274	1274	1283	1274	1274
20x5	ta009	1311	1311	1352	1352	1305	1305	1335	1305	1360	1311
20x5	ta010	1183	1185	1174	1183	1185	1183	1174	1183	1174	1183
20x10	ta011	1720	1720	1720	1715	1715	1720	1720	1720	1720	1720
20x10	ta012	1808	1823	1876	1854	1819	1918	1789	1830	1859	1830
20x10	ta013	1585	1631	1668	1602	1602	1634	1662	1631	1585	1585
20x10	ta014	1526	1526	1549	1543	1551	1544	1526	1526	1526	1514
20x10	ta015	1648	1648	1648	1648	1648	1648	1648	1648	1648	1635
20x10	ta016	1487	1487	1487	1467	1466	1487	1487	1487	1487	1487
20x10	ta017	1540	1575	1586	1597	1592	1607	1531	1608	1582	1585
20x10	ta018	1644	1634	1702	1702	1702	1692	1702	1702	1640	1725
20x10	ta019	1764	1786	1764	1764	1764	1764	1764	1764	1764	1804
20x10	ta020	1738	1717	1775	1723	1711	1753	1723	1745	1738	1735
20x20	ta021	2545	2554	2541	2531	2516	2592	2577	2531	2496	2543
20x20	ta022	2261	2261	2290	2286	2261	2261	2261	2286	2261	2277
20x20	ta023	2504	2503	2513	2532	2451	2524	2533	2504	2511	2524
20x20	ta024	2391	2449	2347	2385	2411	2447	2402	2388	2375	2355
20x20	ta025	2503	2499	2508	2505	2427	2517	2484	2496	2510	2517
20x20	ta026	2383	2354	2369	2312	2402	2360	2426	2389	2370	2397
20x20	ta027	2414	2414	2414	2414	2414	2424	2414	2414	2414	2414
20x20	ta028	2400	2368	2341	2343	2331	2309	2309	2331	2308	2331
20x20	ta029	2558	2542	2573	2540	2569	2573	2573	2556	2556	2573
20x20	ta030	2503	2503	2348	2503	2462	2442	2402	2503	2478	2384

Tablo G.1.5 ANT-Q İz Güncelleme Kuralı Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1297	1303	1297	1297	1297	1305	1297	1297	1297	1305
20x5	ta002	1372	1390	1383	1378	1378	1373	1373	1372	1373	1373
20x5	ta003	1132	1147	1133	1141	1127	1116	1162	1138	1171	1136
20x5	ta004	1380	1364	1388	1374	1400	1376	1354	1355	1366	1371
20x5	ta005	1299	1283	1305	1293	1299	1338	1277	1311	1331	1304
20x5	ta006	1231	1247	1228	1264	1219	1242	1234	1231	1225	1226
20x5	ta007	1278	1286	1292	1302	1271	1276	1280	1290	1293	1279
20x5	ta008	1271	1256	1261	1255	1254	1282	1260	1257	1258	1248
20x5	ta009	1298	1320	1274	1287	1322	1289	1287	1319	1298	1310
20x5	ta010	1181	1186	1174	1191	1193	1161	1161	1171	1178	1171
20x10	ta011	1709	1696	1694	1758	1686	1667	1733	1686	1714	1684
20x10	ta012	1795	1760	1786	1799	1762	1775	1768	1755	1766	1773
20x10	ta013	1624	1628	1635	1604	1599	1623	1560	1629	1613	1612
20x10	ta014	1479	1502	1462	1477	1463	1464	1466	1519	1489	1502
20x10	ta015	1528	1570	1556	1556	1553	1552	1515	1568	1567	1577
20x10	ta016	1482	1476	1491	1459	1509	1495	1484	1496	1500	1446
20x10	ta017	1608	1590	1601	1599	1581	1594	1558	1569	1588	1592
20x10	ta018	1712	1655	1681	1670	1645	1690	1721	1687	1700	1663
20x10	ta019	1680	1712	1737	1670	1698	1692	1644	1710	1734	1675
20x10	ta020	1727	1713	1701	1728	1702	1720	1692	1686	1701	1741
20x20	ta021	2480	2453	2490	2448	2446	2475	2520	2481	2465	2423
20x20	ta022	2240	2233	2213	2228	2237	2235	2258	2258	2200	2210
20x20	ta023	2446	2487	2486	2446	2458	2459	2476	2384	2491	2456
20x20	ta024	2360	2367	2408	2418	2396	2388	2310	2355	2411	2416
20x20	ta025	2437	2486	2459	2505	2503	2517	2495	2440	2438	2458
20x20	ta026	2343	2388	2351	2335	2379	2355	2342	2342	2342	2349
20x20	ta027	2426	2393	2399	2376	2389	2383	2417	2421	2401	2413
20x20	ta028	2283	2344	2401	2352	2388	2359	2354	2383	2372	2383
20x20	ta029	2455	2453	2375	2414	2420	2410	2388	2420	2433	2442
20x20	ta030	2382	2401	2365	2383	2386	2344	2396	2380	2365	2346

Tablo G.1.6 *MMAS* İz Güncelleme Kuralı Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1324	1324	1324	1324	1324	1324	1324	1324	1324	1324
20x5	ta002	1396	1396	1400	1400	1400	1396	1400	1410	1400	1400
20x5	ta003	1162	1162	1162	1162	1162	1162	1162	1162	1162	1162
20x5	ta004	1435	1421	1421	1421	1435	1435	1421	1421	1435	1421
20x5	ta005	1338	1351	1360	1338	1351	1360	1338	1360	1341	1333
20x5	ta006	1305	1305	1313	1305	1305	1305	1305	1305	1305	1305
20x5	ta007	1342	1317	1356	1356	1342	1356	1356	1318	1356	1356
20x5	ta008	1285	1290	1285	1285	1285	1290	1285	1285	1285	1285
20x5	ta009	1335	1383	1377	1377	1392	1377	1384	1383	1392	1387
20x5	ta010	1194	1183	1194	1183	1183	1194	1183	1194	1194	1194
20x10	ta011	1720	1720	1720	1742	1742	1720	1720	1720	1720	1720
20x10	ta012	1913	1912	1881	1912	1912	1912	1931	1913	1931	1913
20x10	ta013	1695	1695	1695	1695	1695	1705	1704	1717	1695	1677
20x10	ta014	1585	1559	1535	1535	1535	1585	1585	1556	1559	1535
20x10	ta015	1648	1648	1651	1648	1648	1648	1637	1635	1648	1672
20x10	ta016	1527	1527	1500	1499	1527	1499	1499	1499	1527	1531
20x10	ta017	1658	1660	1658	1656	1658	1658	1658	1658	1660	1660
20x10	ta018	1702	1702	1758	1702	1702	1702	1739	1702	1702	1702
20x10	ta019	1771	1786	1771	1786	1771	1786	1786	1771	1771	1797
20x10	ta020	1808	1808	1792	1817	1790	1817	1817	1818	1792	1808
20x20	ta021	2612	2657	2703	2657	2700	2521	2703	2665	2700	2612
20x20	ta022	2306	2261	2306	2261	2261	2261	2261	2306	2261	2261
20x20	ta023	2547	2547	2547	2576	2530	2610	2549	2539	2561	2547
20x20	ta024	2514	2532	2500	2514	2500	2518	2514	2523	2514	2523
20x20	ta025	2574	2566	2566	2566	2566	2566	2574	2596	2618	2580
20x20	ta026	2475	2480	2509	2451	2445	2451	2451	2451	2464	2464
20x20	ta027	2417	2417	2417	2414	2414	2417	2414	2414	2414	2414
20x20	ta028	2404	2404	2404	2404	2404	2404	2393	2404	2404	2393
20x20	ta029	2573	2528	2573	2526	2593	2573	2593	2573	2573	2546
20x20	ta030	2561	2544	2561	2561	2561	2556	2544	2561	2488	2561

Tablo G.1.7 AS_{rank} Güncelleme Kuralı Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1343	1325	1325	1343	1325	1325	1343	1343	1325	1325
20x5	ta002	1429	1439	1439	1439	1429	1439	1429	1429	1429	1429
20x5	ta003	1162	1162	1162	1162	1162	1162	1162	1162	1162	1162
20x5	ta004	1435	1421	1421	1421	1421	1435	1421	1421	1458	1435
20x5	ta005	1351	1360	1360	1360	1360	1360	1360	1360	1351	1357
20x5	ta006	1305	1305	1330	1305	1305	1330	1305	1305	1305	1305
20x5	ta007	1307	1356	1356	1356	1356	1356	1342	1356	1342	1356
20x5	ta008	1285	1285	1285	1313	1285	1285	1285	1285	1290	1285
20x5	ta009	1426	1352	1352	1352	1352	1352	1352	1352	1352	1352
20x5	ta010	1219	1194	1208	1194	1220	1194	1194	1194	1194	1194
20x10	ta011	1738	1738	1753	1753	1738	1753	1760	1753	1738	1738
20x10	ta012	1923	1923	1931	1931	1923	1931	1923	1923	1923	1923
20x10	ta013	1695	1711	1711	1695	1695	1695	1695	1695	1695	1695
20x10	ta014	1602	1585	1561	1585	1585	1585	1585	1585	1585	1561
20x10	ta015	1651	1648	1651	1651	1648	1648	1635	1648	1593	1648
20x10	ta016	1499	1499	1499	1499	1527	1499	1527	1527	1499	1499
20x10	ta017	1685	1660	1660	1660	1660	1660	1660	1660	1660	1713
20x10	ta018	1752	1752	1731	1731	1731	1752	1731	1731	1731	1731
20x10	ta019	1824	1770	1824	1836	1770	1770	1824	1770	1784	1824
20x10	ta020	1843	1825	1825	1825	1825	1825	1825	1825	1859	1843
20x20	ta021	2620	2563	2620	2617	2656	2617	2628	2656	2620	2620
20x20	ta022	2335	2306	2261	2261	2261	2306	2261	2261	2261	2306
20x20	ta023	2574	2574	2554	2652	2574	2554	2621	2574	2650	2554
20x20	ta024	2523	2523	2523	2523	2535	2523	2523	2523	2523	2523
20x20	ta025	2596	2596	2652	2596	2650	2596	2596	2635	2596	2596
20x20	ta026	2492	2511	2511	2511	2511	2511	2511	2511	2510	2511
20x20	ta027	2414	2417	2414	2414	2414	2414	2414	2414	2417	2414
20x20	ta028	2421	2404	2404	2404	2404	2406	2404	2404	2404	2404
20x20	ta029	2658	2568	2658	2568	2687	2568	2658	2568	2687	2658
20x20	ta030	2561	2530	2561	2561	2614	2530	2561	2561	2561	2530

Tablo G.2 İz Güncelleme Kuralları (AS, AS_{elite}, ACS_{ib}, ACS_{gb}) Bağlı Hata Sonuçları

$n \times m$	Problem No	AS-PGK				AS _{elite} -PGK				ACS _{ib} -PGK				ACS _{gb} -PGK			
		Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}
20x5	ta001	3.68	4.80	6.81	16.62	3.68	4.23	6.34	12.08	0.63	1.38	1.49	3.47	1.49	2.75	3.60	11.21
20x5	ta002	5.15	5.37	5.89	4.83	5.15	5.45	5.89	5.16	0.59	0.63	0.66	0.53	2.21	2.59	2.94	4.13
20x5	ta003	7.49	7.49	7.49	0.00	7.49	7.49	7.49	0.00	1.94	3.27	3.89	6.02	7.49	7.49	7.49	0.00
20x5	ta004	9.90	10.01	10.98	4.43	9.90	10.33	10.98	7.23	2.40	2.95	3.40	4.07	5.65	9.17	9.90	20.17
20x5	ta005	10.12	10.12	10.12	0.00	7.94	9.90	10.12	8.54	2.11	4.68	6.40	19.80	6.40	7.77	8.50	8.70
20x5	ta006	9.21	10.79	12.47	16.50	9.21	10.28	12.47	17.17	1.26	1.71	2.26	4.58	3.26	8.61	9.21	22.45
20x5	ta007	9.77	9.89	10.17	2.42	8.31	9.33	9.44	4.43	0.97	1.88	2.66	7.67	6.78	8.39	9.20	7.64
20x5	ta008	6.55	6.80	7.79	4.83	6.55	6.67	6.97	2.42	0.83	1.33	1.58	3.23	4.06	5.56	6.38	6.93
20x5	ta009	14.07	14.47	15.93	9.57	9.92	10.18	10.57	4.13	4.63	5.14	5.53	4.69	6.10	7.70	10.57	22.53
20x5	ta010	7.76	8.14	9.03	6.76	7.76	8.94	10.02	10.24	1.26	3.94	4.78	13.38	5.96	6.56	6.95	4.69
20x10	ta011	9.86	10.80	13.15	16.38	9.86	10.05	10.81	6.32	3.29	3.91	4.36	5.69	8.41	8.66	8.72	2.11
20x10	ta012	15.91	16.06	16.40	3.86	15.91	16.12	16.58	4.60	5.49	6.65	7.47	11.35	7.84	10.95	15.61	37.25
20x10	ta013	13.30	13.84	14.77	10.41	13.30	13.64	14.37	6.86	2.54	3.56	4.08	8.08	5.95	8.19	11.50	31.28
20x10	ta014	15.02	15.02	15.02	0.00	13.28	14.97	16.26	9.78	3.41	4.70	6.10	12.53	9.87	11.26	12.55	12.50
20x10	ta015	16.14	16.14	16.14	0.00	16.14	16.18	16.35	1.26	5.78	7.61	8.53	11.66	15.22	16.05	16.14	4.11
20x10	ta016	7.30	7.70	9.31	11.81	7.30	8.36	9.59	15.67	2.72	3.21	4.08	7.00	4.94	6.15	6.44	8.65
20x10	ta017	11.86	13.79	16.91	30.96	11.86	14.13	16.91	27.39	3.17	4.18	5.53	9.45	3.17	6.49	8.36	25.90
20x10	ta018	12.55	13.09	13.91	10.84	12.55	12.93	14.63	12.49	2.60	3.15	3.77	5.44	6.24	9.53	12.16	32.32
20x10	ta019	13.31	13.62	15.25	10.91	11.11	12.20	15.25	28.21	5.34	6.61	7.47	10.22	10.73	11.12	13.25	13.74
20x10	ta020	14.71	15.37	16.84	12.15	14.71	15.05	15.84	8.69	4.84	5.41	6.29	7.34	7.54	9.10	11.57	18.93
20x20	ta021	13.63	14.20	14.58	11.36	13.93	14.25	15.63	14.93	5.31	6.37	7.05	12.02	8.66	10.69	12.84	27.71
20x20	ta022	7.67	8.52	9.81	23.24	7.67	8.43	11.00	26.65	3.29	3.75	4.14	5.58	7.67	8.12	9.05	12.67
20x20	ta023	12.60	13.44	14.23	14.63	9.80	11.00	12.68	28.79	3.70	4.27	4.94	8.88	5.37	7.91	8.90	23.61
20x20	ta024	13.50	14.29	15.34	20.47	13.50	14.29	15.34	20.47	3.19	4.06	4.86	12.19	5.58	7.74	10.17	33.98
20x20	ta025	13.31	14.84	16.32	30.30	13.31	14.10	16.32	29.36	3.58	4.21	4.89	10.41	5.94	8.97	9.86	26.38
20x20	ta026	12.80	13.02	14.56	12.35	11.95	12.65	13.79	12.67	3.37	3.59	4.00	5.34	3.86	6.75	8.98	31.19
20x20	ta027	6.20	6.57	7.22	10.15	6.20	6.33	7.22	7.17	3.43	3.99	4.18	5.01	6.20	6.25	6.64	3.16
20x20	ta028	9.27	9.42	10.68	9.75	9.27	9.58	10.68	11.03	2.73	3.94	4.45	11.35	4.91	6.23	9.09	28.87
20x20	ta029	18.69	18.69	18.69	0.00	14.80	18.55	20.12	30.86	4.38	5.96	6.62	16.21	13.54	14.50	15.02	12.91
20x20	ta030	17.86	18.64	20.02	16.48	15.52	16.81	17.58	18.30	3.58	4.49	5.00	8.76	7.81	12.62	14.92	56.97
Ortalama=		11.31	11.83	12.86	10.73	10.60	11.41	12.57	13.10	3.08	4.02	4.68	8.40	6.63	8.46	9.88	18.42

Tablo G.3 İz Güncelleme Kuralları (ANTQ, MMAS, AS_{rank}) Bağlı Hata Sonuçları

$n \times m$	Problem No	ANTQ-PGK				MMAS-PGK				AS _{rank} -PGK			
		Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}
20x5	ta001	1.49	1.66	2.11	3.58	3.60	3.60	3.60	0.00	3.68	4.24	5.09	9.30
20x5	ta002	0.96	1.29	2.28	5.95	2.72	3.00	3.75	4.05	5.15	5.45	5.89	5.16
20x5	ta003	3.24	5.49	8.33	16.23	7.49	7.49	7.49	0.00	7.49	7.49	7.49	0.00
20x5	ta004	4.72	6.17	8.28	14.27	9.90	10.33	10.98	7.23	9.90	10.51	12.76	12.17
20x5	ta005	3.40	5.59	8.34	19.08	7.94	9.07	10.12	10.61	9.39	9.95	10.12	3.75
20x5	ta006	2.01	3.32	5.77	13.13	9.21	9.27	9.87	2.53	9.21	9.62	11.30	10.54
20x5	ta007	2.58	3.69	5.08	9.51	6.30	8.60	9.44	15.83	5.49	8.82	9.44	15.64
20x5	ta008	3.48	4.49	6.30	9.66	6.55	6.63	6.97	2.11	6.55	6.82	8.87	8.82
20x5	ta009	3.58	5.72	7.48	16.62	8.54	12.09	13.17	16.35	9.92	10.52	15.93	23.40
20x5	ta010	4.78	6.20	7.67	11.26	6.77	7.36	7.76	5.68	7.76	8.35	10.11	10.93
20x10	ta011	5.37	7.63	11.13	26.76	8.72	9.00	10.11	9.28	9.86	10.38	11.25	8.89
20x10	ta012	5.79	6.93	8.44	14.96	13.38	15.31	16.40	13.65	15.91	16.06	16.40	3.86
20x10	ta013	4.28	7.80	9.29	21.79	12.10	13.46	14.77	10.20	13.30	13.52	14.37	6.75
20x10	ta014	6.10	7.57	10.23	19.98	11.39	12.98	15.02	21.85	13.28	14.80	16.26	12.22
20x10	ta015	6.77	9.53	11.13	19.32	15.22	16.16	17.83	9.86	12.26	15.72	16.35	17.87
20x10	ta016	3.51	6.21	8.02	19.24	7.30	8.34	9.59	15.12	7.30	7.90	9.31	13.53
20x10	ta017	4.99	7.01	8.36	15.11	11.59	11.75	11.86	1.26	11.86	12.39	15.43	17.72
20x10	ta018	6.96	9.39	11.90	24.58	10.66	11.27	14.30	20.11	12.55	12.96	13.91	10.14
20x10	ta019	3.20	6.42	9.04	29.19	11.17	11.71	12.81	9.64	11.11	12.97	15.25	28.78
20x10	ta020	5.97	7.55	9.43	17.63	12.51	13.56	14.27	11.36	14.71	15.15	16.84	12.08
20x20	ta021	5.49	7.45	9.71	27.27	9.75	15.50	17.68	58.10	11.58	14.14	15.63	25.59
20x20	ta022	4.76	6.25	7.52	19.23	7.67	8.31	9.81	21.74	7.67	8.66	11.19	28.25
20x20	ta023	2.49	5.71	7.09	31.27	8.77	9.86	12.21	22.79	9.80	11.27	14.02	38.41
20x20	ta024	3.91	7.19	8.77	34.66	12.46	13.14	13.90	9.89	13.50	13.55	14.04	3.79
20x20	ta025	6.37	7.98	9.86	30.80	12.00	12.49	14.27	17.24	13.31	13.96	15.76	24.39
20x20	ta026	4.90	5.69	7.28	17.35	9.84	10.70	12.71	19.55	11.95	12.71	12.80	5.98
20x20	ta027	4.53	5.67	6.73	16.93	6.20	6.26	6.34	1.55	6.20	6.23	6.34	1.26
20x20	ta028	3.77	7.36	9.14	33.19	8.77	9.17	9.27	4.64	9.27	9.36	10.05	5.34
20x20	ta029	6.17	8.23	9.75	26.12	12.92	14.67	15.91	23.90	14.80	17.47	20.12	52.66
20x20	ta030	7.62	9.04	10.24	19.37	14.23	17.07	17.58	22.80	16.16	17.40	20.02	24.79
Ortalama=		4.44	6.34	8.16	19.47	9.52	10.61	11.66	12.96	10.36	11.28	12.74	14.73

Ek H

Tablo H.1.1 TSPT Sezgisel Bilgisi Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1303	1304	1303	1305	1297	1314	1300	1304	1300	1324
20x5	ta002	1367	1372	1377	1370	1373	1371	1372	1372	1369	1374
20x5	ta003	1157	1182	1182	1163	1183	1174	1155	1169	1132	1191
20x5	ta004	1398	1415	1384	1383	1366	1368	1374	1413	1406	1371
20x5	ta005	1287	1299	1326	1320	1283	1314	1316	1304	1316	1288
20x5	ta006	1294	1302	1289	1286	1295	1301	1275	1285	1277	1307
20x5	ta007	1274	1260	1260	1262	1256	1262	1259	1283	1283	1270
20x5	ta008	1288	1305	1286	1311	1283	1294	1283	1298	1304	1284
20x5	ta009	1289	1316	1297	1340	1320	1332	1317	1340	1325	1348
20x5	ta010	1197	1183	1203	1198	1189	1186	1199	1198	1206	1208
20x10	ta011	1717	1717	1729	1702	1689	1709	1710	1708	1722	1699
20x10	ta012	1800	1798	1810	1809	1809	1807	1799	1798	1810	1792
20x10	ta013	1684	1660	1665	1668	1683	1648	1654	1670	1636	1660
20x10	ta014	1490	1524	1529	1530	1534	1547	1520	1517	1529	1528
20x10	ta015	1589	1563	1562	1602	1578	1603	1595	1582	1556	1571
20x10	ta016	1562	1575	1546	1565	1588	1563	1586	1576	1515	1561
20x10	ta017	1635	1593	1624	1624	1624	1627	1621	1593	1623	1632
20x10	ta018	1631	1643	1637	1652	1662	1633	1649	1676	1675	1641
20x10	ta019	1689	1701	1700	1688	1683	1680	1692	1681	1700	1710
20x10	ta020	1670	1715	1677	1696	1667	1657	1689	1683	1667	1697
20x20	ta021	2366	2404	2426	2424	2392	2427	2386	2403	2401	2402
20x20	ta022	2236	2285	2245	2272	2273	2282	2249	2287	2292	2265
20x20	ta023	2463	2491	2476	2457	2488	2489	2479	2481	2477	2472
20x20	ta024	2341	2346	2348	2363	2367	2365	2344	2331	2337	2343
20x20	ta025	2426	2435	2465	2456	2418	2424	2464	2433	2412	2408
20x20	ta026	2409	2375	2390	2351	2404	2356	2328	2372	2394	2386
20x20	ta027	2437	2405	2422	2390	2440	2408	2451	2419	2437	2451
20x20	ta028	2349	2381	2352	2352	2364	2333	2350	2349	2345	2352
20x20	ta029	2440	2424	2410	2422	2457	2446	2412	2443	2430	2412
20x20	ta030	2326	2359	2329	2368	2347	2339	2358	2353	2340	2340

Tablo H.1.2 GJI Sezgisel Bilgisi Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1379	1338	1372	1379	1355	1324	1318	1347	1379	1379
20x5	ta002	1368	1368	1368	1368	1366	1368	1371	1368	1368	1368
20x5	ta003	1182	1195	1182	1148	1208	1205	1188	1208	1195	1162
20x5	ta004	1420	1420	1437	1469	1399	1437	1437	1437	1420	1469
20x5	ta005	1265	1262	1262	1262	1262	1275	1263	1305	1262	1265
20x5	ta006	1271	1268	1272	1258	1268	1270	1268	1259	1265	1263
20x5	ta007	1259	1252	1271	1256	1275	1294	1265	1274	1268	1259
20x5	ta008	1319	1271	1319	1303	1296	1282	1271	1289	1303	1289
20x5	ta009	1319	1300	1309	1316	1317	1304	1341	1309	1299	1316
20x5	ta010	1191	1134	1191	1169	1176	1187	1169	1187	1181	1187
20x10	ta011	1717	1720	1711	1770	1745	1757	1715	1814	1781	1735
20x10	ta012	1800	1813	1797	1792	1810	1791	1770	1796	1798	1804
20x10	ta013	1669	1641	1647	1648	1712	1636	1635	1709	1711	1671
20x10	ta014	1494	1532	1547	1540	1545	1504	1506	1507	1519	1543
20x10	ta015	1581	1591	1646	1637	1601	1632	1620	1650	1637	1604
20x10	ta016	1493	1459	1478	1478	1479	1477	1478	1469	1469	1449
20x10	ta017	1607	1620	1612	1601	1621	1611	1629	1634	1612	1606
20x10	ta018	1670	1640	1713	1708	1695	1666	1686	1652	1671	1685
20x10	ta019	1706	1700	1718	1692	1713	1704	1690	1720	1696	1715
20x10	ta020	1712	1748	1740	1751	1757	1708	1747	1711	1717	1741
20x20	ta021	2445	2450	2450	2474	2459	2481	2457	2444	2460	2392
20x20	ta022	2296	2222	2222	2270	2272	2263	2291	2249	2286	2272
20x20	ta023	2499	2474	2465	2468	2481	2480	2479	2485	2532	2497
20x20	ta024	2364	2405	2362	2405	2381	2406	2365	2379	2436	2412
20x20	ta025	2455	2446	2411	2490	2441	2488	2461	2444	2486	2479
20x20	ta026	2371	2391	2401	2384	2421	2384	2377	2381	2411	2355
20x20	ta027	2413	2426	2412	2389	2410	2406	2402	2424	2401	2411
20x20	ta028	2364	2316	2337	2317	2348	2348	2388	2354	2365	2348
20x20	ta029	2331	2396	2389	2393	2388	2340	2405	2395	2348	2377
20x20	ta030	2381	2321	2391	2379	2367	2376	2349	2349	2337	2361

Tablo H.1.3 SPIRIT Sezgisel Bilgisi Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1324	1305	1300	1293	1322	1292	1297	1324	1324	1302
20x5	ta002	1373	1371	1373	1367	1373	1371	1373	1371	1370	1368
20x5	ta003	1164	1168	1149	1156	1156	1146	1160	1168	1152	1168
20x5	ta004	1369	1336	1367	1335	1368	1381	1342	1354	1357	1336
20x5	ta005	1273	1273	1273	1254	1268	1278	1280	1267	1280	1278
20x5	ta006	1274	1243	1270	1260	1265	1264	1258	1247	1247	1259
20x5	ta007	1277	1267	1262	1259	1271	1266	1263	1277	1261	1273
20x5	ta008	1247	1275	1247	1251	1253	1261	1267	1252	1256	1250
20x5	ta009	1290	1290	1300	1307	1278	1298	1293	1283	1296	1284
20x5	ta010	1170	1150	1170	1159	1170	1166	1163	1166	1163	1161
20x10	ta011	1722	1704	1738	1744	1708	1721	1697	1731	1711	1713
20x10	ta012	1777	1759	1759	1781	1760	1721	1735	1732	1750	1744
20x10	ta013	1608	1610	1601	1560	1590	1597	1607	1614	1599	1594
20x10	ta014	1466	1463	1455	1485	1464	1504	1482	1474	1456	1469
20x10	ta015	1521	1533	1511	1504	1521	1506	1519	1548	1504	1538
20x10	ta016	1494	1501	1512	1493	1491	1495	1495	1511	1517	1519
20x10	ta017	1584	1591	1566	1590	1597	1583	1553	1574	1595	1587
20x10	ta018	1642	1637	1627	1598	1598	1632	1647	1598	1598	1647
20x10	ta019	1664	1664	1648	1648	1664	1664	1664	1664	1661	1660
20x10	ta020	1674	1677	1690	1689	1688	1684	1688	1675	1639	1682
20x20	ta021	2398	2417	2394	2402	2394	2410	2380	2384	2378	2403
20x20	ta022	2234	2265	2259	2258	2232	2231	2230	2215	2242	2253
20x20	ta023	2450	2440	2417	2386	2408	2392	2405	2422	2385	2414
20x20	ta024	2340	2409	2345	2333	2335	2340	2346	2338	2334	2309
20x20	ta025	2458	2428	2433	2437	2419	2474	2430	2416	2439	2459
20x20	ta026	2371	2361	2355	2328	2373	2363	2358	2373	2373	2368
20x20	ta027	2410	2386	2407	2392	2412	2418	2416	2399	2362	2425
20x20	ta028	2282	2319	2319	2303	2305	2338	2308	2313	2327	2322
20x20	ta029	2385	2398	2418	2406	2381	2363	2356	2376	2378	2394
20x20	ta030	2305	2304	2312	2329	2288	2308	2333	2312	2337	2335

Tablo H.1.4 Sezgisel Bilgi Kullanılmadığı Durumda Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1395	1385	1385	1378	1385	1385	1385	1385	1385	1385
20x5	ta002	1452	1475	1395	1407	1515	1489	1439	1407	1481	1481
20x5	ta003	1246	1246	1321	1246	1290	1246	1321	1246	1319	1296
20x5	ta004	1531	1481	1470	1470	1461	1492	1470	1439	1493	1512
20x5	ta005	1366	1371	1371	1366	1366	1366	1366	1366	1366	1366
20x5	ta006	1330	1330	1330	1338	1330	1330	1324	1381	1424	1381
20x5	ta007	1345	1355	1345	1348	1348	1347	1355	1343	1355	1355
20x5	ta008	1377	1320	1342	1315	1348	1323	1365	1367	1387	1349
20x5	ta009	1354	1353	1378	1391	1374	1350	1354	1353	1346	1377
20x5	ta010	1265	1323	1344	1323	1266	1266	1255	1323	1323	1273
20x10	ta011	1844	1816	1816	1821	1817	1824	1845	1879	1799	1764
20x10	ta012	1904	1841	1904	1904	1885	1833	1844	1912	1901	1901
20x10	ta013	1682	1706	1691	1677	1668	1707	1673	1687	1697	1667
20x10	ta014	1615	1604	1637	1582	1637	1650	1623	1626	1583	1626
20x10	ta015	1732	1644	1675	1644	1657	1644	1644	1710	1609	1644
20x10	ta016	1609	1705	1652	1604	1540	1584	1684	1674	1721	1667
20x10	ta017	1697	1663	1744	1744	1699	1744	1729	1679	1744	1663
20x10	ta018	1780	1827	1773	1783	1819	1800	1815	1819	1749	1800
20x10	ta019	1851	1866	1832	1851	1866	1847	1914	1857	1785	1866
20x10	ta020	1886	1886	1886	1886	1886	1886	1747	1868	1886	1868
20x20	ta021	2490	2461	2483	2467	2520	2488	2516	2465	2468	2464
20x20	ta022	2445	2426	2367	2416	2348	2431	2444	2426	2349	2407
20x20	ta023	2534	2567	2400	2535	2501	2535	2590	2538	2559	2461
20x20	ta024	2574	2562	2504	2556	2504	2571	2537	2568	2508	2588
20x20	ta025	2627	2610	2661	2567	2621	2549	2661	2649	2649	2486
20x20	ta026	2400	2501	2484	2458	2479	2466	2481	2462	2421	2456
20x20	ta027	2494	2433	2425	2521	2565	2442	2521	2486	2521	2521
20x20	ta028	2600	2499	2435	2600	2600	2600	2600	2598	2499	2568
20x20	ta029	2563	2577	2560	2570	2555	2544	2544	2577	2515	2536
20x20	ta030	2425	2517	2571	2500	2576	2517	2445	2517	2402	2397

Tablo H.2 Sezgisel Bilgi Kuralları Bağlı Hata Sonuçları

$n \times m$	Problem No	TSPT				GJI				SPIRIT				Sezgisel Bilgi Yok			
		Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}
20x5	ta001	1.49	2.14	3.60	7.92	3.13	6.18	7.90	24.12	1.10	2.37	3.60	13.64	7.82	8.40	9.15	4.06
20x5	ta002	0.59	0.93	1.32	2.75	0.52	0.67	0.88	1.20	0.59	0.88	1.03	2.16	2.65	7.00	11.48	40.79
20x5	ta003	4.72	8.12	10.18	17.60	6.20	9.83	11.75	19.80	6.01	7.19	8.05	8.19	15.26	18.20	22.20	34.91
20x5	ta004	5.65	7.33	9.44	18.82	8.20	10.94	13.61	21.85	3.25	4.76	6.81	16.59	11.29	14.61	18.41	26.34
20x5	ta005	3.89	5.69	7.37	15.36	2.19	2.70	5.67	13.50	1.54	3.03	3.64	7.93	10.61	10.69	11.01	2.11
20x5	ta006	6.69	8.04	9.37	10.64	5.27	5.96	6.44	4.85	4.02	5.33	6.61	10.29	10.79	12.95	19.16	33.69
20x5	ta007	1.37	2.25	3.55	10.02	1.05	2.28	4.44	12.18	1.61	2.31	3.07	6.59	8.39	8.93	9.36	4.88
20x5	ta008	6.38	7.26	8.71	10.36	5.39	7.31	9.37	17.22	3.40	4.14	5.72	9.16	9.04	11.88	15.01	24.79
20x5	ta009	4.80	7.51	9.59	18.89	5.61	6.75	9.02	12.17	3.90	5.03	6.26	8.76	9.43	10.81	13.09	15.44
20x5	ta010	6.77	8.01	9.03	8.33	2.35	6.25	7.49	17.27	3.79	5.04	5.60	6.21	13.27	16.98	21.30	33.65
20x10	ta011	6.76	8.10	9.29	11.71	8.15	10.40	14.66	33.94	7.27	8.65	10.24	15.18	11.50	15.20	18.77	30.28
20x10	ta012	8.02	8.69	9.10	6.51	6.69	8.32	9.28	11.90	3.74	5.59	7.35	19.28	10.49	13.50	15.25	30.91
20x10	ta013	9.36	11.15	12.57	14.80	9.29	11.49	14.44	31.90	4.28	6.82	7.89	15.33	11.43	12.67	14.10	14.67
20x10	ta014	8.13	10.65	12.26	14.70	8.42	10.57	12.26	19.97	5.59	6.81	9.14	15.02	14.80	17.44	19.74	22.67
20x10	ta015	9.65	11.35	12.97	16.99	11.42	14.16	16.28	24.21	5.99	7.15	9.09	15.15	13.39	17.00	22.06	36.21
20x10	ta016	8.45	11.93	13.67	21.25	3.72	5.43	6.87	12.16	6.73	7.57	8.73	10.82	10.24	17.68	23.19	57.72
20x10	ta017	7.35	9.14	10.18	14.65	7.88	8.85	10.11	10.50	4.65	6.60	7.61	13.86	12.06	15.27	17.52	34.40
20x10	ta018	6.05	7.28	8.97	16.33	6.63	9.14	11.38	23.33	3.90	5.49	7.09	21.86	13.72	16.81	18.79	24.88
20x10	ta019	5.46	6.24	7.34	10.01	6.09	7.06	7.97	10.84	3.45	4.21	4.46	6.54	12.05	16.35	20.15	32.23
20x10	ta020	4.15	5.71	7.79	17.65	7.35	8.94	10.43	18.97	3.02	5.51	6.22	15.13	9.81	17.44	18.54	43.34
20x20	ta021	3.00	4.62	5.66	19.24	4.14	6.71	8.01	23.99	3.53	4.31	5.22	12.73	7.14	8.06	9.71	21.54
20x20	ta022	6.48	8.03	9.14	19.39	5.81	7.82	9.33	26.15	5.48	6.76	7.86	16.17	11.81	14.57	16.43	37.45
20x20	ta023	5.63	6.50	7.09	11.07	5.98	6.88	8.86	19.51	2.54	3.69	5.33	21.67	3.18	8.43	11.35	55.58
20x20	ta024	4.86	5.65	6.48	12.37	6.25	7.58	9.58	24.88	3.87	5.39	8.37	25.41	12.64	14.58	16.42	31.71
20x20	ta025	5.11	6.25	7.59	20.89	5.24	7.38	8.69	25.74	5.46	6.47	7.99	18.73	8.51	13.84	16.15	57.31
20x20	ta026	4.58	6.76	8.22	25.47	5.80	7.26	8.76	19.36	4.58	6.12	6.60	13.75	7.82	10.55	12.35	30.25
20x20	ta027	5.15	6.73	7.83	20.59	5.10	6.00	6.73	10.88	3.92	5.71	6.69	18.65	6.69	9.67	12.85	46.18
20x20	ta028	6.05	6.94	8.23	12.53	5.27	6.75	8.55	21.82	3.73	5.16	6.27	15.35	10.68	16.36	18.18	60.16
20x20	ta029	7.73	8.61	9.83	16.33	4.20	6.22	7.51	26.48	5.32	6.64	8.09	19.00	12.43	14.18	15.20	19.66
20x20	ta030	6.80	7.71	8.72	13.57	6.57	8.41	9.78	21.98	5.05	6.35	7.30	16.34	10.06	14.17	18.27	65.64
Ortalama=		5.70	7.18	8.50	14.56	5.66	7.48	9.20	18.76	4.04	5.37	6.60	13.85	10.30	13.47	16.17	32.45

Ek I

Tablo I.1 Aday Liste Büyüklüğünü Belirleme Deneme Sonuçları

<i>cl</i>	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
Problem No=	<i>ta011</i>									
5	5972	6006	6023	5997	5987	5978	5997	6017	5956	6013
10	6009	5988	5954	5967	5973	5995	6011	5972	5989	5995
15	5959	5933	5968	5950	5914	6007	5939	5968	5963	5972
20	5997	5981	6003	5953	5987	5984	5970	5975	5955	5998
25	6007	5973	6021	5987	5988	5996	5981	5985	5989	6004
Problem No=	<i>ta012</i>									
5	5555	5571	5574	5627	5627	5580	5595	5661	5585	5649
10	5565	5632	5598	5659	5560	5564	5562	5574	5566	5558
15	5535	5620	5587	5547	5581	5618	5523	5540	5606	5623
20	5594	5550	5546	5549	5573	5574	5539	5573	5527	5549
25	5565	5558	5542	5603	5575	5554	5563	5532	5592	5604
Problem No=	<i>ta013</i>									
5	5856	5822	5819	5866	5801	5869	5807	5822	5843	5790
10	5867	5840	5826	5892	5799	5826	5834	5858	5867	5839
15	5840	5858	5821	5862	5838	5852	5823	5894	5805	5851
20	5871	5825	5837	5874	5863	5842	5838	5858	5916	5808
25	5863	5856	5861	5835	5896	5860	5867	5852	5862	5846
Problem No=	<i>ta014</i>									
5	6099	6053	6089	6004	6090	6101	6044	6087	5992	6070
10	6049	6029	6088	6089	6015	6010	6031	6068	6007	6039
15	6095	6030	6041	6056	6031	6075	6012	6058	6014	6062
20	6063	6056	6037	6080	6043	6058	6037	6038	6072	6020
25	6006	6037	6059	6060	6038	6087	6009	6045	6016	6058
Problem No=	<i>ta015</i>									
5	5729	5690	5695	5680	5717	5722	5692	5707	5704	5726
10	5650	5707	5695	5672	5680	5673	5712	5673	5687	5706
15	5711	5696	5696	5722	5732	5737	5717	5700	5695	5723
20	5764	5728	5701	5742	5732	5697	5726	5706	5712	5762
25	5713	5724	5724	5709	5700	5720	5730	5729	5726	5696

Tablo I.2 Aday Listenin Kullanımının Etkisini Belirleme Deneme Sonuçları

Problem No	Tekrar No									
	1	2	3	4	5	6	7	8	9	10
<i>cl=n</i>										
<i>ta071</i>	6032	6111	6059	6056	6073	6072	6050	6069	6112	6067
<i>ta072</i>	5695	5649	5693	5679	5683	5676	5670	5651	5674	5630
<i>ta073</i>	5938	5933	5929	5949	5915	5960	5944	5945	5929	5925
<i>ta074</i>	6132	6141	6125	6165	6117	6110	6136	6103	6143	6106
<i>ta075</i>	5788	5779	5803	5772	5807	5771	5814	5791	5785	5783
<i>cl=15</i>										
<i>ta071</i>	5959	5933	5968	5950	5914	6007	5939	5968	5963	5972
<i>ta072</i>	5535	5620	5587	5547	5581	5618	5523	5540	5606	5623
<i>ta073</i>	5840	5858	5821	5862	5838	5852	5823	5894	5805	5851
<i>ta074</i>	6095	6030	6041	6056	6031	6075	6012	6058	6014	6062
<i>ta075</i>	5711	5696	5696	5722	5732	5737	5717	5700	5695	5723

Ek J

Tablo J.1.1 API Yerel Arama Yöntemi Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1286	1278	1279	1278	1292	1286	1286	1295	1286	1279
20x5	ta002	1359	1368	1367	1366	1365	1368	1368	1367	1367	1367
20x5	ta003	1090	1098	1098	1098	1090	1098	1098	1097	1092	1098
20x5	ta004	1304	1304	1306	1306	1306	1306	1306	1306	1304	1305
20x5	ta005	1243	1250	1244	1250	1250	1250	1250	1250	1243	1250
20x5	ta006	1210	1210	1210	1210	1210	1210	1210	1210	1210	1210
20x5	ta007	1251	1247	1251	1251	1249	1251	1251	1251	1251	1251
20x5	ta008	1206	1211	1211	1212	1207	1210	1211	1206	1211	1212
20x5	ta009	1253	1253	1255	1254	1247	1257	1255	1255	1253	1255
20x5	ta010	1113	1127	1109	1127	1110	1108	1108	1112	1109	1112
20x10	ta011	1602	1618	1605	1610	1604	1614	1618	1592	1618	1616
20x10	ta012	1705	1701	1696	1711	1689	1712	1702	1703	1719	1692
20x10	ta013	1516	1518	1514	1521	1522	1527	1524	1527	1532	1531
20x10	ta014	1398	1404	1402	1400	1407	1399	1404	1396	1400	1401
20x10	ta015	1450	1479	1449	1459	1459	1459	1467	1468	1453	1473
20x10	ta016	1412	1421	1409	1419	1420	1422	1412	1418	1424	1413
20x10	ta017	1517	1507	1510	1517	1510	1506	1496	1498	1508	1513
20x10	ta018	1568	1559	1559	1564	1568	1565	1568	1558	1564	1559
20x10	ta019	1640	1649	1640	1635	1636	1624	1649	1632	1639	1621
20x10	ta020	1632	1625	1635	1625	1631	1632	1635	1632	1639	1627
20x20	ta021	2349	2357	2360	2340	2348	2345	2364	2339	2352	2367
20x20	ta022	2140	2143	2155	2132	2139	2149	2135	2144	2144	2147
20x20	ta023	2361	2378	2385	2379	2376	2379	2385	2382	2380	2378
20x20	ta024	2250	2240	2262	2260	2269	2263	2268	2268	2263	2252
20x20	ta025	2355	2337	2324	2345	2348	2348	2353	2346	2344	2332
20x20	ta026	2267	2266	2273	2253	2275	2258	2269	2258	2257	2269
20x20	ta027	2334	2319	2328	2322	2321	2327	2330	2316	2329	2335
20x20	ta028	2226	2227	2243	2234	2234	2240	2232	2238	2233	2243
20x20	ta029	2303	2291	2288	2309	2326	2310	2330	2310	2315	2301
20x20	ta030	2250	2255	2244	2246	2241	2250	2260	2256	2238	2254

Tablo J.1.2 Interchange Yerel Arama Yöntemi Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1278	1278	1279	1278	1278	1278	1278	1278	1278	1278
20x5	ta002	1360	1360	1360	1360	1360	1359	1360	1360	1360	1360
20x5	ta003	1087	1081	1081	1087	1088	1081	1085	1081	1081	1088
20x5	ta004	1297	1299	1299	1297	1297	1293	1293	1298	1293	1299
20x5	ta005	1235	1235	1235	1235	1235	1235	1235	1235	1235	1235
20x5	ta006	1195	1195	1195	1195	1195	1195	1195	1195	1195	1195
20x5	ta007	1239	1239	1239	1239	1239	1239	1239	1239	1239	1239
20x5	ta008	1206	1206	1206	1206	1206	1206	1206	1206	1206	1206
20x5	ta009	1230	1230	1230	1230	1230	1230	1230	1230	1230	1230
20x5	ta010	1108	1108	1108	1108	1108	1108	1108	1108	1108	1108
20x10	ta011	1586	1592	1587	1586	1586	1590	1587	1591	1585	1588
20x10	ta012	1675	1672	1660	1677	1676	1664	1674	1669	1670	1671
20x10	ta013	1508	1512	1508	1508	1509	1508	1517	1511	1510	1510
20x10	ta014	1381	1379	1380	1385	1388	1379	1379	1380	1379	1379
20x10	ta015	1426	1425	1432	1428	1425	1425	1428	1429	1427	1432
20x10	ta016	1401	1406	1397	1408	1397	1403	1407	1397	1398	1401
20x10	ta017	1484	1490	1487	1487	1486	1486	1492	1489	1491	1486
20x10	ta018	1555	1548	1550	1546	1555	1546	1552	1550	1550	1550
20x10	ta019	1608	1593	1612	1602	1602	1597	1601	1605	1597	1608
20x10	ta020	1604	1608	1602	1609	1605	1605	1604	1608	1606	1611
20x20	ta021	2310	2297	2307	2307	2320	2307	2311	2300	2300	2310
20x20	ta022	2109	2104	2110	2106	2106	2103	2106	2118	2116	2108
20x20	ta023	2347	2338	2344	2340	2336	2334	2344	2334	2345	2342
20x20	ta024	2238	2239	2242	2224	2235	2238	2239	2235	2229	2236
20x20	ta025	2301	2301	2304	2309	2309	2296	2304	2311	2294	2311
20x20	ta026	2234	2235	2237	2236	2244	2235	2246	2234	2229	2234
20x20	ta027	2290	2296	2300	2297	2289	2293	2281	2285	2294	2286
20x20	ta028	2209	2217	2217	2213	2212	2202	2218	2210	2212	2206
20x20	ta029	2249	2242	2252	2259	2248	2246	2253	2257	2250	2244
20x20	ta030	2196	2183	2197	2181	2197	2183	2188	2181	2192	2196

Tablo J.1.3 Insert Yerel Arama Yöntemi Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1278	1278	1278	1278	1278	1278	1278	1278	1278	1278
20x5	ta002	1359	1359	1359	1360	1360	1360	1359	1359	1359	1360
20x5	ta003	1081	1081	1081	1081	1087	1081	1081	1081	1081	1081
20x5	ta004	1293	1293	1297	1294	1299	1293	1299	1299	1297	1293
20x5	ta005	1235	1235	1236	1235	1236	1235	1235	1239	1235	1235
20x5	ta006	1195	1195	1195	1195	1195	1195	1195	1195	1195	1195
20x5	ta007	1239	1239	1239	1239	1239	1239	1239	1239	1239	1239
20x5	ta008	1206	1206	1206	1206	1206	1206	1206	1206	1206	1206
20x5	ta009	1230	1230	1230	1230	1230	1230	1230	1230	1230	1230
20x5	ta010	1108	1108	1108	1108	1108	1108	1108	1108	1108	1108
20x10	ta011	1585	1582	1582	1583	1582	1582	1585	1583	1586	1582
20x10	ta012	1674	1668	1672	1678	1674	1672	1674	1678	1678	1673
20x10	ta013	1507	1508	1508	1509	1509	1508	1512	1509	1499	1509
20x10	ta014	1379	1383	1379	1382	1379	1379	1383	1379	1379	1378
20x10	ta015	1426	1419	1425	1419	1425	1425	1427	1425	1429	1429
20x10	ta016	1401	1398	1397	1401	1398	1397	1397	1402	1398	1397
20x10	ta017	1486	1484	1484	1484	1484	1484	1486	1484	1486	1484
20x10	ta018	1550	1544	1546	1544	1544	1545	1549	1545	1545	1538
20x10	ta019	1615	1612	1607	1611	1608	1619	1612	1617	1609	1602
20x10	ta020	1605	1608	1591	1610	1607	1591	1603	1603	1610	1608
20x20	ta021	2300	2306	2305	2317	2303	2317	2307	2302	2312	2314
20x20	ta022	2101	2100	2100	2101	2100	2099	2100	2099	2099	2099
20x20	ta023	2342	2346	2340	2345	2348	2337	2348	2334	2345	2331
20x20	ta024	2224	2231	2225	2231	2223	2229	2230	2223	2231	2235
20x20	ta025	2296	2296	2297	2299	2300	2299	2303	2296	2302	2300
20x20	ta026	2236	2234	2235	2235	2229	2239	2233	2230	2235	2235
20x20	ta027	2273	2281	2273	2283	2273	2281	2279	2282	2278	2288
20x20	ta028	2207	2213	2214	2218	2211	2213	2214	2211	2212	2213
20x20	ta029	2237	2237	2242	2242	2242	2242	2242	2242	2239	2242
20x20	ta030	2191	2193	2181	2194	2195	2194	2194	2199	2186	2185

Tablo J.2 Yerel Arama Yöntemleri Bağlı Hata Sonuçları

$n \times m$	Problem No	API				Interchange				Insert			
		Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}	Δ_{eniyi}	Δ_{ort}	Δ_{enkt}	Δ_{std}
20x5	ta001	0.00	0.51	1.33	5.97	0.00	0.01	0.08	0.32	0.00	0.00	0.00	0.00
20x5	ta002	0.00	0.53	0.66	2.70	0.00	0.07	0.07	0.32	0.00	0.03	0.07	0.52
20x5	ta003	0.83	1.36	1.57	3.53	0.00	0.28	0.65	3.27	0.00	0.06	0.56	1.90
20x5	ta004	0.85	0.95	1.01	0.95	0.00	0.27	0.46	2.55	0.00	0.21	0.46	2.75
20x5	ta005	0.65	1.05	1.21	3.23	0.00	0.00	0.00	0.00	0.00	0.05	0.32	1.26
20x5	ta006	1.26	1.26	1.26	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20x5	ta007	0.65	0.92	0.97	1.35	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20x5	ta008	0.00	0.31	0.50	2.41	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20x5	ta009	1.38	1.93	2.20	2.67	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20x5	ta010	0.00	0.50	1.71	7.32	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20x10	ta011	0.63	1.75	2.28	8.77	0.19	0.37	0.63	2.39	0.00	0.08	0.25	1.55
20x10	ta012	1.81	2.65	3.62	9.29	0.06	0.71	1.08	5.39	0.54	0.91	1.15	3.21
20x10	ta013	1.20	1.82	2.41	6.12	0.80	0.94	1.40	2.81	0.20	0.79	1.07	3.36
20x10	ta014	1.31	1.68	2.10	3.25	0.07	0.21	0.73	3.11	0.00	0.15	0.36	1.89
20x10	ta015	2.11	3.00	4.23	9.95	0.42	0.61	0.92	2.67	0.00	0.42	0.70	3.48
20x10	ta016	0.86	1.43	1.93	5.10	0.00	0.32	0.79	4.33	0.00	0.11	0.36	1.96
20x10	ta017	0.81	1.63	2.22	7.02	0.00	0.26	0.54	2.57	0.00	0.04	0.13	0.97
20x10	ta018	1.30	1.64	1.95	4.13	0.52	0.79	1.11	3.16	0.00	0.46	0.78	3.23
20x10	ta019	1.76	2.73	3.52	9.20	0.00	0.60	1.19	5.87	0.56	1.14	1.63	5.03
20x10	ta020	2.14	2.53	3.02	4.55	0.69	0.96	1.26	2.74	0.00	0.79	1.19	7.09
20x20	ta021	1.83	2.40	3.05	9.69	0.00	0.43	1.00	6.67	0.13	0.49	0.87	6.25
20x20	ta022	1.52	2.04	2.62	6.73	0.14	0.41	0.86	4.93	-0.05	-0.01	0.05	0.79
20x20	ta023	1.50	2.25	2.54	6.77	0.34	0.62	0.90	4.72	0.21	0.67	0.95	5.95
20x20	ta024	0.76	1.64	2.07	9.38	0.04	0.56	0.85	5.32	0.00	0.23	0.54	4.16
20x20	ta025	1.44	2.28	2.79	9.58	0.13	0.57	0.87	6.06	0.22	0.34	0.52	2.53
20x20	ta026	1.21	1.73	2.20	7.49	0.13	0.47	0.90	5.02	0.13	0.36	0.58	2.88
20x20	ta027	1.89	2.34	2.73	6.37	0.35	0.80	1.19	5.97	0.00	0.27	0.66	4.98
20x20	ta028	1.18	1.59	1.95	5.98	0.09	0.53	0.82	5.10	0.32	0.57	0.82	2.80
20x20	ta029	2.28	3.19	4.16	13.45	0.22	0.58	0.98	5.42	0.00	0.17	0.22	2.16
20x20	ta030	2.75	3.28	3.76	7.07	0.14	0.52	0.87	6.95	0.14	0.61	0.96	5.49
Ortalama=		1.20	1.76	2.25	6.00	0.15	0.40	0.67	3.25	0.08	0.30	0.51	2.54

Ek K

Tablo K.1.1 20x5-10-20 Problem Boyutu için Algoritma Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x5	ta001	1278	1278	1278	1278	1278	1278	1278	1278	1278	1278
20x5	ta002	1359	1359	1359	1360	1359	1359	1359	1359	1359	1359
20x5	ta003	1086	1081	1081	1088	1088	1082	1081	1088	1081	1081
20x5	ta004	1297	1299	1296	1299	1293	1293	1297	1293	1297	1293
20x5	ta005	1235	1235	1235	1235	1239	1235	1236	1235	1235	1235
20x5	ta006	1195	1195	1195	1195	1195	1195	1195	1195	1195	1195
20x5	ta007	1239	1239	1239	1239	1239	1239	1239	1239	1239	1239
20x5	ta008	1206	1206	1206	1206	1206	1206	1206	1206	1206	1206
20x5	ta009	1236	1240	1240	1240	1236	1235	1230	1233	1230	1239
20x5	ta010	1108	1108	1108	1108	1108	1108	1108	1108	1108	1108
20x10	ta011	1582	1586	1586	1586	1583	1582	1586	1592	1591	1586
20x10	ta012	1667	1669	1671	1669	1667	1675	1660	1672	1667	1673
20x10	ta013	1509	1507	1508	1510	1507	1507	1508	1512	1508	1507
20x10	ta014	1379	1379	1379	1377	1381	1386	1377	1379	1383	1382
20x10	ta015	1419	1424	1422	1424	1419	1425	1419	1424	1423	1419
20x10	ta016	1410	1400	1401	1405	1405	1401	1407	1407	1406	1407
20x10	ta017	1486	1484	1484	1484	1486	1484	1484	1486	1484	1484
20x10	ta018	1547	1556	1546	1553	1549	1553	1559	1544	1548	1546
20x10	ta019	1594	1602	1596	1608	1608	1605	1608	1594	1603	1607
20x10	ta020	1608	1608	1602	1606	1605	1606	1608	1598	1609	1598
20x20	ta021	2313	2307	2312	2305	2303	2302	2316	2306	2303	2317
20x20	ta022	2099	2112	2099	2105	2110	2100	2111	2100	2100	2101
20x20	ta023	2343	2340	2332	2335	2331	2334	2344	2344	2345	2343
20x20	ta024	2223	2224	2225	2231	2223	2229	2224	2224	2232	2225
20x20	ta025	2296	2299	2298	2298	2294	2294	2306	2296	2301	2296
20x20	ta026	2232	2232	2230	2234	2232	2230	2234	2234	2235	2232
20x20	ta027	2278	2281	2285	2280	2281	2280	2276	2280	2297	2281
20x20	ta028	2206	2204	2200	2208	2210	2217	2214	2200	2213	2208
20x20	ta029	2242	2242	2242	2242	2237	2243	2237	2243	2242	2242
20x20	ta030	2194	2186	2190	2190	2184	2186	2189	2181	2184	2187

Tablo K.1.2 50x5-10-20 Problem Boyutu için Algoritma Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
50x5	ta031	2724	2724	2724	2724	2724	2724	2724	2724	2724	2724
50x5	ta032	2845	2838	2838	2838	2843	2839	2838	2839	2843	2838
50x5	ta033	2621	2621	2621	2621	2622	2621	2621	2622	2621	2621
50x5	ta034	2753	2751	2753	2753	2753	2753	2751	2753	2753	2753
50x5	ta035	2863	2864	2863	2863	2863	2863	2863	2863	2863	2863
50x5	ta036	2831	2831	2831	2831	2831	2829	2831	2831	2831	2831
50x5	ta037	2728	2725	2729	2732	2730	2728	2725	2728	2725	2732
50x5	ta038	2686	2694	2683	2690	2686	2683	2683	2683	2692	2683
50x5	ta039	2552	2557	2557	2559	2555	2559	2554	2554	2557	2559
50x5	ta040	2782	2782	2782	2782	2782	2782	2782	2782	2782	2782
50x10	ta041	3081	3073	3076	3064	3075	3050	3066	3080	3065	3063
50x10	ta042	2951	2939	2947	2954	2960	2941	2961	2950	2948	2950
50x10	ta043	2940	2927	2942	2940	2930	2934	2918	2932	2925	2932
50x10	ta044	3084	3092	3099	3093	3103	3100	3097	3097	3097	3103
50x10	ta045	3063	3054	3060	3045	3058	3062	3062	3047	3061	3037
50x10	ta046	3075	3064	3074	3066	3070	3075	3068	3067	3062	3071
50x10	ta047	3145	3156	3140	3151	3145	3152	3162	3143	3165	3140
50x10	ta048	3065	3070	3065	3075	3070	3073	3072	3059	3065	3072
50x10	ta049	2964	2959	2964	2954	2950	2960	2967	2959	2952	2951
50x10	ta050	3131	3137	3136	3143	3139	3143	3146	3131	3132	3148
50x20	ta051	3976	3978	3979	3972	3983	3973	3971	3951	3961	3956
50x20	ta052	3841	3837	3833	3835	3843	3813	3829	3829	3800	3834
50x20	ta053	3799	3788	3798	3785	3784	3791	3803	3796	3789	3792
50x20	ta054	3862	3853	3849	3812	3825	3844	3849	3858	3855	3860
50x20	ta055	3756	3756	3739	3753	3769	3739	3767	3764	3754	3753
50x20	ta056	3804	3797	3818	3804	3798	3808	3810	3799	3798	3798
50x20	ta057	3839	3818	3842	3843	3833	3828	3845	3837	3822	3838
50x20	ta058	3837	3826	3838	3814	3830	3834	3827	3808	3835	3842
50x20	ta059	3867	3869	3871	3881	3875	3872	3863	3866	3865	3868
50x20	ta060	3864	3878	3876	3890	3876	3869	3890	3898	3885	3888

Tablo K.1.3 100x5-10-20 Problem Boyutu için Algoritma Deneme Sonuçları

$n \times m$	Problem No	Tekrar No				
		1	2	3	4	5
100x5	ta061	5493	5493	5493	5493	5493
100x5	ta062	5274	5268	5274	5274	5274
100x5	ta063	5179	5175	5175	5177	5178
100x5	ta064	5018	5018	5018	5018	5018
100x5	ta065	5250	5252	5252	5250	5250
100x5	ta066	5135	5135	5135	5135	5135
100x5	ta067	5247	5247	5246	5246	5246
100x5	ta068	5098	5094	5097	5094	5100
100x5	ta069	5453	5448	5448	5448	5448
100x5	ta070	5328	5328	5328	5328	5328
100x10	ta071	5810	5804	5786	5796	5804
100x10	ta072	5376	5378	5370	5366	5380
100x10	ta073	5689	5691	5691	5691	5689
100x10	ta074	5851	5845	5858	5847	5854
100x10	ta075	5526	5531	5521	5520	5524
100x10	ta076	5308	5309	5317	5308	5309
100x10	ta077	5629	5647	5635	5638	5631
100x10	ta078	5657	5668	5665	5658	5649
100x10	ta079	5928	5928	5931	5923	5924
100x10	ta080	5850	5881	5848	5878	5877
100x20	ta081	6402	6416	6424	6418	6412
100x20	ta082	6393	6382	6376	6393	6364
100x20	ta083	6466	6461	6469	6470	6455
100x20	ta084	6413	6424	6429	6398	6438
100x20	ta085	6494	6500	6507	6502	6510
100x20	ta086	6528	6545	6542	6541	6548
100x20	ta087	6486	6500	6484	6469	6465
100x20	ta088	6637	6640	6581	6639	6620
100x20	ta089	6483	6482	6469	6466	6474
100x20	ta090	6609	6603	6605	6613	6604

Tablo K.2.1 20x5-10-20 Problem Boyutu için Algoritma Performans Sonuçları

$n \times m$	Problem No	Δ_{enlyi}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}	İş Sıralaması
20x5	ta001	0.000	0.000	0.000	0.000	17,9,15,6,8,3,11,1,19,13,5,7,14,16,18,4,2,10,20,12
20x5	ta002	0.000	0.007	0.074	0.316	6,10,17,7,18,20,12,15,3,19,14,11,9,1,4,13,16,8,5,2
20x5	ta003	0.000	0.250	0.648	3.335	3,4,16,14,13,19,18,20,7,5,10,1,12,11,17,9,6,8,15,2
20x5	ta004	0.000	0.209	0.464	2.497	13,17,19,9,16,11,2,12,15,20,7,1,4,10,5,8,6,14,3,18
20x5	ta005	0.000	0.040	0.324	1.269	12,19,9,10,17,4,3,16,6,5,2,15,13,11,14,7,18,1,20,8
20x5	ta006	0.000	0.000	0.000	0.000	11,14,17,13,16,7,1,6,18,9,12,19,4,15,20,10,8,2,5,3
20x5	ta007	0.000	0.000	0.000	0.000	10,13,1,20,3,2,16,11,12,17,19,15,5,8,4,6,7,9,14,18
20x5	ta008	0.000	0.000	0.000	0.000	17,12,6,2,9,4,16,19,5,7,14,8,20,15,13,10,1,3,18,11
20x5	ta009	0.000	0.480	0.813	3.929	4,12,10,2,8,1,20,16,17,18,3,11,6,14,7,15,13,19,9,5
20x5	ta010	0.000	0.000	0.000	0.000	11,7,6,1,16,10,13,8,19,18,12,2,5,20,14,3,15,17,4,9
20x10	ta011	0.000	0.253	0.632	3.367	18,5,2,12,9,10,4,14,13,15,17,3,6,19,8,20,11,7,1,16
20x10	ta012	0.060	0.603	0.964	4.190	19,15,12,17,9,1,5,11,20,7,10,13,2,4,16,8,14,3,6,18
20x10	ta013	0.735	0.822	1.070	1.636	4,7,9,5,2,6,13,1,16,10,15,11,3,20,17,14,19,12,18,8
20x10	ta014	-0.073	0.160	0.581	2.821	3,11,6,20,10,12,16,18,13,7,1,9,2,15,8,14,19,4,5,17
20x10	ta015	0.000	0.197	0.423	2.530	16,8,4,18,14,15,3,17,13,12,6,19,1,11,2,20,9,7,5,10
20x10	ta016	0.215	0.565	0.931	3.247	19,8,7,18,3,16,17,9,20,6,11,14,13,12,4,5,10,15,2,1
20x10	ta017	0.000	0.040	0.135	0.966	19,4,7,1,6,17,13,18,9,12,5,20,2,10,11,3,16,14,8,15
20x10	ta018	0.390	0.787	1.365	4.909	8,7,18,2,4,3,19,16,14,9,17,15,20,10,1,5,11,13,6,12
20x10	ta019	0.063	0.596	0.942	5.817	14,12,20,11,8,1,7,17,16,18,4,2,19,3,13,15,10,5,6,9
20x10	ta020	0.440	0.867	1.131	4.104	5,4,13,19,15,6,17,7,8,10,18,1,20,16,14,2,9,3,11,12
20x20	ta021	0.218	0.496	0.871	5.621	16,18,14,7,13,8,9,15,1,20,5,12,10,3,17,11,2,6,4,19
20x20	ta022	-0.048	0.176	0.571	5.334	18,11,6,12,4,15,1,16,19,13,8,14,3,10,20,5,9,7,17,2
20x20	ta023	0.215	0.563	0.817	5.507	19,4,5,18,1,2,16,15,3,17,12,13,14,9,11,6,20,10,8,7
20x20	ta024	0.000	0.135	0.405	3.367	14,3,18,16,4,8,13,20,2,15,12,6,1,7,5,19,10,17,9,11
20x20	ta025	0.131	0.297	0.655	3.615	9,14,2,1,15,16,17,10,19,3,18,13,5,12,4,20,7,11,6,8
20x20	ta026	0.180	0.292	0.404	1.716	6,14,13,1,2,18,15,10,3,5,8,9,16,17,20,11,4,7,12,19
20x20	ta027	0.132	0.392	1.056	5.782	10,12,5,9,11,14,1,6,15,17,4,19,2,20,8,18,16,7,3,13
20x20	ta028	0.000	0.364	0.773	5.715	4,20,5,16,10,7,2,17,11,18,14,19,13,12,3,15,6,8,1,9
20x20	ta029	0.000	0.188	0.268	2.251	1,8,14,6,2,13,7,18,17,4,3,11,15,20,9,10,12,16,19,5
20x20	ta030	0.138	0.418	0.735	3.755	3,7,17,19,6,13,18,1,15,12,10,5,4,2,11,8,20,9,16,14

Tablo K.2.2 50x5-10-20 Problem Boyutu için Algoritma Performans Sonuçları

$n \times m$	Problem No	Δ_{eni}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}	İş Sıralaması
50x5	ta031	0.000	0.000	0.000	0.000	31,41,34,6,50,17,3,44,18,10,39,38,45,28,46,2,49,4,7,29,26,9,30,42,40,24,5,16,22,43,23,27,11,13,32,1,25,37,21,14,20,8,15,47,33,35,12,48,19,36
50x5	ta032	0.141	0.208	0.388	2.685	50,49,10,42,15,5,33,18,27,35,22,26,37,19,13,14,34,41,11,8,2,24,46,20,28,40,9,6,36,44,30,45,47,12,7,48,3,39,25,23,38,29,43,16,21,32,1,31,17,4
50x5	ta033	0.000	0.008	0.038	0.422	27,8,2,24,3,10,36,49,11,18,23,4,6,47,38,37,35,25,40,45,9,17,41,43,42,16,29,15,7,28,39,13,1,20,5,26,19,44,34,50,32,33,48,46,31,22,30,21,14,12
50x5	ta034	0.000	0.058	0.073	0.843	42,30,49,2,13,3,14,11,46,9,10,26,50,29,7,24,47,45,33,36,17,19,27,35,5,25,6,4,41,32,28,18,8,43,34,40,20,39,22,1,12,38,37,48,23,15,31,16,21,44
50x5	ta035	0.000	0.003	0.035	0.316	46,27,30,48,36,23,22,21,3,40,32,41,7,16,44,12,28,5,14,19,1,20,34,9,29,6,35,2,4,13,43,17,38,50,37,25,45,8,31,47,15,26,11,39,33,10,24,18,49,42
50x5	ta036	0.000	0.064	0.071	0.632	47,29,41,30,15,24,10,25,28,5,38,44,23,9,39,46,6,7,8,48,45,49,11,32,17,20,3,18,21,37,34,19,33,2,43,40,16,50,36,4,22,42,13,35,14,27,26,1,12,31
50x5	ta037	0.000	0.117	0.257	2.658	27,43,22,30,49,37,39,42,38,40,5,6,34,33,15,9,25,29,19,35,12,48,44,11,2,8,20,31,13,46,10,18,36,21,17,32,4,7,41,3,45,26,14,16,24,23,1,50,47,28
50x5	ta038	0.000	0.123	0.410	4.218	34,47,7,19,14,46,25,45,27,24,8,20,49,28,37,5,43,31,21,41,36,3,44,17,30,15,16,22,13,18,35,26,1,39,10,6,2,33,9,50,48,29,12,11,42,32,4,23,38,40
50x5	ta039	0.000	0.168	0.274	2.452	13,1,3,24,12,48,40,44,19,15,2,23,33,9,26,7,5,11,30,37,42,4,46,32,45,22,41,10,39,21,34,43,49,47,6,38,25,36,35,28,20,8,16,29,18,27,31,50,17,14
50x5	ta040	0.000	0.000	0.000	0.000	6,2,11,1,44,30,8,35,29,23,15,19,18,36,34,28,10,22,50,31,24,27,12,14,16,45,40,48,42,3,17,39,49,47,25,21,46,43,4,41,5,7,13,37,33,9,38,20,26,32
50x10	ta041	0.826	1.464	1.851	9.499	18,34,44,33,49,25,12,20,43,42,38,32,6,14,30,4,31,15,26,5,46,36,45,21,41,17,8,10,37,27,13,19,9,35,11,29,28,3,40,23,47,16,2,22,50,24,7,1,48,39
50x10	ta042	1.625	2.009	2.386	7.094	49,3,33,12,20,23,24,31,35,10,25,47,46,11,38,9,39,4,21,34,18,8,14,7,28,6,17,26,32,50,36,19,43,27,42,44,37,29,5,30,16,1,13,40,15,41,2,45,48,22
50x10	ta043	1.885	2.374	2.723	7.498	24,4,28,25,46,10,33,3,9,11,38,20,47,35,16,45,49,44,40,14,29,1,7,42,5,2,43,41,48,50,13,18,19,32,31,27,17,21,39,36,15,34,30,37,6,26,22,23,12,8

Tablo K.2.2 50x5-10-20 Problem Boyutu için Algoritma Performans Sonuçları
(Devam)

$n \times m$	Problem No	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}	İş Sıralaması
50x10	ta044	0.653	1.061	1.273	5.701	20,5,10,13,28,39,31,41,8,14,48,2,18,22,36,37,1,23,38,49,4,40,35,34,21,11,32,43,30,6,12,17,44,19,47,9,24,46,26,15,7,16,3,29,45,50,33,25,27,42
50x10	ta045	1.708	2.307	2.579	8.950	6,42,35,33,11,10,14,1,48,2,46,16,38,36,49,8,30,45,3,13,47,4,17,34,41,24,31,5,40,7,26,20,18,43,39,12,15,37,27,29,23,25,28,19,50,44,21,9,22,32
50x10	ta046	1.863	2.102	2.295	4.590	11,3,42,25,9,44,39,1,29,27,5,28,22,13,10,14,2,16,46,26,20,23,18,48,32,35,36,24,33,41,15,38,30,49,7,21,47,34,31,40,50,37,4,6,8,17,43,45,19,12
50x10	ta047	1.062	1.381	1.867	8.875	9,20,10,37,28,15,49,45,23,42,24,33,32,22,40,19,29,12,39,1,5,47,3,17,44,21,7,26,48,14,16,13,43,2,50,30,46,11,35,4,27,25,31,38,8,41,36,6,18,34
50x10	ta048	0.658	0.974	1.185	4.926	26,9,1,44,12,6,17,14,11,47,21,19,18,13,29,35,37,25,45,50,46,34,10,43,48,36,33,42,22,49,8,15,20,5,16,32,38,41,4,24,40,23,27,2,31,3,30,7,39,28
50x10	ta049	1.654	1.930	2.240	6.000	33,44,22,47,21,23,39,16,48,45,1,27,30,36,41,6,42,31,37,19,3,26,12,40,14,2,46,50,35,4,18,8,11,17,34,7,43,28,32,15,20,9,24,29,5,10,13,38,49,25
50x10	ta050	1.294	1.540	1.844	6.240	49,6,17,1,20,8,27,24,47,32,44,18,21,22,42,28,11,39,4,29,41,34,40,46,16,9,31,10,3,48,7,33,35,25,43,5,38,13,36,19,15,30,37,26,50,12,14,23,2,45
50x20	ta051	1.961	2.452	2.787	10.551	35,31,39,37,47,15,20,1,44,43,22,36,42,7,33,34,17,46,48,5,14,16,49,8,45,38,12,28,11,10,19,32,26,23,2,29,27,6,13,24,18,40,25,9,21,30,41,50,4,3
50x20	ta052	2.288	3.079	3.445	13.235	33,20,41,32,43,38,36,45,18,17,1,21,40,39,49,12,5,42,19,15,29,14,46,28,25,47,11,35,50,2,4,13,37,44,48,30,34,7,24,31,26,9,16,10,6,3,8,22,23,27
50x20	ta053	3.162	3.394	3.680	6.311	24,4,28,25,8,15,18,22,5,46,11,2,12,35,17,32,30,31,39,41,3,16,48,44,26,14,34,20,27,36,9,45,50,6,47,37,21,33,29,1,43,19,23,42,10,49,38,40,13,7
50x20	ta054	1.599	2.524	2.932	16.125	20,11,45,14,31,49,3,21,38,39,13,34,19,17,44,5,29,46,23,30,15,35,37,47,7,43,6,41,32,8,9,50,48,18,42,22,12,26,33,10,36,40,24,1,4,27,2,25,16,28
50x20	ta055	2.861	3.301	3.686	10.242	40,48,4,2,19,31,21,23,43,33,49,32,47,27,50,20,25,28,38,22,17,41,18,45,6,8,7,36,3,14,39,34,16,5,10,26,9,1,12,44,42,13,46,30,37,24,15,29,35,11
50x20	ta056	2.677	2.850	3.245	6.883	14,37,3,50,47,20,8,46,21,42,13,5,43,33,45,40,4,30,36,49,18,39,11,32,16,48,22,17,28,26,1,7,10,15,12,27,9,2,35,29,34,31,19,24,23,6,25,44,41,38

Tablo K.2.2 50x5-10-20 Problem Boyutu için Algoritma Performans Sonuçları
(Devam)

$n \times m$	Problem No	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}	İş Sıralaması
50x20	ta057	2.745	3.189	3.471	9.132	4,12,10,11,13,20,33,38,49,19,31,46,1,32,41,8,14,47,29,28,15,45,2,5,23,34,22,50,9,3,39,24,37,17,35,48,21,40,44,7,30,27,36,6,26,18,25,42,43,16
50x20	ta058	2.669	3.238	3.586	10.826	39,32,18,7,26,4,20,21,36,8,12,27,15,47,13,35,48,41,49,30,19,46,5,6,16,37,33,25,28,38,45,3,9,29,31,10,17,2,40,43,24,44,22,42,14,1,34,11,23,50
50x20	ta059	2.603	2.781	3.081	5.314	37,35,32,8,12,34,50,3,28,48,41,9,1,46,18,38,24,36,21,19,17,27,4,23,22,49,33,30,31,26,6,7,43,45,13,16,14,2,39,40,10,11,5,42,15,47,25,44,20,29
50x20	ta060	2.303	2.764	3.204	10.575	33,1,19,12,32,3,2,44,36,50,25,5,8,22,31,11,16,21,29,23,28,17,40,39,45,14,41,30,47,9,27,20,38,6,26,15,7,13,18,42,10,24,35,37,46,43,49,4,48,34

Tablo K.2.3 100x5-10-20 Problem Boyutu için Algoritma Performans Sonuçları

$n \times m$	Problem No	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}	İş Sıralaması
100x5	ta061	0.000	0.000	0.000	0.000	31,62,3,90,16,5,93,65,81,96,75,11,9,34,54,45,70,6,74,39,35,42,26,85,15,72,2,71,79,100,37,57,50,21,51,44,55,94,13,60,49,77,95,97,12,27,19,73,4,40,25,14,46,18,67,69,38,17,53,58,28,30,7,41,24,76,98,59,63,29,43,47,80,56,8,36,10,32,64,88,91,22,33,61,92,83,52,78,89,87,68,99,48,20,66,86,23,82,1,84
100x5	ta062	0.000	0.091	0.114	2.683	88,16,86,40,7,14,19,29,3,90,33,28,59,58,34,20,8,61,38,74,53,80,45,43,82,64,98,87,96,32,27,37,18,2,6,46,76,94,35,11,12,56,72,84,77,68,24,66,50,83,26,39,92,78,81,5,100,67,51,57,55,1,97,31,36,25,42,63,65,95,54,15,89,93,4,85,69,71,44,47,52,75,30,21,49,48,23,13,22,99,73,41,91,70,17,9,62,79,10,60
100x5	ta063	0.000	0.035	0.077	1.789	68,41,40,25,30,93,16,45,62,54,98,29,73,37,20,12,14,21,35,81,85,7,59,51,65,96,3,87,80,24,4,52,91,61,46,69,32,100,44,13,17,88,82,6,79,49,28,31,83,38,23,92,99,48,2,33,63,11,10,86,67,26,84,15,18,95,76,8,94,71,57,70,75,74,53,56,19,47,1,27,58,43,22,66,89,78,9,60,72,34,97,77,5,90,50,36,64,42,39,55
100x5	ta064	0.080	0.080	0.080	0.000	20,59,71,69,51,70,26,25,19,47,77,12,48,45,46,13,7,74,27,83,15,6,88,18,75,90,53,29,73,10,44,87,52,63,32,54,9,86,2,60,11,39,36,78,64,50,42,72,89,62,81,16,94,96,8,41,79,82,95,92,34,40,23,22,97,3,49,68,37,55,31,66,57,80,35,33,99,76,67,14,98,65,4,28,1,30,43,100,24,84,21,17,61,91,85,5,58,56,38,93
100x5	ta065	0.000	0.015	0.038	1.095	72,51,21,12,86,64,82,100,38,48,74,32,42,15,9,37,4,2,10,40,90,41,59,69,71,28,75,25,22,26,35,65,57,66,14,6,11,61,93,45,55,67,91,54,87,77,81,99,76,89,5,34,88,29,52,7,80,27,47,50,56,17,30,92,3,95,53,23,84,60,20,94,98,49,16,39,78,44,96,85,24,62,1,43,58,31,8,73,18,63,19,33,36,46,13,70,83,68,97,79
100x5	ta066	0.000	0.000	0.000	0.000	83,54,20,73,80,29,37,61,72,49,97,34,22,60,82,17,43,64,94,85,21,15,3,27,81,42,13,93,33,87,62,9,16,47,84,59,98,75,30,25,8,74,24,11,28,76,65,7,69,96,71,56,38,92,100,5,77,52,55,91,2,50,23,95,26,70,58,31,45,14,36,86,99,41,78,67,35,57,68,6,66,79,88,48,46,19,10,44,89,40,12,18,53,4,51,1,63,39,90,32
100x5	ta067	0.000	0.008	0.019	0.548	45,98,19,77,53,76,85,72,86,37,52,11,59,92,55,6,13,9,80,58,26,47,57,18,68,62,60,14,84,23,79,78,3,29,48,96,25,89,100,69,95,81,75,31,51,34,10,67,20,94,2,46,65,42,74,17,90,36,15,21,91,33,40,83,43,50,88,30,16,54,38,71,7,56,41,63,66,61,27,73,39,49,35,87,99,24,8,12,4,44,32,70,93,1,64,82,97,22,5,28
100x5	ta068	-0.235	-0.184	-0.118	2.608	56,53,21,83,91,80,96,51,77,57,19,47,87,60,31,27,17,70,39,68,4,20,90,23,26,89,76,55,54,9,8,82,58,34,52,61,86,100,44,69,63,14,46,73,6,49,64,38,78,18,97,28,48,99,7,72,50,25,74,30,2,29,71,33,11,1,36,13,43,84,62,24,88,10,3,45,93,75,22,65,12,94,42,92,37,81,67,85,41,16,79,35,59,32,5,98,15,95,66,40

Tablo K.2.3 100x5-10-20 Problem Boyutu için Algoritma Performans Sonuçları
(Devam)

$n \times m$	Problem No	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötü}$	Δ_{std}	İş Sıralaması
100x5	ta069	-0.110	-0.092	-0.018	2.236	70,48,84,57,59,19,35,2,37,25,5,74,67,100,6,13,77,55,96,76,32,11,24,23,86,83,80,1,40,99,58,9,61,34,31,14,7,18,12,54,88,71,29,65,43,26,87,3,21,95,68,63,64,41,50,89,39,72,28,20,94,62,91,22,85,56,92,36,42,82,98,73,45,75,44,33,10,93,30,52,97,46,17,60,49,79,4,51,27,53,38,16,90,66,69,78,8,15,81,47
100x5	ta070	0.000	0.000	0.000	0.000	24,69,31,43,93,63,27,17,64,45,73,72,82,94,22,40,48,21,57,83,47,71,100,88,12,29,77,79,55,20,95,13,76,46,1,81,97,68,50,44,16,59,34,11,75,4,90,58,89,39,9,86,6,3,30,91,42,96,52,26,53,67,84,5,87,56,49,74,14,70,18,25,32,19,41,36,85,66,98,78,23,51,92,80,33,15,28,8,38,10,99,37,60,65,61,54,7,35,62,2
100x10	ta071	0.277	0.520	0.693	9.274	70,21,15,17,58,20,92,39,44,60,24,38,87,65,57,28,81,55,47,95,83,52,97,14,49,61,78,96,73,50,63,64,41,51,8,36,85,27,84,66,43,10,34,77,5,35,37,68,100,31,32,22,2,54,48,82,11,76,1,94,42,56,46,6,89,4,16,19,80,88,67,18,33,86,23,79,69,29,99,90,26,59,74,53,3,93,7,25,71,98,9,91,40,13,75,62,45,30,72,12
100x10	ta072	0.318	0.467	0.580	5.831	69,46,98,87,8,12,66,22,64,54,10,78,37,65,91,6,96,4,53,41,29,60,79,35,80,18,32,93,77,27,89,72,67,74,45,61,25,100,1,86,82,20,40,31,15,17,26,75,49,68,11,58,16,95,81,28,52,57,48,71,73,5,84,34,55,83,97,30,51,50,59,14,88,9,23,3,13,62,85,90,92,33,76,19,94,70,42,43,2,47,21,38,63,44,99,56,36,39,7,24
100x10	ta073	0.211	0.233	0.247	1.095	58,96,45,49,94,62,52,3,86,27,92,93,11,74,72,50,84,77,29,37,64,68,71,4,48,19,22,9,61,59,100,91,65,41,21,32,51,70,43,31,33,76,83,47,34,30,7,79,89,28,69,53,67,56,25,16,85,8,87,66,99,42,5,6,14,73,88,38,98,13,40,12,23,26,78,1,54,10,2,15,80,95,44,63,36,82,75,57,90,35,60,17,97,20,81,24,46,18,39,55
100x10	ta074	0.932	1.036	1.157	5.244	5,53,93,56,24,71,19,32,3,58,6,2,81,52,48,21,72,89,86,80,47,20,35,91,42,17,7,84,34,74,14,100,18,16,67,40,76,75,79,31,92,96,83,45,82,57,99,30,49,55,97,28,26,25,36,8,4,13,78,51,66,77,43,22,70,46,29,60,38,1,23,12,54,41,98,90,68,69,61,73,87,33,27,15,88,9,50,63,39,59,10,94,65,62,37,64,11,44,95,85
100x10	ta075	0.951	1.031	1.152	4.393	79,83,75,56,97,87,10,30,33,42,76,37,66,9,44,72,23,1,36,73,48,69,27,22,35,4,49,20,90,61,65,25,18,28,99,51,40,58,39,94,100,31,80,13,86,85,62,5,88,41,77,34,84,70,89,55,26,64,54,74,2,92,96,60,3,91,47,29,21,15,38,81,19,78,82,95,57,68,71,45,63,8,16,50,59,53,17,7,14,12,98,67,24,93,6,52,46,43,11,32
100x10	ta076	0.094	0.136	0.264	3.834	78,47,24,54,76,53,68,64,48,52,80,46,21,15,38,97,14,35,30,6,36,79,74,31,98,84,60,58,96,51,4,1,29,55,93,57,2,86,87,49,95,8,39,67,9,19,50,81,66,94,37,16,13,88,72,77,41,18,3,23,56,70,12,40,11,92,61,73,22,62,7,5,69,63,34,20,90,89,71,91,10,85,26,42,99,45,83,27,43,17,59,75,28,32,25,100,82,33,65,44

Tablo K.2.3 100x5-10-20 Problem Boyutu için Algoritma Performans Sonuçları
(Devam)

$n \times m$	Problem No	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkötlü}$	Δ_{std}	İş Sıralaması
100x10	ta077	0.536	0.661	0.857	7.071	99,14,25,83,50,55,93,66,70,95,69,15,46,32,5,97,34,3,68,33,23,16,39,1,47,91,78,92,13,58,30,42,60,4,100,31,21,37,6,44,51,71,35,61,62,43,63,80,27,87,45,28,67,36,26,20,18,8,81,9,53,86,90,82,56,57,40,94,24,11,96,64,10,89,7,22,41,52,74,98,2,77,73,48,54,88,19,72,49,84,17,75,79,85,59,29,65,76,38,12
100x10	ta078	0.462	0.647	0.800	7.436	70,31,48,78,19,15,13,38,14,23,22,88,9,43,90,18,12,29,85,77,50,4,61,26,75,30,65,95,6,56,5,62,55,52,28,87,60,59,80,89,100,68,2,37,32,74,83,93,71,20,99,69,72,73,58,46,92,44,39,67,94,63,3,21,97,64,11,27,49,47,40,41,36,79,54,24,16,84,91,98,1,57,53,42,82,76,81,8,33,34,17,7,86,51,96,45,35,25,66,10
100x10	ta079	0.817	0.882	0.953	3.271	54,80,61,2,23,38,5,64,67,98,15,75,82,16,6,12,74,77,45,37,29,21,84,97,19,48,95,49,94,42,100,35,63,58,50,10,79,8,7,66,60,78,93,88,13,86,87,73,46,32,57,68,26,31,28,18,76,44,56,1,43,17,53,25,11,30,59,14,71,27,47,39,22,3,4,24,70,41,89,40,55,85,69,51,34,72,92,65,36,81,90,62,33,20,83,52,99,96,9,91
100x10	ta080	0.051	0.373	0.616	16.331	71,57,97,17,78,8,48,66,92,63,30,80,37,35,58,42,15,54,9,87,2,50,83,49,18,45,84,89,53,36,59,65,95,29,85,60,32,81,94,1,79,68,20,13,46,70,40,88,74,28,21,33,96,100,72,6,14,77,51,52,99,47,75,93,98,55,12,4,7,76,69,16,22,73,11,5,39,38,23,44,61,27,43,31,67,25,26,64,19,86,56,82,3,90,10,24,41,91,62,34
100x20	ta081	1.845	2.043	2.195	8.173	22,54,78,1,55,33,82,40,81,79,10,9,59,12,35,28,21,76,93,60,48,6,3,62,86,50,4,83,74,61,80,23,25,99,39,89,24,56,44,37,45,32,51,46,20,18,53,91,96,66,30,41,19,85,5,94,16,11,72,100,67,2,97,47,27,31,88,26,98,15,68,70,71,65,17,49,64,57,42,36,87,43,13,92,8,29,63,7,95,14,58,84,34,69,90,75,38,52,77,73
100x20	ta082	1.971	2.253	2.436	12.260	49,95,68,66,19,69,52,62,3,97,36,50,47,76,1,14,23,100,65,5,84,40,16,98,10,77,60,86,67,31,82,75,72,32,93,59,71,90,61,30,89,51,43,35,11,64,96,20,13,70,34,78,99,33,48,44,26,85,15,87,38,22,74,2,81,21,53,58,28,9,92,79,8,25,37,54,42,27,83,91,41,17,6,63,56,7,45,94,80,4,46,24,73,18,12,88,29,39,57,55
100x20	ta083	1.991	2.136	2.228	6.221	64,87,10,61,51,11,24,29,5,8,28,94,44,46,48,4,41,56,42,70,97,92,69,43,32,98,2,67,20,35,13,88,60,54,72,96,7,90,23,12,58,26,53,16,49,37,25,66,47,1,19,40,100,6,77,57,78,34,84,76,22,93,73,86,85,27,3,31,33,99,71,80,39,52,81,50,21,30,36,95,15,89,14,65,91,17,74,18,9,55,79,68,38,75,83,62,45,63,59,82
100x20	ta084	1.459	1.814	2.093	15.437	1,68,91,45,69,50,79,44,60,38,55,67,78,89,58,98,86,29,74,57,92,64,84,56,70,87,99,72,46,23,52,95,2,26,42,71,16,51,32,5,8,22,54,88,11,85,9,12,13,41,15,61,63,31,77,35,62,24,93,96,47,34,90,33,48,21,53,25,3,18,49,65,43,7,17,28,30,4,37,82,6,10,19,83,39,66,76,75,20,81,100,59,14,27,73,94,97,80,36,40

Tablo K.2.3 100x5-10-20 Problem Boyutu için Algoritma Performans Sonuçları
(Devam)

$n \times m$	Problem No	Δ_{eniyi}	Δ_{ort}	$\Delta_{enkütlü}$	Δ_{std}	İş Sıralaması
100x20	ta085	1.835	1.970	2.086	6.229	51,98,50,38,94,64,97,12,30,61,46,32,10,15,90,45,68,70,72,11,14,36,53,16,56,52,69,78,28,60,6,33,37,87,41,1,8,29,76,74,5,85,96,17,2,24,31,65,58,7,88,39,40,92,26,86,18,48,21,25,79,62,95,59,44,4,23,66,22,49,3,55,93,71,35,34,9,57,75,82,13,100,89,42,99,20,83,91,43,47,27,63,81,19,80,77,73,84,54,67
100x20	ta086	1.414	1.613	1.724	7.662	56,43,39,92,99,33,83,67,72,2,37,40,80,12,82,86,52,55,32,31,7,34,1,94,4,88,3,100,25,58,8,69,47,35,96,60,77,17,73,28,76,65,98,22,89,66,23,16,10,68,97,45,48,71,81,21,49,27,13,20,38,29,15,6,95,9,53,74,79,50,54,93,64,14,62,59,78,24,75,84,26,44,41,42,57,5,36,18,85,87,46,61,51,91,70,90,11,63,30,19
100x20	ta087	1.875	2.124	2.427	14.096	57,85,93,67,49,95,94,96,58,28,27,48,64,45,20,16,80,99,12,98,39,30,100,53,34,91,13,33,38,72,31,77,43,89,88,42,82,22,19,11,73,50,52,60,79,17,8,9,66,7,10,84,62,61,21,92,41,29,18,46,55,54,90,6,51,47,36,32,4,37,69,59,26,44,56,14,40,71,2,63,5,83,3,1,23,70,97,68,24,65,87,15,74,86,76,81,75,25,78,35
100x20	ta088	1.543	2.197	2.453	25.066	39,62,29,73,90,78,44,56,31,50,67,70,72,59,12,87,6,10,45,74,33,14,68,83,93,9,82,5,48,24,37,95,55,25,53,100,94,81,69,71,22,1,66,35,40,30,46,8,36,23,98,51,19,27,4,77,64,28,65,63,18,75,7,57,52,16,86,38,76,91,41,96,11,17,49,3,61,88,84,58,15,43,21,80,89,2,85,47,92,54,42,34,32,60,13,79,99,20,97,26
100x20	ta089	1.699	1.837	1.966	7.596	60,42,41,50,95,80,15,89,62,16,47,19,54,4,22,84,43,90,81,79,46,27,12,73,40,7,39,61,65,20,33,1,78,21,25,94,26,10,2,82,74,30,13,32,92,38,34,23,11,68,86,28,51,63,14,69,56,75,59,100,87,85,93,77,99,36,57,64,58,44,18,67,55,97,83,76,45,52,17,3,88,48,49,70,53,37,9,35,8,31,29,6,96,24,71,72,66,91,5,98
100x20	ta090	2.135	2.193	2.289	4.147	6,87,40,100,83,52,34,15,61,71,51,20,47,43,56,77,86,78,27,22,97,79,70,11,92,18,98,7,24,12,72,1,26,67,37,10,35,46,2,76,88,16,82,36,32,89,99,96,74,94,65,54,68,60,64,31,62,95,45,69,21,14,81,30,75,48,42,84,90,4,63,23,33,29,49,85,13,73,39,50,8,9,25,57,58,5,19,17,59,41,28,44,91,38,3,66,80,55,53,93

Ek L

Tablo L.1 NEH ve HC Algoritmaları için Deneme Sonuçları

$n \times m$	Problem No	HC				NEH			
		f_1	f_2	f_3	f_{toplam}	f_1	f_2	f_3	f_{toplam}
20x5	ReC01	1342	16577	748	18667	1303	16931	647	18881
20x5	ReC03	1166	14868	809	16843	1132	15103	703	16938
20x5	ReC05	1281	16255	823	18359	1281	16371	813	18465
20x10	ReC07	1626	22739	3939	28304	1626	22739	3939	28304
20x10	ReC09	1583	22435	4249	28267	1583	22435	4249	28267
20x10	ReC11	1550	20732	4031	26313	1550	20732	4031	26313
20x15	ReC13	2002	29320	8859	40181	2002	29320	8859	40181
20x15	ReC15	2031	29417	8685	40133	2013	29150	9487	40650
20x15	ReC17	2037	28979	8288	39304	2019	29300	8051	39370
30x10	ReC19	2185	43000	3768	48953	2185	43000	3768	48953
30x10	ReC21	2126	41294	3714	47134	2131	41330	3729	47190
30x10	ReC23	2157	44031	4460	50648	2155	44937	4393	51485
30x15	ReC25	2644	55250	9205	67099	2644	55250	9205	67099
30x15	ReC27	2552	52778	10253	65583	2498	53142	9961	65601
30x15	ReC29	2391	48941	9501	60833	2391	48941	9501	60833
50x10	ReC31	3188	95970	4095	103253	3173	96278	4156	103607
50x10	ReC33	3183	100791	3597	107571	3163	101074	3482	107719
50x10	ReC35	3333	99389	4941	107663	3308	101667	4662	109637

Tablo L.2 KKE Algoritması için Deneme Sonuçları

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
f_1											
20x5	ReC01	1354	1383	1357	1383	1397	1383	1345	1396	1405	1361
20x5	ReC03	1149	1148	1141	1148	1148	1172	1171	1172	1166	1149
20x5	ReC05	1295	1293	1293	1293	1306	1292	1294	1295	1311	1296
20x10	ReC07	1601	1627	1601	1627	1622	1601	1601	1627	1627	1627
20x10	ReC09	1584	1584	1582	1582	1584	1584	1584	1584	1582	1584
20x10	ReC11	1515	1515	1515	1510	1515	1532	1515	1515	1532	1510
20x15	ReC13	2007	2007	2007	2007	2007	2026	2018	2007	2007	2007
20x15	ReC15	2109	2043	2069	2069	2104	2102	2104	2103	2087	2087
20x15	ReC17	2012	2012	1992	1992	1997	1992	2000	2011	2000	1992
30x10	ReC19	2269	2197	2306	2291	2269	2250	2227	2265	2248	2282
30x10	ReC21	2120	2175	2212	2198	2183	2172	2112	2122	2198	2135
30x10	ReC23	2159	2142	2151	2130	2099	2146	2144	2151	2192	2182
30x15	ReC25	51995	51881	51587	51566	51870	52169	51592	51746	51429	51650
30x15	ReC27	50584	50421	50523	50967	50766	50382	50633	50758	50349	50837
30x15	ReC29	47125	46777	46573	46851	46290	46920	46799	46990	46549	46420
50x10	ReC31	3257	3298	3318	3302	3286	3328	3318	3284	3290	3279
50x10	ReC33	3317	3348	3265	3248	3269	3330	3296	3279	3294	3332
50x10	ReC35	3455	3401	3391	3474	3437	3410	3434	3463	3426	3439
f_2											
20x5	ReC01	15023	15009	15087	15009	14974	15075	15030	14956	14872	15026
20x5	ReC03	13006	12945	12955	12945	12945	12969	12934	12969	12962	13006
20x5	ReC05	15212	15229	15210	15210	15231	15291	15222	15236	15220	15268
20x10	ReC07	21608	21745	21608	21752	21684	21608	21608	21740	21740	21750
20x10	ReC09	20375	20375	20514	20514	20375	20375	20375	20375	20514	20375

Tablo L.2 KKE Algoritması için Deneme Sonuçları (Devam)

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
20x10	ReC11	19461	19461	19461	19544	19461	19383	19461	19461	19383	19548
20x15	ReC13	28212	28212	28212	28212	28212	28399	28065	28212	28212	28212
20x15	ReC15	27853	27933	28055	28055	27881	28024	27881	27977	28031	28031
20x15	ReC17	27185	27185	27177	27177	27242	27177	27228	27213	27228	27177
30x10	ReC19	40409	40775	40288	40446	40285	40743	40781	40630	40723	40451
30x10	ReC21	38809	38964	38684	38672	38724	38861	38624	38874	38521	38752
30x10	ReC23	38295	38684	38655	38474	38592	38660	38191	38234	38191	38162
30x15	ReC25	51995	51881	51587	51566	51870	52169	51592	51746	51429	51650
30x15	ReC27	50584	50421	50523	50967	50766	50382	50633	50758	50349	50837
30x15	ReC29	47125	46777	46573	46851	46290	46920	46799	46990	46549	46420
50x10	ReC31	87894	88218	88196	88448	88304	88632	88696	88762	88205	87912
50x10	ReC33	90441	91408	90646	90952	91170	91307	91312	91172	91153	91667
50x10	ReC35	90787	91480	91276	90859	91162	91267	90816	91575	91426	91200
f_3											
20x5	ReC01	645	629	603	629	660	629	605	676	684	615
20x5	ReC03	642	638	610	638	638	611	645	611	632	642
20x5	ReC05	788	756	756	756	796	754	786	764	806	773
20x10	ReC07	3537	3394	3537	3394	3444	3537	3537	3394	3394	3394
20x10	ReC09	3574	3574	3550	3550	3574	3574	3574	3574	3550	3574
20x10	ReC11	3408	3408	3408	3385	3408	3483	3408	3408	3483	3372
20x15	ReC13	7896	7896	7896	7896	7896	7782	8078	7896	7896	7896
20x15	ReC15	7627	7672	7430	7430	7562	7486	7562	7520	7494	7494
20x15	ReC17	7591	7591	7612	7612	7618	7612	7508	7579	7508	7612
30x10	ReC19	3736	3496	3782	3930	3608	3554	3665	3642	3576	3724
30x10	ReC21	2894	2859	3058	3039	2863	3018	3136	2870	3132	3029
30x10	ReC23	3728	3683	3771	3861	3745	3750	3770	3853	3842	3910

Tablo L.2 KKE Algoritması için Deneme Sonuçları (Devam)

$n \times m$	Problem No	Tekrar No									
		1	2	3	4	5	6	7	8	9	10
30x15	ReC25	8284	8457	8550	8448	8683	8217	8677	8474	9015	8450
30x15	ReC27	8999	9323	9174	9053	9270	9196	9354	9027	9135	9036
30x15	ReC29	7453	7732	7925	7810	7872	7524	7485	7614	7674	7703
50x10	ReC31	4689	4329	4452	4411	4046	4109	4060	3977	4281	4466
50x10	ReC33	3132	3271	2955	3242	3184	3066	3329	3084	3127	3283
50x10	ReC35	4269	4236	4672	4665	4270	4447	4259	4320	4377	4291
f_{toplam}											
20x5	ReC01	17022	17021	17047	17021	17031	17087	16980	17028	16961	17002
20x5	ReC03	14797	14731	14706	14731	14731	14752	14750	14752	14760	14797
20x5	ReC05	17295	17278	17259	17259	17333	17337	17302	17295	17337	17337
20x10	ReC07	26746	26766	26746	26773	26750	26746	26746	26761	26761	26771
20x10	ReC09	25533	25533	25646	25646	25533	25533	25533	25533	25646	25533
20x10	ReC11	24384	24384	24384	24439	24384	24398	24384	24384	24398	24430
20x15	ReC13	38115	38115	38115	38115	38115	38207	38161	38115	38115	38115
20x15	ReC15	37589	37648	37554	37554	37547	37612	37547	37600	37612	37612
20x15	ReC17	36788	36788	36781	36781	36857	36781	36736	36803	36736	36781
30x10	ReC19	46414	46468	46376	46667	46162	46547	46673	46537	46547	46457
30x10	ReC21	43823	43998	43954	43909	43770	44051	43872	43866	43851	43916
30x10	ReC23	44182	44509	44577	44465	44436	44556	44105	44238	44225	44254
30x15	ReC25	62933	63004	62821	62631	63266	62992	62974	62906	63173	62815
30x15	ReC27	62054	62281	62267	62490	62535	62205	62524	62309	61971	62426
30x15	ReC29	57053	56947	56954	57100	56590	56883	56737	57050	56695	56546
50x10	ReC31	95840	95845	95966	96161	95636	96069	96074	96023	95776	95657
50x10	ReC33	96890	98027	96866	97442	97623	97703	97937	97535	97574	98282
50x10	ReC35	98511	99117	99339	98998	98869	99124	98509	99358	99229	98930

Tablo L.3 Problem Bazında Algoritma Bağlı Hata Sonuçları

$n \times m$	Problem No	f_1						f_2						f_3						f_{toplam}			
		HC	Δ_{NEH}	KKE			HC	Δ_{NEH}	Δ_{eniyi}	Δ_{ort}	$\Delta_{enklilik}$	HC	Δ_{NEH}	Δ_{eniyi}	Δ_{ort}	$\Delta_{enklilik}$	HC	Δ_{NEH}	Δ_{eniyi}	Δ_{ort}	$\Delta_{enklilik}$	KKE	KKE
				Δ_{eniyi}	Δ_{ort}	$\Delta_{enklilik}$																	
20x5	ReC01	2.99	0.00	3.22	5.63	7.83	11.46	13.84	0.00	0.00	0.00	24.05	7.30	0.00	0.00	5.72	38.50	21.14	3.22	5.63	13.55		
20x5	ReC03	3.00	0.00	0.80	2.16	3.53	14.95	16.77	0.00	0.00	0.00	32.62	15.25	0.00	0.00	0.00	50.58	32.02	0.80	2.16	3.53		
20x5	ReC05	0.00	0.00	0.86	1.23	2.34	6.87	7.63	0.00	0.00	0.00	9.15	7.82	0.00	0.00	0.00	16.02	15.46	0.86	1.23	2.34		
20x10	ReC07	1.56	1.56	0.00	0.00	0.06	5.23	5.23	0.00	0.00	0.00	16.06	16.06	0.00	0.00	0.00	22.85	22.85	0.00	0.00	0.06		
20x10	ReC09	0.06	0.06	0.00	0.03	0.06	10.11	10.11	0.00	0.00	0.00	19.69	19.69	0.00	0.00	0.00	29.86	29.86	0.00	0.03	0.06		
20x10	ReC11	2.65	2.65	0.00	0.00	0.00	6.96	6.96	0.00	0.00	0.00	19.54	19.54	0.00	0.00	0.00	29.15	29.15	0.00	0.00	0.00		
20x15	ReC13	0.00	0.00	0.25	0.40	1.20	4.47	4.47	0.00	0.00	0.00	13.84	13.84	0.00	0.00	0.00	18.31	18.31	0.25	0.40	1.20		
20x15	ReC15	0.89	0.00	1.49	3.71	4.77	5.62	4.66	0.00	0.00	0.00	16.89	27.69	0.00	0.00	0.00	23.40	32.34	1.49	3.71	4.77		
20x15	ReC17	2.26	1.36	0.00	0.00	0.00	6.63	7.81	0.00	0.00	0.00	10.39	7.23	0.00	0.00	0.00	19.28	16.40	0.00	0.00	0.00		
30x10	ReC19	0.00	0.00	0.55	3.45	5.54	6.74	6.74	0.00	0.00	0.00	7.78	7.78	0.00	0.00	4.30	14.52	14.52	0.55	3.45	9.84		
30x10	ReC21	0.66	0.90	0.00	1.73	4.05	7.20	7.29	0.00	0.00	0.00	29.91	30.43	0.00	0.00	0.00	37.77	38.62	0.00	1.73	4.05		

Tablo L.3 Problem Bazında Algoritma Bağlı Hata Sonuçları (Devam)

$n \times m$	Problem No	f_1				f_2				f_3				f_{toplam}			
		HC	NEH	KKE		HC	NEH	KKE		HC	NEH	KKE		HC	NEH	KKE	
		Δ_{HC}	Δ_{NEH}	Δ_{enlyi}	Δ_{ort}	Δ_{HC}	Δ_{NEH}	Δ_{enlyi}	Δ_{ort}	Δ_{HC}	Δ_{NEH}	Δ_{enlyi}	Δ_{ort}	Δ_{HC}	Δ_{NEH}	Δ_{enlyi}	Δ_{ort}
30x10	ReC23	2.76	2.67	0.00	0.00	1.72	15.38	0.00	0.00	21.10	19.28	0.00	0.00	39.24	39.70	0.00	0.00
30x15	ReC25	1.46	1.46	0.00	1.27	3.21	7.43	0.00	0.00	12.02	12.02	0.00	0.00	20.91	20.91	0.00	1.27
30x15	ReC27	3.32	1.13	0.00	1.18	5.16	4.82	0.00	0.00	13.93	10.69	0.00	0.00	22.08	17.37	0.00	1.18
30x15	ReC29	0.00	0.00	1.34	2.34	3.51	5.73	0.00	0.00	27.48	27.48	0.00	0.00	33.21	33.21	1.34	2.34
50x10	ReC31	0.47	0.00	2.65	3.88	4.88	9.19	0.00	0.00	2.97	4.50	0.00	4.57	12.63	14.04	2.65	8.44
50x10	ReC33	0.63	0.00	2.69	4.26	5.85	11.44	0.00	0.00	21.73	17.83	0.00	0.00	33.80	29.59	2.69	4.26
50x10	ReC35	0.76	0.00	2.51	3.78	5.02	9.47	0.00	0.00	16.64	10.06	0.00	0.00	26.87	22.04	2.51	3.78
Ortalama=		1.30	0.65	0.91	1.95	3.26	8.32	0.00	0.00	17.54	15.25	0.00	0.25	27.17	24.86	0.91	2.20

ÖZGEÇMİŞ

Betül YAĞMAHAN, 1974 yılında Bursa' da doğdu. İlk, orta ve lise öğrenimini yine Bursa' da tamamladı. 1995 yılında Uludağ Üniversitesi Mühendislik-Mimarlık Fakültesi Endüstri Mühendisliği Bölümünden mezun oldu. Aynı yıl Bürosit A.Ş. bünyesinde çalıştı. Bu arada Uludağ Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalında yüksek lisans eğitime başladı. 1999 yılında yüksek mühendis unvanını aldı. Aynı yıl İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalında doktora eğitime başladı. 1997-2003 yılları arasında Uludağ Üniversitesi Mühendislik-Mimarlık Fakültesi Endüstri Mühendisliği Anabilim Dalında araştırma görevlisi olarak çalıştı. Üretim Planlama, Çizelgeleme, Sezgisel&Metasezgisel Eniyileme ve Bilişim Sistemleri konularıyla ilgilenmektedir.

