ÇOK İŞLEMCİLİ İŞLERİN ÇOK KATMANLI PARALEL İŞLEMCİLİ AKIŞ ATÖLYELERİNDE ÇİZELGELENMESİ

Funda SİVRİKAYA ŞERİFOĞLU[1]

Gündüz ULUSOY[2]

Abant İzzet Baysal Üniversitesi, İktisadi ve İdari Bilimler Fakültesi[1]

Sabancı Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi[2]

ÖZET

Bu çalışmada, her biri birden çok işlemcide işlenmesi gereken *n* adet işin *m* katmanlı bir paralel işlemcili akış atölyesinde çizelgelenmesi problemi ele alınmıştır. Bu problemi çözmek üzere bir genetik algoritma geliştirilmiştir. Eniyilemenin amacı enbüyük bitiş zamanını enküçüklemektir; diğer bir deyişle, en son aşamada tüm işlemlerin tamamlandığı zamanın enküçüklenmesi amaçlanmaktadır. Genetik algoritma sonuçları, teknik yazında rapor edilen bir alt sınır ile karşılaştırılmıştır. Test problemi kümesi 100 iş, 10 katman ve her katmanda 5 işlemciye kadar işlemci içeren 400 problemi içermektedir. Deneysel çalışma sonucu, önerilen genetik algoritmanın iyi çözümleri kısa sürede veren etkin bir algoritma olduğu gösterilmiştir.

Anahtar kelimeler: Paralel işlemcili akış atölyesi, çok işlemcili işler, bitiş zamanının enküçüklenmesi, genetik algoritma.

ABSTRACT

A genetic algorithm is developed to schedule multi-processor tasks in a multistage hybrid flow shop environment. The objective is to minimize the make-span, i.e. the completion time of all jobs. The genetic algorithm is tested against a lower bound from literature on a test bed comprising of 400 problems with up to 100 jobs, 10 stages, and up to 5 processors on each stage. It has proven itself an effective and efficient algorithm for the stated problem by finding optimal and near optimal solutions in reasonable times.

Keywords: Hybrid flowshops, multiprocessor jobs, makespan minimization, genetic algorithms

GİRİŞ

Paralel işlemcili akış atölyesi (*hybrid flow shop*) çizelgeleme problemleri, akış atölyesi çizelgeleme problemleri ile paralel makina (işlemci) çizelgeleme problemlerinin özelliklerini birleştirmektedir. Akış atölyelerinde bütün işler atölyenin aşamalarını aynı sırayla ziyaret ederler. Paralel işlemcili akış atölyelerinde,

her aşamada, işleri işleyecek bir veya daha fazla birbirinin aynı (*identical*) işlemci vardır. Bu durum üretim sistemine ek bir esneklik kazandırır. Eniyileme kriteri genellikle enbüyük bitiş zamanının (*makespan*) enküçüklenmesidir.

İki aşamalı paralel işlemcili akış atölyesi probleminin NP-zor (*NP-hard*) olduğu Gupta [7] tarafından gösterilmiştir. Hoogeven vd. [11] işlerin tezgahtan sökülerek işleme ara verilebildiği (*preemptive*) durumun da NP-zor olduğunu göstermiştir. Bütün bu olumsuz teorik sonuçlara rağmen paralel işlemcili akış atölyesi problemi pek çok araştırmacının ilgisini çekmektedir. Bu tip üretim sistemlerine tekstil ve proses endüstrileri gibi endüstrilerde rastlanmaktadır.

Paralel işlemcili akış atölyesi ile ilgili araştırmalar genel olarak iki-aşamalı problemler üzerine yoğunlaşmıştır ([9] ve [15] gibi). Az sayıda araştırmacı üç aşamalı problemleri ele almıştır ([19] ve [3] gibi). Grangeon vd. [6] çok aşamalı akış atölyeleri için stokastik algoritmalar geliştirmişlerdir. Vignier ve Venturini [22] çok aşamalı paralel işlemcili akış atölyesi problemini çözmek üzere bir paralel genetik algoritma (GA) geliştirmiştir. Portman vd. [16] dal sınır algoritmasını genetik algoritma ile çaprazlayarak melez bir algoritma geliştirmiştir. Çok aşamalı akış atölyeleri ve paralel işlemcili akış atölyeleri ile ilgili bir tarama Riane ve Artiba [17] tarafından hazırlanmıştır. Paralel işlemcili akış atölyeleri ile ilgili kaynakça ve problem kütüphanesi için şu web sitesine bakılabilir : *http://192.54.142.54/grangeon/francais/literature_fr.html*. Son zamanlarda enbüyük gecikme [21] ve geç biten iş sayısı [8] gibi termin tarihi ile ilgili performans kriterleri de kullanılmaya başlanmıştır.

Yukarıda sözü edilen tüm araştırmalarda, işler her aşamada sadece tek bir işlemcide işlenmektedir. Bu kısıt gevşetilebilir. Böylece birden çok işlemci gerektiren işlerle ilgili problemler incelenebilir. Birden çok işlemci gerektiren işlerle ilgili problemler üzerine [1], [4] ve [12] gibi tarama araştırmaları vardır.

Paralel işlemcili akış atölyelerinde birden çok işlemci gerektiren işlerin çizelgelenmesi problemi henüz oldukça yeni bir araştırma alanıdır. Oğuz vd. [15] çok aşamalı paralel işlemcili akış atölyesinde birden çok işlemci gerektiren işlerin çizelgelenmesi problemi için tabu arama esaslı sezgisel bir yaklaşım geliştirmişlerdir. Oğuz vd. de [14] iki aşamalı paralel işlemcili atölye için benzer bir problemi çözmek üzere bir genetik algoritma geliştirmişlerdir.

PROBLEMİN TANIMI

Bu çalışmada ele alınan problemde m aşamalı bir paralel işlemcili akış atölyesi söz konusudur. Bu atölyede her biri m işlem gerektiren n tane iş çizelgelenecektir. Her aşamada, m_i adet birbirinin aynı paralel işlemci vardır, i=1,..,m. j işinin herhangi bir i aşamasında işlenebilmesi için $size[i,j]$ ile belirtilen adette işlemci aynı anda j işine atanmalıdır. Diğer bir deyişle, i aşamasında j işine atanan $size[i,j]$ adet işlemci aynı anda j işini işlemeye başlayarak $p[i,j]$ uzunluğunda bir süre boyunca j işini işlemeye devam ederler. Burada $p[i,j]$ i=1,..,m j=1,..,n, j işinin i aşamasındaki işlem süresini ifade eder.

Her işlemci herhangi bir anda sadece bir işlemi yürütebilmektedir. İşlemciler arıza yapmazlar. Bütün işler çizelgeleme dönemi başında hazırdırlar. İşler işlenmeye başlandıktan sonra kesintisiz işlenir. Amaç, enbüyük bitirme zamanının enküçüklenmesidir; yani en son aşamada tüm işlemlerin tamamlandığı zamanın enküçüklenmesi amaçlanmaktadır.

GENETİK ALGORİTMA

Genetik algoritmalar, John Holland tarafından yapay adaptif sistemler olarak geliştirildiler [10]. Bu algoritmalar, eniyileme problemlerini çözmede giderek daha yaygın olarak kullanılıyorlar. Bu araştırma için geliştirdiğimiz GA, *n* adet işin sıralamasına dayalı bir kromozom yapısı kullanmaktadır. Bu sıralama, işlerin ilk aşamada çizelgelenme sırasında ele alınış sırasını vermektedir. Sonraki aşamaların her birinde (i=2,..,m), işler bir önceki (i-1) aşamadaki bitiş zamanlarına göre sıralanmaktadırlar.

Herhangi bir i aşamasında, i=1,..,m , çizelgeleme yapmak için basit bir gecikmesiz (*nondelay*) algoritma kullanılmaktadır: i aşaması ile ilgili iş sıralamasında yer alan bir sonraki iş j, ilk boş kalan *size[i,j]* işlemciye birden atanır. Bu tip çizelgeleme benzer problemler üzerinde araştırmacılar tarafından kullanılmıştır ([18] ve [15]).

Bu araştırmada amaç, enbüyük bitirme zamanını enküçüklemektir. Amaç fonksiyonu enbüyük bitirme zamanının alt sınırdan yüzdelik sapması olarak tanımlanmıştır. Matematiksel olarak ifade edilirse, z=100*(C_{max}-LB)/LB. Burada z enküçüklenecek amaç fonksiyonunu, C_{max} enbüyük bitirme zamanını ve LB altsınırı ifade etmektedir. GA'nın enbüyüklemeye çalışacağı uygunluk (*fitness*) fonksiyonu ise amaç fonksiyonu değerinin nesil içindeki enbüyük (yani en kötü) amaç fonksiyonu değerinden sapması olarak tanımlanmıştır. Matematiksel olarak ifade edilirse f=z_{max}-z; burada f uygunluk fonksiyonunu ve z_{max} nesil içindeki enbüyük amaç fonksiyonu değerini ifade etmektedir.

Alt sınır Oğuz vd. [15] tarafından geliştirilmiştir ve formülü şöyledir:

$$LB = \max_{i \in M} \left\{ \min_{j \in J} \left\{ \sum_{k=1}^{i-1} p[k,j] \right\} + \frac{1}{m_i} \sum_{j \in J} p[i,j] size[i,j] + \min_{j \in J} \left\{ \sum_{k=i+1}^{m} p[k,j] \right\} \right\} \quad (1)$$

Seçme operatörü olarak rulet-çemberi (*roulette wheel*) seçimi kullanılmıştır. Koşturmalar sırasında yaratılan iyi çözümleri kaybetmemek üzere seçkinci (*elitist*) strateji kullanılmaktadır. Seçkinci strateji, her neslin en iyi belli sayıda kromozomunun hiç bozulmadan bir sonraki nesle aktarılmasıdır.

Mutasyon operatörü seçmek üzere iş değişimi (*job exchange*) ve yer değişimi (*job replacement*) operatörleri ile bir dizi ön deneme yapılmıştır. İş değişimi operatörü, kromozom üzerinde iki işi rassal olarak seçmekte ve yerlerini değiştirmektedir. Yer değişimi operatörü ise rassal olarak seçilen bir işi yerinden alarak rassal olarak seçilen bir başka yere nakletmektedir. Denemeler sonucu iş değişimi operatörü tercih edilmiştir.

Çaprazlama operatörünü seçmek için iki farklı operatör ile deneyler yapılmıştır: Murata ve diğerlerinin [13] önerdiği iki-nokta çaprazlama operatörü ve Davis'in geliştirdiği [2] düzgün sıralama temelli çaprazlama operatörü UOBX. Diğer GA parametrelerinin çeşitli değerleri için yapılan koşularda UOBX ile daha iyi sonuçlara ulaşılmıştır. Bu nedenle UOBX tercih edilmiştir.

İlk nesil rassal olarak yaratılmış fakat üç kromozom ile tohumlanmıştır: Birinci aşamanın işlem sürelerine göre en kısa süre (SPT) sıralaması, birinci aşamanın işlem sürelerine göre en uzun işlem süresi (LPT) sıralaması, ve son olarak, en kısa toplam işlem süreleri (STPT) sıralaması. SPT ve LPT sıralamaları yapılırken, herhangi bir iterasyonda, iki veya daha fazla iş eşit değere sahipse, eşitliği bozma kuralı (*tie-break rule*) olarak STPT'si daha kısa olan işe öncelik verilmiştir. STPT sıralaması yaparken, eşitlik durumunda, birinci aşama işlem süresi daha kısa olan işe öncelik verilmiştir.

GA'nın koşturulması sonunda elde edilen en iyi çözüm 2-opt yerel arama algoritmasına tabi tutulmuş ve böylelikle GA'nın bulduğu tepeye tırmanmak amaçlanmıştır.

GA'nın parametreleri bir dizi ayarlama deneyi sonucu şu şekilde belirlenmiştir: Nesil büyüklüğü 50, nesil sayısı 400, çaprazlama ve mutasyon olasılıkları 0.75 ve 0.75. Seçkinci stratejide her neslin en iyi iki kromozomu bir sonraki nesile aynen aktarılmıştır. GA her problem için 5 kez koşturulmuştur. Herhangi bir koşturma

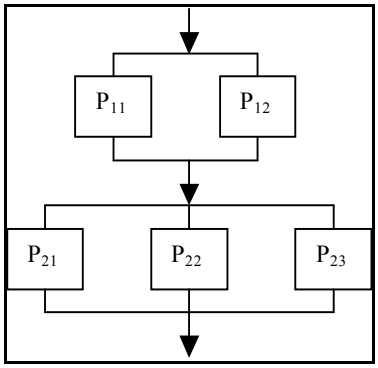
sırasında elde edilen çözümlerden biri alt sınır değerine eşit bir C_{max} değeri veriyorsa, eniyi (*optimal*) çözüm elde edildiği için, GA'nın koşturulması 400 neslin tamamlanması beklenmeden sona erdirilmiştir.

ÖRNEK PROBLEM

Örnek olmak üzere beş iş içeren iki aşamalı (n=5, m=2) bir problemi ele alalım. Aşağıda verilen Şekil 1'de, iki aşamalı paralel işlemcili akış atölyesinin şeması görülmektedir. İlk aşamada 2 işlemci (P_{11} ve P_{12}) ve ikinci aşamada da 3 işlemci (P_{21}, P_{22} ve P_{23}) bulunmaktadır. Tablo 1'de ise bu paralel işlemcili akış atölyesinde çizelgelenmesi istenen beş adet işe ait bilgiler verilmiştir. İş 1, örneğin, birinci aşamada bir işlemcide 75 birim zaman ve ikinci aşamadaki üç işlemcide birden 22 birim zaman işlem görmelidir. Bu problemin alt sınır değeri 232'dir.

Şekil 2, örnek bir kromozomu ve bu kromozomda önerilen çözümün Gantt şemasını göstermektedir. [4 2 1 5 3] kromozomunun ifade ettiği çözümde enbüyük bitirme zamanının (C_{max}) değeri 317'dir.

Örnek kromozom, en kısa süre (SPT) kuralının ilk aşama işlem sürelerine ($p[1,j]$) uygulanması ile bulunmuş olan bir sıralamayı içermektedir. Gantt şemasında da görüldüğü gibi, bu problem için, SPT sıralaması ikinci aşamada da aynen geçerli kalmıştır çünkü ilk aşamada işlerin bitişleri de bu sırayla gerçekleşmiştir. Ama farklı problemlerde ilk aşamayı takip eden aşamalara ait sıralamalar farklı olabilmektedir.

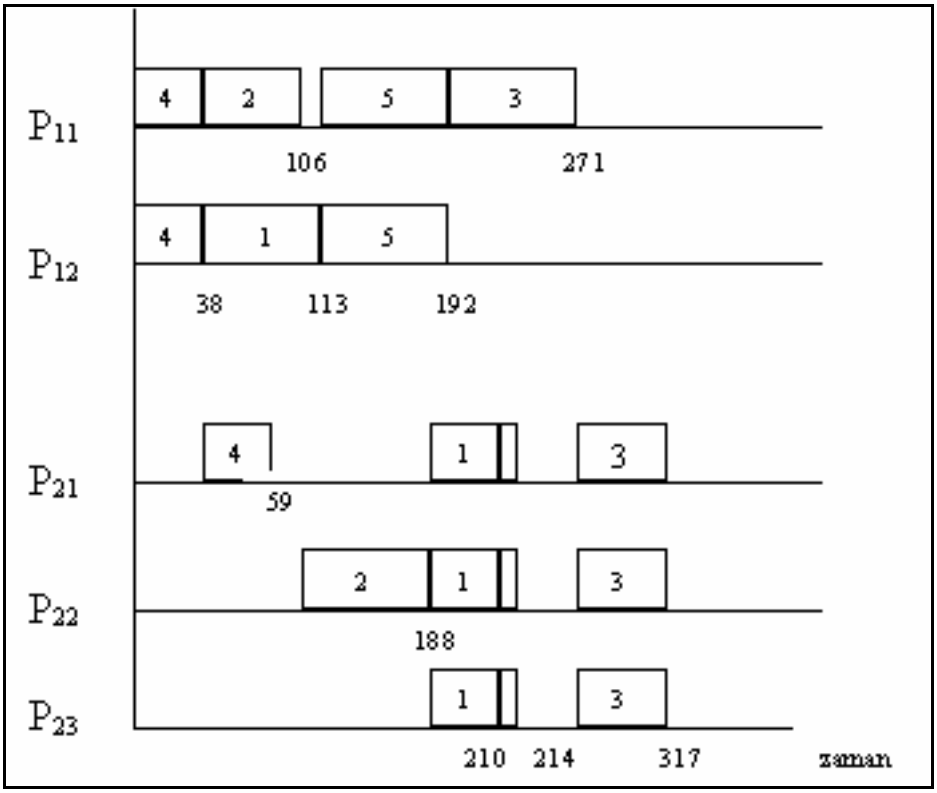


Şekil 1. *Örnek Probleme Ait İki Aşamalı Paralel İşlemcili Akış Atölyesi Şeması*

Tablo 1. Örnek Probleme Ait Veriler

iş	aşama	işlem süresi	işlemci ihtiyacı
1	1	75	1
	2	22	3
2	1	68	1
	2	82	1

3	1 2	79 46	1 3
4	1 2	38 21	2 1
5	1 2	79 4	2 3



Şekil 2. *[4 2 1 5 3] Örnek Kromozomunun Çizelgesinin Gantt Şeması*

Örnek problem için GA koşturmalarından elde edilen en iyi çözümlerden birisi [3 2 1 4 5] kromozomu ile ifade edilmiştir. Bu çözümde enbüyük bitirme zamanı 264'tür. GA, yapılan tüm deneylerde her koşturmada bu çözümü bulmuştur. Bu çözümün, deneysel çalışma bölümünde anlatılacak olan listeleme sonucu, eniyi çözüm olduğu bulunmuştur.

DENEYSEL ÇALIŞMA

Oğuz ve diğerlerinin [15] yaptığına benzer bir deneysel çalışma gerçekleştirilmiştir. Test problemi kümesi hazırlanırken iş sayısı için n=5,10,20,50,100 değerleri ve paralel işlemcili akış atölyesi aşamaları sayısı için

m=2,5,8,10 değerleri alınmıştır. Her n ve m ikilisi için 10'ar adet A ve B tipi problem yaratılmıştır. A tipi problemlerde, akış atölyesinin aşamalarındaki işlemci sayısı, m_i, değişken olabilmektedir. Bu sayı [1,..,5] aralığından rassal olarak seçilmektedir. B tipi problemlerde ise işlemci sayısı her aşamada sabit ve 5'e eşittir, yani m_i =5 i=1,..,m. Her iki tip problemde de, işlerin işlemci ihtiyaçları, yani *size[i,j]*, [1,..,m_i] aralığından rassal olarak seçilmiştir. İşlem süreleri *p[i,j]* ise [1,..,100] aralığından tam sayı ve rassal olarak seçilmiştir. Test problemi kümesi 400 problemden oluşmaktadır.

GA, her n ve m ikilisi için yaratılmış olan 10 adet A tipi ve 10 adet B tipi problemin her biri üzerinde beşer kez koşturulmuştur. Beş koşumun sonunda elde edilen en iyi çözüm (alt sınırdan en düşük yüzdelik sapmaya sahip olan çözüm), GA'nın eldeki problem için ürettiği çözüm olarak alınmıştır. Aynı n ve m ikilisine ait olan 10 problem için GA'nın bulduğu en iyi çözümlerin yüzdelik sapma değerlerinin ortalaması alınmıştır. Tablo 2'de bu ortalamalar (alt sınırdan yüzdelik sapmaların ortalaması) A ve B tipi problemler için gösterilmiştir. Tablo 2'de ayrıca GA'nın her koşturulmasının CPU saniyesi cinsinden ortalama süresi de verilmiştir. GA, TURBO Pascal 7.0'da derlenmiş ve 1000 MHz'lik PIII işlemciye ve 512 MB RAM'e sahip bir bilgisayarda koşturulmuştur.

Tablo 2. A Tipi ve B Tipi Problemler İçin Alt Sınırdan Yüzdelik Sapmaların ve CPU Saniye Olarak Koşturulma Zamanlarının Ortalaması

		A tipi problemler		B tipi problemler	
n	m	ort%sapma	CPU sn.	ort%sapma	CPU sn.
5	2	9,26	0,50	24,89	0,60
	5	25,71	0,80	40,21	0,92
	8	26,30	1,00	32,42	1,18
	10	24,15	1,14	34,56	1,40
10	2	2,88	0,48	12,83	0,88
	5	10,45	1,06	19,59	1,56
	8	18,88	1,74	34,59	2,18
	10	13,63	2,08	30,91	2,60
20	2	5,53	1,18	6,74	1,50
	5	6,28	1,96	14,75	3,04
	8	6,98	3,60	26,65	4,54
	10	10,23	4,34	32,21	5,56
50	2	2,72	3,00	6,66	4,56
	5	2,06	7,00	14,10	12,60
	8	4,53	15,32	21,87	19,44
	10	4,47	18,54	26,36	24,28
100	2	2,37	20,50	6,43	36,06
	5	2,09	65,94	13,01	127,86
	8	3,58	102,64	20,50	182,28
	10	1,72	121,90	24,78	205,66

Elde edilen sonuçlar, Oğuz vd. [15] tarafından tabu arama algoritması ile bulunan sonuçlara hem büyüklük açısından hem de artan n ve m karşısında gösterdikleri değişim açısından benzemektedir. Ayrıca her iki

araştırmada da B tipi problemler için alt sınırdan sapmalar daha büyük olarak bulunmuştur. Genel olarak, bu araştırmada GA ile bulunan sapma değerleri tabu arama ile bulunan değerlerden daha küçüktür, fakat test problemleri farklı olduğu için iki yaklaşımın performansı ile ilgili kararlar vermek zordur.

Tablo 2 incelendiğinde, *n* sabit tutulduğunda, artan *m* değeri ile genel olarak yüzdelik sapmaların arttığı görülmektedir. *m* sabit tutulduğunda, artan *n* değeri ile sapmalar düşmektedir. Oğuz vd'nin de [15] bulgularına göre genel olarak artan *m* ve m_i değeri ile alt sınırın etkinliği azalmaktadır. Yani, artan *m* değeri ile sapmalardaki artışın ve B tipi problemlerde gözlenen büyük sapmaların nedeni alt sınırın zayıflayışıdır. Yine aynı kaynakta söz edildiği gibi alt sınırdan sapmaların artan *n* değeri ile düşmesi beklenen bir gelişmedir ve enbüyük bitirme zamanının değerinin büyümesi ile ilgilidir.

GA'nın performansı ile ilgili daha fazla bilgi edinmek üzere GA'nın bulduğu en iyi çözüm, ilk nesilde tohumlama için kullanılan SPT, LPT ve STPT sezgisel kurallarıyla elde edilen çözümlerin en iyisi ile karşılaştırılmıştır. Böylelikle GA'nın bu kurallara kıyasla getirdiği iyileştirme ölçülmek istenmiştir. Tablo 3'te, her *n* ve *m* çiftine ait 10 problem için, GA çözümünün sezgisel kuralların en iyi çözümünden sapmalarının ortalaması verilmiştir.

Tablo 3 incelendiğinde GA'nın sezgisel kurallara göre önemli iyileştirmeler getirdiği anlaşılmaktadır. Problem zorluğu arttıkça (artan *n*, *m* ve m_i değerleri ile) GA'nın göreli performansı da iyileşmektedir. Bu, sezgisel bir algoritma için önemli bir özelliktir.

Tablo 3. A Tipi ve B Tipi Problemler İçin GA Çözümlerinin SPT, LPT ve STPT Çözümlerinin En İyisinden Ortalama Yüzdelik Sapmaları ve T-Testi Değerleri

		A tipi problemler		B tipi problemler	
n	*m*	ort%sapma	t değeri	ort%sapma	t değeri.
5	2	-5,22	2,89	-8,80	4,15
	5	-6,88	3,84	-6,93	4,56
	8	-7,77	3,60	-7,22	3,63
	10	-5,28	5,61	-7,19	3,88
10	2	-7,32	3,73	-16,93	7,40
	5	-15,00	7,79	-17,45	8,80
	8	-17,00	9,23	-13,68	9,05
	10	-14,59	12,27	-15,03	9,28
20	2	-12,70	4,92	-18,97	7,80
	5	-16,44	19,10	-21,99	12,39
	8	-15,13	9,80	-20,57	18,74
	10	-17,99	13,58	-18,79	19,81
50	2	-12,86	6,14	-17,47	26,12
	5	-11,77	6,20	-21,92	15,03
	8	-14,69	7,91	-20,24	17,67
	10	-15,45	10,91	-21,58	16,83
100	2	-8,99	5,01	-15,59	16,07
	5	-10,65	4,37	-20,59	21,39
	8	-12,20	6,56	-19,03	13,07
	10	-11,85	10,11	-19,68	21,29

GA'nın koşturulma süreleri 100 iş içeren problemler dışında kalan problemler için oldukça düşüktür (Tablo 2). En uzun koşturma süresi 100 iş ve 10 aşama içeren B tipi problemlerde gerçekleşmiştir. Bu problemler için GA çözümünün elde edilmesi ortalama 205,66 saniye (3,43 dakika) sürmüştür. Tablo 2 ve 3'deki sonuçlara bakıldığında GA'nın iyi çözümleri kısa sürede veren etkin bir algoritma olduğu söylenebilir.

Alt sınırdan sapma değerlerinin GA'nın performansı ile ilgili olmayıp altsınırın etkinliğini yitirmesinin bir sonucu olarak büyüdüğü savını test etmek üzere 5 iş içeren problemlerin bütün çözümlerinin listelenmesi (*total enumaration*) yoluna gidilmiştir. Yani A ve B tipine ait 80 adet problemin 5!=120'şer çözümü yaratılarak enbüyük bitirme zamanları hesaplanmış ve her problem için eniyi çözüm bulunmuştur. GA çözümlerinin eniyi çözümlerden sapmaları hesaplanmış ve Tablo 4'te listelenmiştir. GA, 5 iş içeren 80 problemin 50'sinde (yani problemlerin %62,5'unda) eniyi çözümü bulmuştur. 20 problemde (%25) eniyi çözümden ortalama sapma %1'in altında kalmış ve 10 problemde (%12.5) ortalama sapma %1,15 olarak gerçekleşmiştir. Bu bulgular önerilen GA'nın problemi çözmek için etkin bir algoritma olduğunu göstermektedir.

Tablo 4. Beş İşli Problemler İçin GA Çözümlerinin Eniyi Çözümlerden Ortalama Yüzdelik Sapmaları ve T-Testi Değerleri

		A tipi problemler		B tipi problemler	
n	*m*	ort%sapma	t değeri	ort%sapma	t değeri.
5	2	0,86	1,00	1,15	1,50
	5	0,00	-	0,97	1,42
	8	0,00	-	0,00	-
	10	0,00	-	0,00	-

SONUÇ

Bu makalede, oldukça yeni ve ilginç bir probleme bir çözüm yaklaşımı olarak geliştirilen bir GA rapor edilmektedir. Deneysel çalışma sonucu, önerilen GA'nın iyi çözümleri kısa sürede bulan etkin bir yaklaşım olduğu ortaya konmuştur. Bu araştırmada önerilen GA'nın daha da geliştirilmesine yönelik çalışmalarımız devam etmektedir. Aynı probleme farklı sezgisel yaklaşımlar geliştirmek ve bunları GA ile çaprazlamak üzerinde çalışıyoruz. GA'nın ilk neslini tohumlamak üzere çözüm üretme üzerinde çalışıyoruz. Daha önceki bir çalışmada iki aşamalı akış atölyesi için geliştirdiğimiz FSH2 algoritması [20] ile de bir tohumlama sırası yaratma çabasındayız.

KAYNAKÇA

1. Brucker, P. ve Kraemer, A. (1996). Polynomial Algorithms for Resource-Constrained ve Multiprocessor Task Scheduling Problems, *European Journal of Operational Research*, 90, 214-226.
2. Davis, L. (1991). *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York.
3. Dessouky, M.M., Dessouky, M.I., ve Verma, S.K. (1998). Flowshop Scheduling with Identical Jobs and Uniform Parallel Machines, *European Journal of Operational Research*, 109, 620-631.
4. Drozdowski, M. (1996). Scheduling Multiprocessor Tasks – An Overview, *European Journal of Operational Research*, 94, 215-230.
5. Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley.
6. Grangeon, N., Gourgand, M. ve Norre, S. (1998). Heuristic and Stochastic Algorithms for the Hybrid Flow Shop Problem, *16th European Conference on Operational Research (EURO XVI)* Bruxelles.
7. Gupta, J. N. D. (1988). Two-Stage Hybrid Flowshop Scheduling Problem, *Operational Research Society*, 39 (4), 359-364.
8. Gupta, J. N. D. ve Tunc, E.A. (1998). Minimizing Tardy Jobs in a Two-Stage Hybrid Flowshop, *International Journal of Production Research*, 36 (9), 2397-2417.
9. Haouari, M. ve M'Hallah, R. (1997). Heuristic Algorithms for the Two-Stage Hybrid Flowshop Problem, *Operations Research Letters*, 21, 43-53.
10. Holland, J. (1975). Adaptation in Natural and Artificial Systems, The University of Michigan Press, Ann Arbor.
11. Hoogeveen, J.A., Lenstra, J.K., ve Veltman, B. (1996). Preemptive Scheduling in a Two-Stage Multiprocessor Flowshop is NP-Hard, *European Journal of Operational Research*, 89, 172-175.
12. Lee, C-Y., Lei, L. ve Pinedo, M. (1997). Current Trends in Deterministic Scheduling, *Annals of Operations Research*, 70, 1-41.
13. Murata, T., Ishibuchi, H. ve Tanaka, H. (1996). Genetic Algorithms for Flowshop Scheduling Problems, *Computers and Industrial Engineering*, 30 (4), 1061-1071.
14. Oğuz, C., Qi, X.T., ve Fung, Y.F. (2001). Scheduling Multiprocessor Tasks in a Hybrid Flow-Shop Using a Genetic Algorithm, *Working Paper*, The Hong Kong Polytechnic University, Hong Kong SAR.
15. Oğuz, C., Zinder, Y., Ha Do, V., Janiak, A. ve Lichtenstein, M. (2001) Hybrid Flow-Shop Scheduling Problems with Multiprocessor Task Systems, *basım için sunuldu*.
16. Portmann, M.-C., Vignier, A., Dardilhac, D. ve Dezalay, D. (1998). Branch and Bound Crossed with GA to Solve Hybrid Flowshops, *European Journal of Operational Research*, 107, 389-400.
17. Riane, F. ve Artiba, A. (1999). Scheduling Multistage Flowshop Problem: A Brief Review, *International Conference on Industrial Engineering and Production Management*, ISBN 2-930294-02-7, 2, 323-335.
18. Riane, F., Raczy, C. ve Artiba, A. (1999). Hybrid Auto-Adaptable Simulated Annealing Based Heuristic, *Computers and Industrial Engineering*, 37, 277-280.
19. Riane, F., Artiba, A., ve Elmaghraby, S.E. (1998). A Hybrid Three-Stage Flowshop Problem: Efficient Heuristics to Minimize Makespan, *European Journal of Operational Research*, 109, 321-329.
20. Sivrikaya-Şerifoğlu, F. ve Ulusoy, G. (1998). A Bicriteria Two-Machine Permutation Flowshop Problem, *European Journal of Operational Research*, 107, 414-430.
21. Vignier, A., Billaut, J-C., ve Proust, C. (1996). Minimizing Maximum Tardiness in Some Two-Stage Hybrid Flowshops, *Proceedings of the 5th International Workshop on Project Management and Scheduling*, Poznan, Poland, 253-257.
22. Vignier, A. ve Venturini, G. (1996). Resolution of a Hybrid Flowshop with a Parallel Genetic Algorithm, *Proceedings of the 5th International Workshop on Project Management and Scheduling*, Poznan, Poland, 258-261.