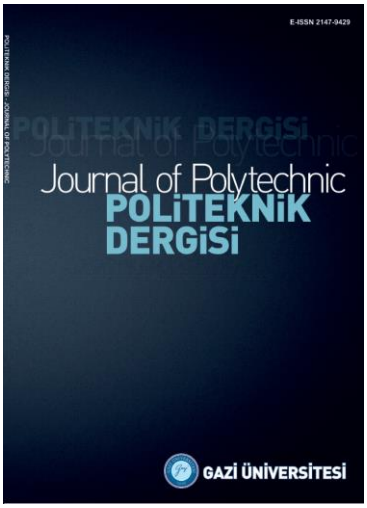


POLİTEKNİK DERGİSİ

JOURNAL of POLYTECHNIC

ISSN: 1302-0900 (PRINT), ISSN: 2147-9429 (ONLINE)
URL: http://dergipark.org.tr/politeknik

Kaynak kısıtlı yazılım proje çizelgeleme probleminin hibrit bir yaklaşım ile çözümü

Hybrid approach for solving resource-constrained software project scheduling problem

***Yazar(lar) (Author(s))**: Nurhan GÜL[1], Nursal ARICI[2]*

ORCID[1]: 0000-0002-9144-4357

ORCID[2]: 0000-0002-4505-1341

To cite to this article: Gül N. ve Arıcı N., "Kaynak Kısıtlı Yazılım Proje Çizelgeleme Probleminin Hibrit Bir Yaklaşım ile Çözümü", ***Journal of Polytechnic***, 28(1): 283-295, (2025).

Bu makaleye şu şekilde atıfta bulunabilirsiniz: Gül N. ve Arıcı N., "Kaynak Kısıtlı Yazılım Proje Çizelgeleme Probleminin Hibrit Bir Yaklaşım ile Çözümü", ***Politeknik Dergisi***, 28(1): 283-295, (2025).

Erişim linki (To link to this article): http://dergipark.org.tr/politeknik/archive

DOI: 10.2339/politeknik.1439675

Kaynak Kısıtlı Yazılım Proje Çizelgeleme Probleminin Hibrit Bir Yaklaşım ile Çözümü

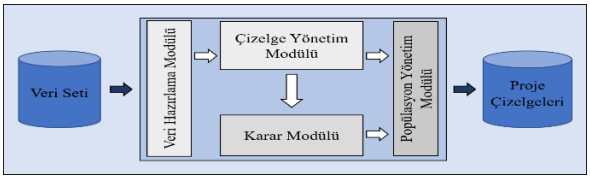
A Hybrid Approach for Solving Resource-Constrained Software Project Scheduling Problem

Önemli noktalar (Highlights)

- *Yazılım proje yönetiminde proje çizelgeleme problemi çözümü için hibrit bir model tasarlanması / Designing a hybrid model to solve the software project scheduling problem*
- *Çoklu insan kaynağı atama yaklaşımıyla probleme alternatif çözüm modeli geliştirilmesi /Developing an alternative solution model for the problem with multiple human resource assignment approach*
- *iMOPSE veri kümesini kullanarak yöntemin başarısının belirlenmesi /Determining the success of the method using the iMOPSE dataset*

Grafik Özet (Graphical Abstract)

Bu araştırma makalesinde kaynak kısıtlı yazılım proje çizelgeleme problemini çözmek amacıyla Genetik Algoritmanın Bozkurt Optimizasyonu, Yapay Arı Kolonisi Algoritması ve kaotik lojistik harita ile desteklendiği hibrit bir yaklaşım önerilmiştir. / In this research paper, a hybrid approach is proposed to solve the resource-constrained software project scheduling problem by supporting the Genetic Algorithm with Grey Wolf Optimization, Artificial Bee Colony Algorithm, and a chaotic logistic map.



Şekil. Mimari **/Figure.** Architecture

Amaç (Aim)

Önerilen hibrit yaklaşım ile yazılım proje yönetiminde proje çizelgeleme sürecine katkı sağlanması amaçlanmıştır. / It is aimed to contribute to the project scheduling process in software project management with the proposed hybrid approach.

Tasarım ve Yöntem (Design & Methodology)

Genetik algoritmanın yakınsama başarısına katkıda bulunmak için bozkurt optimizasyonundaki hiyerarşi modeli, yerel minimuma takılma sorunundan kaçınmak için yapay arı kolonisi algoritmasındaki kâşif arı metodolojisi kullanılmıştır. Daha iyi rastgelelik sağlamak adına kaotik lojistik harita kullanılmıştır./ To contribute to the convergence success of the genetic algorithm, the hierarchy model in grey wolf optimization was used, and the scout bee methodology in the artificial bee colony algorithm was used to avoid the problem of getting stuck in local minima. Chaotic logistic map was used to provide better randomness.

Özgünlük (Originality)

Özgün bir çalışmadır./ It is an original work.

Bulgular (Findings)

Test sonuçlarına göre, tekli insan kaynağı atama modelinde %7, çoklu insan kaynağı atama modelinde ise %15 oranına varan iyileşme gözlenmiştir./ According to test results, an improvement of up to 7% in the one employee assignment model and up to 15% in the multiemployee assignment model has been observed.

Sonuç (Conclusion)

Geliştirilen yöntemin çözüm kararlılığı ve optimum çözümlere yakınsama açısından iyi ve rekabetçi bir performansa sahip olduğunu gösteren sonuçlar elde edilmiştir./ Results have been obtained showing that the developed method has a good and competitive performance in terms of solution stability and convergence to optimum solutions.

Etik Standartların Beyanı (*Declaration of Ethical Standards*)

Bu makalenin yazar(lar)ı çalışmalarında kullandıkları materyal ve yöntemlerin etik kurul izni ve/veya yasal-özel bir izin gerektirmediğini beyan ederler. / The author(s) of this article declare that the materials and methods used in this study do not require ethical committee permission and/or legal-special permission.

Kaynak Kısıtlı Yazılım Proje Çizelgeleme Probleminin Hibrit Bir Yaklaşım ile Çözümü

Araştırma Makalesi / Research Article

Nurhan GÜL[1*], Nursal ARICI[2]

[1]Teknoloji Fakültesi, Bilgisayar Mühendisliği, Gazi Üniversitesi, Türkiye
[2]Gazi Üniversitesi, Türkiye

(Geliş/Received : 19.02.2024 ; Kabul/Accepted : 26.04.2024 ; Erken Görünüm/Early View :22.07.2024)

ÖZ

Bir yazılım proje çizelgesi, proje başarısını doğrudan etkileyen iki temel faktör olan proje süresini ve bütçesini izlemek için gereklidir. Görevler için insan kaynağı ataması, göreve başlama sırası, görev tamamlama süresi, göreve başlamada olası gecikmeler, harcanan para ve kalan bütçe, yazılım proje çizelgesi ile takip edilir. Bu çalışma, yazılım proje yöneticilerine, kaynak kısıtlı yazılım proje çizelgeleme problemini minimum proje süresi, minimum proje bütçesi ve minimum görev bekleme süresi ile çözmek için güçlü bir araç sağlamayı amaçlamaktadır. Bu amaçla, bu çalışmada Genetik Algoritmanın(GA) Bozkurt Optimizasyonu(BO), Yapay Arı Kolonisi Algoritması(YAKA) ve kaotik lojistik harita ile desteklendiği hibrit bir yaklaşım kullanılmıştır. GA'nın yakınsama başarısına katkıda bulunmak için BO'daki hiyerarşi modeli, yerel minimuma takılma sorunundan kaçınmak için YAKA'daki kâşif arı metodolojisi kullanılmıştır. Daha iyi rastgelelik sağlamak adına kaotik lojistik harita kullanılmıştır. Geliştirilen hibrit yöntem Akıllı Çok Amaçlı Proje Çizelgeleme Ortamı (Intelligent Multi Objective Project Scheduling Environment - iMOPSE)'de bulunan veri setleriyle test edilmiştir. Yöntemin sonuçları, literatürdeki yöntemler ile karşılaştırılmış ve parametrik olmayan testler kullanılarak istatistiksel olarak analiz edilmiştir. Test sonuçlarına göre, tekli insan kaynağı atama modelinde %7, çoklu insan kaynağı atama modelinde ise %15 oranına varan iyileşme gözlenmiştir. Sonuçlar, yönteminin çözüm kararlılığı ve optimum çözümlere yakınsama açısından iyi ve rekabetçi bir performansa sahip olduğunu göstermiştir.

Anahtar Kelimeler: Kaynak kısıtlı yazılım proje çizelgeleme problemi, genetik algoritma, bozkurt optimizasyonu, yapay arı kolonisi algoritması, lojistik harita.

A Hybrid Approach for Solving Resource-Constrained Software Project Scheduling Problem

ABSTRACT

A software project schedule management tool is essential for monitoring the project duration and budget, two key factors that will directly affect project success. Assignment of personnel for tasks, initiation order of tasks, task completion time, possible delays in starting a task, and money already spent and the remaining budget are tracked via the software project tool. This study aims to provide software project managers with a powerful tool to solve the Resource-Constrained Software Project Scheduling Problem (RCSPSP) with minimum project duration, minimum project budget, and minimum waiting time of tasks. For this purpose, a hybrid approach is used in this study, in which the Genetic Algorithm (GA) is supported by Grey Wolf Optimization (GWO), Artificial Bee Colony Algorithm (ABC) and chaotic logistic map. The pack hierarchy model in GWO is used to contribute to the convergence success of GA, and scout bee methodology in ABC is adopted into the method to avoid being trapped at local minima. The chaotic logistic map technique is also used to improve randomness. The developed hybrid method has been tested with datasets in the Intelligent Multi-Objective Project Scheduling Environment (iMOPSE). The results are compared with in literature algorithms and statistically analyzed using non-parametrik tests. According to test results, an improvement of up to 7% in the one employee assignment model and up to 15% in the multiemployee assignment model has been observed. The results show that the method has good and competitive performance in terms of solution stability and closeness to optimal solutions.

Keywords: Resource-constrained software project scheduling problem, genetic algorithm, grey wolf optimization, artificial bee colony, logistic map.

1. GİRİŞ (INTRODUCTION)

Yazılım projelerinde başarı, ürün, bütçe ve zaman faktörlerinin bir arada ele alınmasına bağlıdır. Bir projede amaç, ürünü müşterinin istediği özelliklerde, proje bütçesini ve hedeflenen bitiş tarihini aşmadan teslim etmektedir [1], [2].

Başarı kriterleri aynı olsa da yazılım projelerini diğer proje türlerinden ayıran özellikler yazılım projelerinin başarısı için kritik öneme sahiptir:

- Beklenmeyen olaylar meydana geldiğinde proje hedeflerine ulaşma süreci, yazılım ürününün tamamlanmamış parçalarının yeniden planlanmasını gerektirebilir.

*Sorumlu Yazar (Corresponding Author)
e-posta : nurhangul13@gmail.com

- Nihai ürün, proje ekibinin yetkinliği ve müşterinin gereksinimleri ifade etme yeteneği ile doğrudan ilişkilidir. Proje ekibi yetkin olsa bile, ürünün başarısı yine de müşterinin ürünü doğru ve tam olarak tanımlayabilmesine bağlıdır.
- Yazılım proje yönetimi her yeni projede projeye özel olmalıdır. Bu zorunluluk, bazı projeler için proje yöneticisinin tecrübesinin yetersiz kalmasına neden olabilmektedir.

Bu özellikler yazılım proje yönetimini zorlaştırmakta ve bu alanda bir standart geliştirilememesine yol açmaktadır. Bu zorluklara rağmen, nihai hedef her zaman projeyi başarıyla tamamlamak olmalıdır. Bunun için yazılım projeleri iyi bir planlama, izleme ve risk yönetimi gerektirir.

Yazılım proje yöneticisi, projenin planlama, izleme ve yönetim süreçlerinden sorumludur. Bunu yaparken de proje çizelgesinden yararlanır. Yazılım proje çizelgesi, projede görevlendirilen insan kaynaklarının yönetilmesi, proje görevlerinin uygun şekilde yürütülmesi, tahsis edilen proje bütçesinin uygun kullanımı ve proje teslim tarihi için önemli bir kontrol mekanizmasıdır. Kaynakları ve proje kilometre taşları arasındaki karmaşık parametreler ve bağımlılıklar nedeniyle, yazılım proje çizelgeleme problemi için çözüm uzayı oldukça büyüktür. Yazılım proje çizelgeleme probleminin bu özelliği, onu kaynak kısıtlı proje çizelgeleme problemi olarak ele almayı mümkün kılar. Genel olarak kaynak kısıtlı proje çizelgeleme problemi, öncelik kısıtlamalarına ve insan kaynaklarının sınırlı kullanılabilirliğine tabi olan bir dizi görevi minimum bir süre içinde planlamayı amaçlar. Kaynak kısıtlı proje çizelgeleme problemi, polinom olmayan zor yapıda(NP-zor) bir problemdir, arama alanı çok büyüktür ve bir polinomsal zaman algoritması ile optimal olarak çözülemez[3]. Yapay zekâ ve optimizasyon yöntemlerinin kullanılmasını gerektirir.

Literatürdeki birçok çalışma, yazılım proje çizelgeleme problemi için yapay zekâ optimizasyon yöntemleri kullanılarak uygun bir çözüm bulmaya çalışmaktadır. Bu çalışmalardan, bazıları Çizelge 1'de sunulmuştur.

Çizelge 1. Literatür çalışmaları (Literature studies)

Optimizasyon yöntemleri	Yapılan çalışmalar
Yapay arı kolonisi	[4]–[6]
Genetik algoritma	[3], [7]–[10]
Bozkurt optimizasyonu	[11]–[13]
Diğer	[14]–[23]

Bu çalışma, kaynak kısıtlı yazılım proje çizelgeleme problemini verimli bir şekilde çözmek için proje süresini, proje bütçesini ve insan kaynaklarının görev bekleme süresini en aza indirmeyi amaçlayan hibrit bir yaklaşım önermektedir. Önerilen yöntem, BO ve YAKA entegreli gelişmiş bir GA sürümüdür. BO'nun sürü hiyerarşi modeli kullanılarak uygun çözümler arasında daha verimli bir arama gerçekleştirilmiştir. YAKA'daki kaşif arı yaklaşımı, durgunluğa neden olan yerel minimuma yakınsama sorununun üstesinden gelmek için kullanılmıştır. Daha iyi rastgelelik sağlamak için kaotik lojistik harita kullanılmıştır. Sonuçlar, iMOPSE veri setini [26], [27] kullanarak problemi çözen GA, GreedyDo, HAntCO, GRASP, CSM ve DEM algoritmaları [27]–[30] ile karşılaştırılmıştır. Bu karşılaştırmalar sonucunda, etkin ve daha hızlı üretilebilen proje çizelgeleri elde edilebildiği belirlenmiştir.

Makalenin geri kalan kısmı şu şekildedir: Kullanılan veri setinin detayları Bölüm 2'de, kaynak kısıtlı yazılım proje çizelgeleme probleminin çözümünde kullanılan alt fonksiyonlar Bölüm 3'te verilmiştir. Bölüm 4'te elde edilen deneysel sonuçlar, Bölüm 5'te ise bu sonuçlara ilişkin istatistiksel çalışmalara yer almaktadır. Geliştirilen hibrit yönteme ait genel sonuçlar ise Bölüm 6'da verilmiştir.

2. MATERYAL VE YÖNTEM (MATERIAL and METHOD)

2.1. Veri Seti (Dataset)

Problemin çözümünde, her biri farklı büyüklük ve yapıya sahip 36 adet proje verisi ve 6 adet test proje verisinden oluşan iMOPSE veri seti kullanılmıştır [26], [27]. Veri seti içerisinde projede görev yapacak insan kaynakları ve aylık ücretleri, bu insan kaynaklarının yetkinlik seviyeleri, projede gerçekleştirilmesi zorunlu olan görevlerin gerektirdiği asgari yetkinlik düzeyi ve görevlerin tamamlanması için gerekli süre bilgileri yer almaktadır. Veri setinde her bir görev için başlama zamanı ve görevin tamamlanma süresi tanımlıdır. Görevlerin birbirine bağımlılıkları, görevlerin başlama sırasını belirler. Her bir insan kaynağı için bir yetkinlik seviyesi tanımlanmıştır ve bu yetkinlik seviyeleri insan kaynaklarının hangi görevleri icra edebileceğini belirler.

36 adet veri seti içerisinden literatürde en çok kullanılan ve Çizelge 2'de belirtilen on adet proje verisi bu çalışmada kullanılmak üzere seçilmiştir.

Çizelge 2. Kullanılan veri setleri (Used dataset)

Veri Seti Adı	Görev	İnsan Kaynağı	Öncel Sayısı	Yetkinlik Türü
100_5_22_15	100	5	22	15
100_10_26_15	100	10	26	15
100_10_47_9	100	10	47	9

Çizelge 3. Kullanılan veri setleri (Used dataset)

Veri Seti Adı	Görev	İnsan Kaynağı	Öncel Sayısı	Yetkinlik Türü
100_20_46_15	100	20	46	15
100_20_65_9	100	20	65	9
200_10_50_15	200	10	50	15
200_20_54_15	200	20	54	15
200_20_55_9	200	20	55	9
200_40_91_15	200	40	91	15
200_40_133_15	200	40	133	15

3. YAZILIM PROJE ÇİZELGELEME (SOFTWARE PROJECT SCHEDULING)

Etkin bir yazılım proje çizelgesi oluşturmak için projenin görev bağımlılıklarının belirlenmesi, görevlerin insan kaynaklarının yetkinlik ve deneyimlerine göre atanması ve görevlerin yaklaşık tamamlanma sürelerinin tahmin edilmesi gerekmektedir. Bu yapılırken, proje bütçesi ve hedeflenen proje tamamlanma süresi aşılmamalıdır. Tüm bu faktörler arama uzayını çok büyük (NP-zor) yapar ve çözüm süresini azaltmak için çeşitli yapay zeka teknikleri uygulanabilir[31], [32].

Bu çalışmada, yazılım proje çizelgeleme problemlerinin çözümü için hibrit bir yaklaşım kullanılmıştır. Rastgele fonksiyonlara ihtiyaç duyan süreçler için kaotik lojistik harita kullanılmıştır. İnsan kaynakları, görevler ve bu varlıklar arasındaki ilişkiler kullanılarak, minimum tamamlanma süresi, minimum bütçe ve insan kaynaklarının minimum görev bekleme süresi hedeflenerek en uygun yazılım proje çizelgesi oluşturma amaçlanmıştır. Sonuçları karşılaştırmak için iMOPSE veri seti kullanılmış olup, üç amaç fonksiyonu ayrı ayrı ele alınmıştır.

3.1. Kaotik Rastgele Sayı Üretimi(Chaotic Random Number Generation)

Kaotik sistemler, doğrusal olmayan dinamik sistemlerin bir parçasıdır. Başlangıç koşullarına aşırı duyarlıdırlar ve kaotik bölgede tahmin edilemez davranışlar gösterirler[33]. Kaotik sistemlerin bu özelliği, rastgele sayı üretiminde kaotik sistemlerin de tercih edilmesini sağlamaktadır[34], [35]. Yapay zeka optimizasyon algoritmalarında bu davranışlar başlangıç popülasyonlarının konumlandırılması için de bir alternatif olmaktadır[36], [37].

Bu çalışmada rastgele sayı üretimini gerektiren tüm süreçler için rastgele sayı üreteci olarak Eş. (1)'de ifade edilen lojistik harita kullanılmıştır. Bu eşitlikte $3.57<r\leq 4$ olmak üzere, r büyüme oranını temsil etmektedir.

$$x_{n+1} = rx_n(1 - x_n) \quad (1)$$

3.2. Uygunluk Değeri Hesaplama(Fitness Calculation)

Uygunluk değeri, problemin çözümünde hedeflenen değere ne kadar yakınsandığını gösteren bir parametredir. Amaç fonksiyonları optimize edilirken, her biri için minimum değere yakınsanmaya çalışılır. Görevler icra edilirken, her bir fonksiyonun o ana kadar elde ettiği değer belirlenir. Son göreve gelindiğinde, artık proje tamamlanmıştır ve projenin bütçesi, süresi ve görev bekleme süresi için nihai değere ulaşılır. Bu aşamada, hesaplanan uygunluk değeri ile değerlendirme yapılarak, sonucun iyileşip iyileşmediği belirlenir. Eş. (2), Eş. (3) ve Eş. (4)'te uygunluk değerlerinin nasıl hesaplandığı belirtilmiştir. Bu eşitliklerde $TS\ddot{u}re, TB\ddot{u}t\c{c}e$ ve $TBekle$ değerleri görevlerin her biri için elde edilen değerleri, $PS\ddot{u}re, PB\ddot{u}t\c{c}e$ ve $PBekle$ değerleri ise, projeye ait toplam değerleri ifade etmektedir.

$$PS\ddot{u}re = \sum_{a=1}^{p} TS\ddot{u}re_a \qquad uygunluk_{S\ddot{u}re} = {}^{1}/_{PS\ddot{u}re} \quad (2)$$

$$PB\ddot{u}t\c{c}e = \sum_{a=1}^{p} TB\ddot{u}t\c{c}e_a \qquad uygunluk_{B\ddot{u}t\c{c}e} = {}^{1}/_{PB\ddot{u}t\c{c}e} \quad (3)$$

$$PBekle = \sum_{a=1}^{p} TBekle_a \qquad uygunluk_{GBekle} = {}^{1}/_{PBekle} \quad (4)$$

3.3. Hibrit Bir Yaklaşım ile Yazılım Proje Çizelgesi Oluşturma (Project Scheduling with a Hybrid Approach)

Bu çalışmada, kaynak kısıtlı yazılım proje çizelgeleme probleminin çözümü için BO ve YAKA ile desteklenmiş GA kullanılmıştır. Standart BO yöntemindeki birey hiyerarşisi alfa, beta, delta ve omega şeklindedir. Çalışmada yöntemin bu hiyerarşik modeli kullanılmış ve kurtlarla temsil edilen proje çizelgelerinin değerlendirilmesi uygunluk fonksiyonu kullanılarak gerçekleştirilmiştir.

Hiyerarşideki en baskın kurt olan alfa, o ana kadar üretilmiş en iyi proje çizelgesini temsil eder. Beta ve delta kurtları, en iyi ikinci ve üçüncü proje çizelgesi ile, omega olarak adlandırılan sürünün diğer kurtları ise sıralamada ilk üçe giremeyen çizelgeler ile ilişkilendirilmiştir.

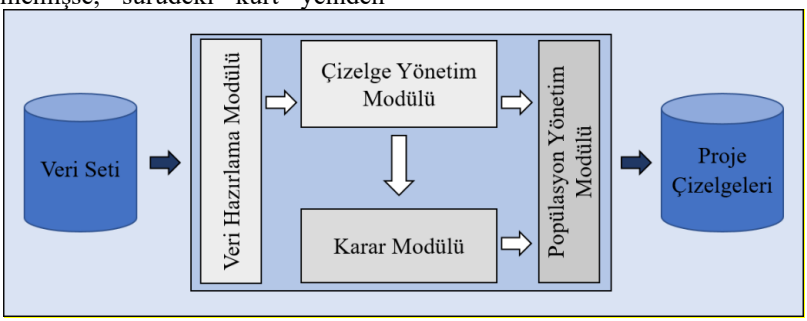
Popülasyon çeşitliliği GA'nın temel modülleri olan mutasyon ve çaprazlama yöntemleri kullanılarak sağlanmıştır. YAKA'daki yiyecek arama metodu, olası durgunluk probleminin çözümünde kullanılmıştır.

Geliştirilen hibrit yöntemde her kurt bir proje çizelgesini temsil eder. Görevler ile ilişkilendirilmiş kısıtlamalar dikkate alınarak N adet proje çizelgesi oluşturulur. Sürüdeki hiyerarşi göz önünde bulundurularak alfa, beta ve delta kurtları ile en iyi üç proje çizelgesi ilişkilendirilir. Diğer proje çizelgeleri rastgele olarak omegalara dağıtılır ve omegalar diğer üç kurtla ilişkili proje çizelgesi değerlerine yakınsamaya çalışır. Sürüdeki çeşitlilik GA'daki iki noktalı çaprazlama yöntemi kullanılarak gerçekleştirilir. Çaprazlama boyutu ve mutasyon sayısı, görev sayısına bağlı olarak belirli bir sınırın altında kalacak şekilde rastgele belirlenir. Geçerli çizelgeler üzerinde çaprazlama ve mutasyon işlemleri gerçekleştirildiğinden ve bu işlemler gerçekleştirilirken görevlerle ilgili kısıtlama kuralları dikkate alındığından, ortaya çıkan çizelgeler de geçerli çizelgeler olmaktadır.

Yönteme, YAKA'daki kaşif arı fazı entegre edilmiştir. Amaç fonksiyonu ile hedeflenen değer belirli bir süre boyunca iyileştirilememişse, sürüdeki kurt yeniden üretilerek olası durgunluk ve yerel minimum ve maksimuma takılma sorunları önlenmeye çalışılmıştır.

Geliştirilen sistemin mimarisi Şekil 1'de yer almakta olup, yazılım dört ana modülden oluşmaktadır:

- **Veri hazırlama modülü**, veri kümesini diğer modüllerin kullanabileceği uygun bir veri biçimine dönüştürür.
- **Çizelge yönetim modülü**, proje verilerini kullanarak, kaynak kısıtlı yazılım proje çizelgeleme problemine uygun şekilde proje çizelgeleri oluşturur.
- **Karar modülü**, üretilen proje çizelgelerinin uygunluk fonksiyonlarına göre kalite değerlendirmesini yapar.
- **Popülasyon yönetim modülü**, geliştirilen yöntem doğrultusunda popülasyon güncellemelerini gerçekleştirir.



Şekil 1. Sistem mimarisi (System architecture)

3.4. Çoklu İnsan Kaynağı - Görev Ataması (Multiemployee Task Assignment)

Çoklu insan kaynağı - görev atamasının proje süresine katkısını değerlendirmek amacıyla, her bir görev için en az bir insan kaynağı atanması yaklaşımı da incelenmiştir. Bu yaklaşımda, görev atama sürecinde her bir görev için, o göreve yetkinlik ve zaman açısından uygun olanlar insan kaynaklarının bir kısmı veya tümü seçilmiştir. Burada görevin gerçekleştirilme süresinin azaltılması ve bunun sonucunda da toplam proje süresinin iyileştirilmesi amaçlanmıştır. Bu yöntemin test amaçlı olarak 10_3_5_3 veri setine uygulanması ile, bazı görevler için görev tamamlanma süresini azaltacak şekilde birden fazla insan kaynağı görevlendirilmesinin gerçekleşebildiği ve üretilen yazılım çizelgelerinin sürelerinin de iyileştiği görülmüştür.

Çizelge 4'te 10_3_5_3 isimli proje verisi kullanılarak görevlere en az bir insan kaynağı atanmasına ilişkin iki adet proje çizelgesi örneği gösterilmiştir. Her bir göreve tek bir insan kaynağı atanması sonucunda 146 saat olarak elde edilen proje süresi, her bir görev için en az bir insan kaynağı atanması sonucunda 'Yazılım Çizelgesi 1' için 100 saat, 'Yazılım Çizelgesi 2' için ise 92 saat olarak elde edilmiştir.

Çizelge 4. Çoklu insan kaynağı - görev ataması (Multiemployee-one task assignment)

Çizelgeler												Proje Süresi*
'Yazılım Çizelgesi 1'	Görev	0	1	2	3	4	5	7	9	6	8	100
	İnsan kaynağı	0,1	1	2	3	2	0	1,2	0	2,0	0,1	
'Yazılım Çizelgesi 2'	Görev	0	1	2	3	4	5	7	6	8	9	92
	İnsan kaynağı	0	1	2	0	2	0	1	2	0,1	2	

**Süre saat cinsindendir.*

4. DENEYSEL SONUÇLAR(EXPERIMENTAL RESULTS)

Kaynak kısıtlı yazılım proje çizelgeleme problemi, proje süresi, proje bütçesi ve görevlerin bekleme süresi olmak üzere üç amaç fonksiyonu kullanılarak aşağıda belirtilen kriterler dikkate alınarak hibrit bir yöntem ile çözülmüştür:

- Görevlerin önceliği

- Görev-yetkinlik seviyesi ilişkisi
- İnsan kaynağı yetkinlik seviyesi
- İnsan kaynağı ücreti

Problemin çözümünde ortaya çıkan sonuçlar, gerçek hayattaki projelerle ve çalışmamızda çalışılan problemle uyumlu olması sebebiyle 36 adet farklı boyutta proje verisi içeren iMOPSE veri seti kullanılarak analiz edilmiştir[26], [27]. Problemin çözümü için kullanılan yöntemler ve kısaltmalar Çizelge 4'te açıklanmıştır.

Çizelge 5. Önerdiğimiz yöntemler (Developed methods)

Yöntem		Açıklama
THibrit	Tek insan kaynağı ataması	Hibrit modelde her bir göreve bir insan kaynağı atanır.
TGA		GA yönteminde her bir göreve bir insan kaynağı atanır.
ÇHibrit	Çoklu insan kaynağı ataması	Hibrit modelde her bir göreve en az bir insan kaynağı atanır.
ÇGA		GA yönteminde her bir göreve en az bir insan kaynağı atanır.

Çizelge 6′ da iMOPSE veri setinden seçilmiş on adet proje verisi kullanılarak, geliştirilen hibrit yöntem ve bu yöntemin temelinde kullanılan GA ile elde edilen sonuçlar, literatürdeki benzer çalışmalarla kıyaslanmıştır[27]–[30]. Gerçekleştirilen deneylerde popülasyon boyutu 100, maksimum çaprazlama ve mutasyon oranı görev sayısının yüzde 20'si olarak alınmıştır. Çeşitliliği sağlayabilmek adına çaprazlama ve mutasyon oranının her döngüde bu limitin altında kalması sağlanmıştır.

Çizelge 6'te belirtilen sonuçlara göre,

- THibrit 7 proje verisi için en iyi sonucu üretmiştir. 2 proje verisinde literatürdeki benzer çalışmalardan daha iyi sonuç üretmiş ve TGA yöntemi ile elde edilen en iyi sonuca yaklaşmıştır.
- Arama uzayının oldukça büyük olduğu 200_40_133_15 veri setinde GA ve Hibrit yöntemler yerel minimuma yakınsamışlardır.
- Görevlere birden fazla insan kaynağının atandığı yöntemlerde çözüm uzayı oldukça büyük olmaktadır. Bu yöntemlerden ÇHibrit yönteminin ÇGA'ya oranla daha iyi yakınsama hızına sahip olduğu görülmüştür.

Çizelge 6. Proje sürelerinin karşılaştırılması (Comparison of the project durations)*

	Literatürdeki Yöntemler					Önerdiğimiz Yöntemler			
Veri Seti	GreedyDO	HAntCO	GRASP	CSM	DEM	TGA	THibrit	ÇGA	ÇHibrit
100_5_22_15	630	504	503	488	485	**485**	**485**	481	**449,5**
100_10_26_15	370	266	251	247	244	240	**238**	232	**227**
100_10_47_9	549	297	263	268	270	260	**259**	254	**257,25**
100_20_46_15	394	194	170	188	181	166	**164**	157	**127,24**
100_20_65_9	408	180	**135**	174	ND	143	**135**	137,31	**124,31**
200_10_50_15	763	529	524	500	496	492	**487**	**489**	**489**
200_20_54_15	488	336	304	329	301	**280**	287	273	**255,44**
200_20_55_9	999	313	257	304	ND	**253**	256	424,27	334,45
200_40_91_15	519	207	**153**	197	ND	165	**153**	168	**145**
200_40_133_15	512	214	**163**	ND	205	174	165	166,05	177,62

*Süreler saat cinsindendir.

Çizelge 5'te görüldüğü üzere, geliştirilen hibrit yöntemler arama uzayında etkin arama gerçekleştirerek benzer yöntemlere oranla daha başarılı sonuçlar elde etmektedir. Görevlere çoklu insan kaynağı ataması seçeneğiyle de, proje süresinin daha da iyileştirilmesine olanak sağlanmaktadır. Literatürdeki benzer çalışmalarla kıyaslandığında, tekli insan kaynağı atama modelinde %7, çoklu insan kaynağı atama modelinde ise %15 oranına varan iyileşme gözlenmiştir.

100_20_46_15 proje verisi kullanılarak TGA ve THibrit yöntemleriyle elde edilen sonuçlar Çizelge 6 ve Çizelge 7'de gösterilmektedir. Her bir görev için yalnızca bir insan kaynağının görevlendirildiği bu yöntemlerde, proje süresi ve görev bekleme sürelerinin optimizasyonunda THibrit yönteminin, bütçe optimizasyonunda ise TGA yönteminin daha iyi sonuçlar ürettiği görülmüştür.

Çizelge 7. TGA* kullanarak proje süresi, proje bütçesi ve görev bekleme süresinin (GBekle) karşılaştırılması (Project duration, budget, and task waiting time optimization using one employee assignment per task for GA)

Süre Optimizasyonu			Bütçe Optimizasyonu			GBekle Optimizasyonu		
Süre	Bütçe	B. Zamanı	Süre	Bütçe	B.Zamanı	Süre	Bütçe	B.Zamanı
164	146865	5621	598	71491,2	21383	164	146605,8	5269
165	148824,9	5691	627	71576,7	20516	172	148319,9	5278
166	144743,7	6006	667	71576,7	20601	181	145317,1	5304
168	147439,7	5694	610	71714,7	20245	171	149642,8	5312
171	150411,2	5702	596	71727	20422	178	144552,4	5320
171	144919	5666	596	72006,1	20723	194	146636,9	5333
172	143792,2	5649	599	72006,1	20279	200	142517,5	5341
172	149548,5	6021	693	72006,1	21327	217	146609,4	5345
173	152379,1	6006	631	72144,1	20143	179	146660,5	5359
173	148320,6	5960	663	72240,1	20972	190	145427,9	5364

* 10 ayrı çalıştırma

Çizelge 8. THibrit* kullanarak proje süresi, proje bütçesi ve görev bekleme süresinin (GBekle) karşılaştırılması (Project duration, budget, and task waiting time optimization using one employee assignment per task for hybrid method)

Süre Optimizasyonu			Bütçe Optimizasyonu			GBekle Optimizasyonu		
Süre	Bütçe	B. Zamanı	Süre	Bütçe	B.Zamanı	Süre	Bütçe	B.Zamanı
164	142172,7	5572	595	79174,8	18140	204	143402,1	5228
164	143281,4	5773	526	79574,8	15876	204	139202,9	5237
165	145825,3	5758	522	79652,8	16879	192	142046,3	5268
168	141859,7	5916	594	79871,5	15954	181	144252,4	5275
168	140119	5615	486	80149,9	16889	184	141086	5282
168	141201	5432	521	80646,4	17497	192	144039	5286
168	145508,5	5704	445	81063,9	14115	187	143031,3	5292
170	147548,8	5636	454	81129,5	14816	194	140458,4	5310
171	142572,7	5896	521	81378,6	16807	193	143628,6	5343
171	144088,7	5734	635	81386,4	18536	186	143046,5	5359

* 10 ayrı çalıştırma

THibrit ve TGA yöntemleri ile gerçekleştirilen testlerdeki aynı parametreler kullanılarak bir görev için en az bir insan kaynağının atandığı çoklu atama yöntemleri olan ÇGA ve ÇHibrit yöntemleri 100_20_46_15 veri setinde test edilmiştir ve sonuçlar Çizelge 8 ve Çizelge 9'da gösterilmiştir. Bu yöntemlerde arama uzayı oldukça büyüktür. Her üç amaç fonksiyonunun optimizasyonunda, ÇHibrit yönteminin ÇGA yöntemine göre daha iyi sonuçlar elde ettiği görülmüştür.

Çizelge 9. ÇGA* kullanarak proje süresi, proje bütçesi ve görev bekleme süresinin (GBekle) karşılaştırılması (Project duration, budget, and task waiting time optimization using multi employee assignment per task for GA)

Süre Optimizasyonu			Bütçe Optimizasyonu			GBekle Optimizasyonu		
Süre	Bütçe	B.Zamanı	Süre	Bütçe	B.Zamanı	Süre	Bütçe	B.Zamanı
157,00	153679,15	5176,00	596	71491,2	20902	191,72	142639,78	5105,84
170,70	151439,49	6022,00	604	71491,2	20772	246,75	143236,77	5307,97
184,29	154740,12	6192,81	610	71491,2	20449	206,50	134271,77	5424,78
193,93	140806,08	7282,04	614	71491,2	20858	199,80	137700,53	5447,28
198,17	142452,46	6760,73	616	71491,2	20076	203,00	142798,34	5729,08
198,92	141399,78	7018,19	619	71491,2	20711	223,41	140881,54	5761,97

Çizelge 8. (devam) ÇGA* kullanarak proje süresi, proje bütçesi ve görev bekleme süresinin (GBekle) karşılaştırılması ((continued)Project duration, budget, and task waiting time optimization using multi employee assignment per task for GA)

203,32	142265,47	7853,74	626	71491,2	20776	191,50	141742,85	5982,20
204,83	142589,40	7441,37	628	71491,2	20569	228,50	147943,17	6253,51
208,73	140610,68	8140,63	631	71491,2	20652	208,92	142875,12	6254,08
216,41	140157,49	8216,35	631	71491,2	20284	206,96	142376,63	6460,96

* 10 ayrı çalıştırma

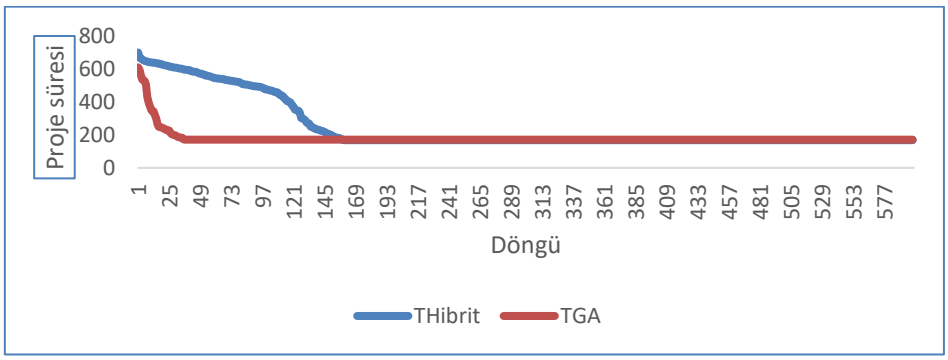
Çizelge 10. ÇHibrit* kullanarak proje süresi, proje bütçesi ve görev bekleme süresinin (GBekle) karşılaştırılması (Project duration, budget, and task waiting time optimization using multi employee assignment per task for hybrid method)

Süre Optimizasyonu			Bütçe Optimizasyonu			GBekle Optimizasyonu		
Süre	Bütçe	B.Zamanı	Süre	Bütçe	B.Zamanı	Süre	Bütçe	B.Zamanı
127,239	145791	4257,47	601	70061,1	19817	142,25	134909,4	3224,002
127,476	141176,8	4166,641	641	70061,1	19996	140,679	136398,7	3330,85
130	146866,6	4087,417	659	70061,1	20188	123,667	142431,4	3420,3
130,594	141702,7	4673,636	601	70061,1	20249	150,5	145813,4	3441,082
131,143	143695,1	4295,732	601	70061,1	19822	164	141176	3595,908
135,333	146217,1	4986,842	601	70061,1	19805	155,95	140684,3	3841,297
139,667	140704	5023,44	643	70061,1	19979	142	137906,9	3853,036
141,348	146092,1	4419,727	626	70061,1	19925	151,643	145863,4	3969,697
141,8	140380,7	4874,199	624	70061,1	20205	152,667	134163,4	4001,135
143,833	141040,7	4723,128	689	70061,1	20963	164	147769,7	4334,843

* 10 ayrı çalıştırma

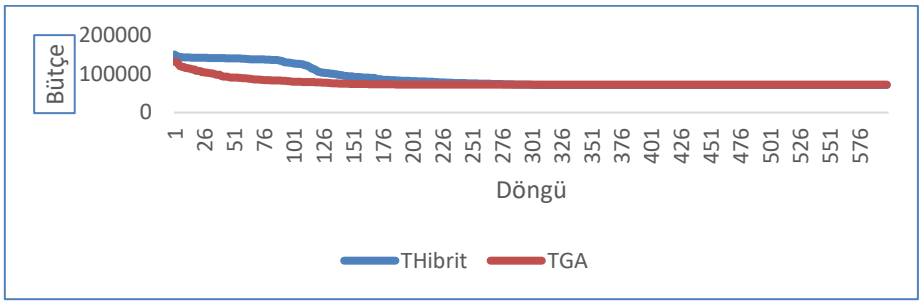
Her bir görev için tek bir insan kaynağının atandığı yöntemlerle kıyaslandığında çoklu insan kaynağı ataması yöntemleri, proje süresini, proje bütçesini ve insan kaynaklarının görev bekleme süresini daha da azaltmaktadır. Fakat, Çizelge 5'te de görüldüğü gibi bazı durumlarda çözüm yerel minimuma takılabilmektedir.

Gerçekleştirilen deneysel çalışmalarda 100_20_46_15 veri seti kullanılarak yöntemlerin yakınsama hızları da incelenmiştir ve elde edilen sonuçlar Şekil 2 ve Şekil 3'te gösterilmiştir. Her iki atama yönteminde GA'nın hızlı yakınsama davranışı gösterdiği gözlenmiş olup, hızlı yakınsama sonucunda da yerel minimuma takıldığı görülmüştür. Arama uzayının daha karmaşık olduğu çoklu insan kaynağı atama yöntemlerinde ise, ÇHibrit yönteminin ÇGA yöntemine göre daha iyi performans gösterdiği görülmüştür.

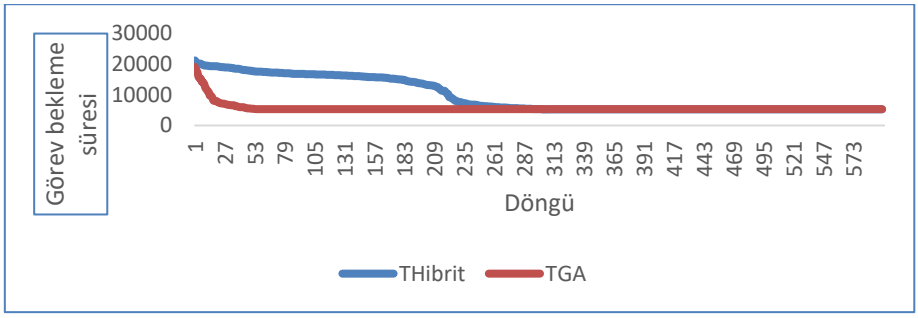


Proje süresi

Şekil 2. Tekli insan kaynağı atama modelinde yakınsama hızı (Convergence speeds for one employee-task assignment)

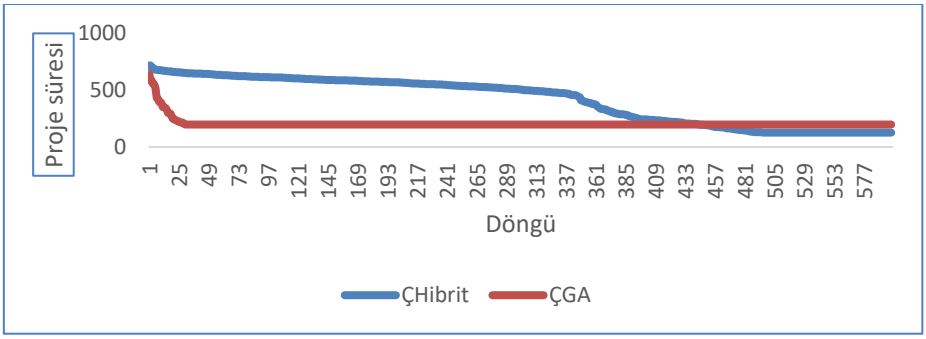


Bütçe

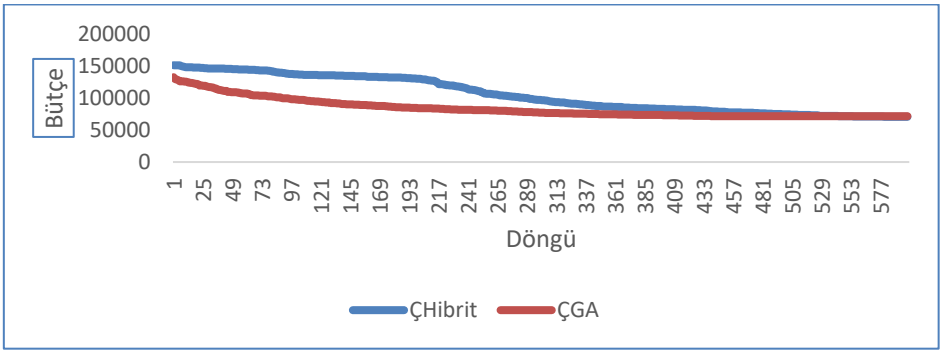


Görev bekleme süresi

Şekil 3. (Devam). Tekli insan kaynağı atama modelinde yakınsama hızı (Convergence speeds for one employee-task assignment)

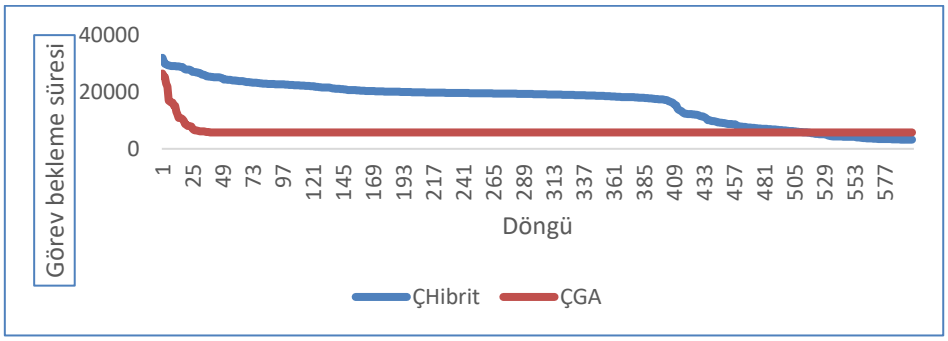


Proje süresi



Bütçe

Şekil 4. Çoklu insan kaynağı atama modelinde yakınsama hızı (Convergence speeds for multiemployee-task assignment)



Görev bekleme süresi

Şekil 5. (Devam). Çoklu insan kaynağı atama modelinde yakınsama hızı (Convergence speeds for multiemployee-task assignment)

5. SONUÇLARIN İSTATİSTİKSEL ANALİZİ (STATISTICAL ANALYSIS OF THE RESULTS)

Bu çalışmada ayrıca, proje süreleri bakımından önerdiğimiz hibrit modele ilişkin algoritmaları literatürdeki algoritmalarla karşılaştırarak istatistiksel olarak anlamlı bir fark olup olmadığı araştırılmıştır. Bu kesimde bu araştırmaya ait bulgular açıklanmaktadır.

Çizelge 5'te belirtilen ve farklı yöntemlerle optimize edilen proje süreleri açısından istatistiksel olarak anlamlı bir fark olup olmadığı, parametrik olmayan bir test olan Tek Örneklem İşaret Testi (TÖİT) ile araştırılmıştır. η fark değerlerinin medyanı olmak üzere, Eş. (5)'te verilen hipotez TÖİT kullanılarak test edilmiştir.

Sıfır hipotez	$H_0: \eta = 0$	(5)
Alternatif hipotez	$H_1: \eta \neq 0$	

Çizelge 11. Proje süresi için TÖİT* (TÖİT for Project duration)

Algoritmalar	Sayı < 0	Sayı = 0	Sayı > 0	P-Değeri	Karar
TGA_ THibrit	2	1	7	0.180	Anlamlı fark yok
ÇGA_ÇHibrit	2	1	7	0.180	Anlamlı fark yok
TGA_ÇGA	2	0	8	0.109	Anlamlı fark yok
TGA_ÇHibrit	2	0	8	0.109	Anlamlı fark yok
THibrit_ÇGA	5	0	5	1.000	Anlamlı fark yok
THibrit_ÇHibrit	3	0	7	0.344	Anlamlı fark yok
GreedyDO_TGA	0	0	10	0.002	Anlamlı fark var
GreedyDO_THibrit	0	0	10	0.002	Anlamlı fark var
GreedyDO_ÇGA	0	0	10	0.002	Anlamlı fark var
GreedyDO_ÇHibrit	0	0	10	0.002	Anlamlı fark var
HAntCO_TGA	0	0	10	0.002	Anlamlı fark var
HAntCO_THibrit	0	0	10	0.002	Anlamlı fark var
HAntCO_ÇGA	1	0	9	0.021	Anlamlı fark var
HAntCO_ÇHibrit	1	0	9	0.021	Anlamlı fark var
GRASP_TGA	3	0	7	0.344	Anlamlı fark yok
GRASP_THibrit	1	2	7	0.070	Anlamlı fark yok
GRASP_ÇGA	4	0	6	0.754	Anlamlı fark yok
GRASP_ ÇHibrit	2	0	8	0.109	Anlamlı fark yok
CSM_TGA	0	0	9	0.004	Anlamlı fark var
CSM_THibrit	0	0	9	0.004	Anlamlı fark var
CSM_ÇGA	1	0	8	0.039	Anlamlı fark var
CSM_ÇHibrit	1	0	8	0.039	Anlamlı fark var
DEM_TGA	0	1	6	0.031	Anlamlı fark var
DEM_THibrit	0	1	6	0.031	Anlamlı fark var

Çizelge 12. (devam) Proje süresi için TÖİT* ((continued)TÖİT for Project duration)

DEM_ÇGA	0	0	7	0.016	Anlamlı fark var
DEM_ÇHibrit	0	0	7	0.016	Anlamlı fark var

*$\alpha = 0{,}05$

Çizelge 11'a göre;

- Algoritmalar sütunu karşılaştırılan algoritmaları göstermektedir.
- 'Sayı<0' sütunu, sıfırdan küçük (negatif) fark değerine sahip örneklemdeki gözlem sayısını göstermektedir.
- 'Sayı>0' sütunu, sıfırdan büyük (pozitif) fark değerine sahip örneklemdeki gözlem sayısını göstermektedir.
- 'Sayı=0' sütunu, sıfır fark değerine sahip örneklemdeki gözlem sayısını göstermektedir.
- Eğer P-Değeri > α ise, çalışılan örnek algoritmalar arasında önemli bir fark yoktur. Aksi takdirde, sıfır hipotez reddedilir ve bu karşılaştırılan algoritmalar arasında önemli bir fark olduğu anlamına gelir. Karşılaştırmaların sonuçları 'Karar' sütununda belirtilmektedir.

Çizelge 11'da yer alan TGA_THibrit karşılaştırması incelendiğinde, $\mathrm{P-Değeri} > \alpha$ (0.180>0.05) olduğundan, algoritmaların proje sürelerinin istatistiksel olarak anlamlı bir farklılık taşımadığı, GreedyDO_TGA karşılaştırması incelendiğinde, $\mathrm{P-Değeri} < \alpha$ (0.002<0.05) olduğundan GreeedyDO ve TGA algoritmalarının proje sürelerinin ise önemli farklılık taşıdığı görülmektedir. Sayı >0 fark değerli gözlemlerin sayısı, Sayı < 0 (0) ve Sayı = 0 (0) fark değerli gözlemlerin sayısından büyük olduğundan, TGA algoritmasının ürettiği proje süresinin GreedyDO algoritmasının ürettiği proje süresinden daha küçük olduğu sonucuna varılmaktadır.

Tüm sonuçlar değerlendirildiğinde, geliştirilen yöntemlerin GRASP algoritması haricindeki diğer algoritmalara oranla istatistiksel olarak daha başarılı olduğu belirlenmiştir.

TGA-THibrit ve ÇGA-ÇHibrit yöntemleri ile elde edilen Çizelge 7, Çizelge 8,
Çizelge **9** ve
Çizelge **10**'daki sonuçlar kullanılarak proje süresi, proje bütçesi ve boşta bekleme süreleri parametrik olmayan Mann-Whitney testi kullanılarak karşılaştırılmıştır.
Değişkenlere ait kısaltmalar
Çizelge **13**'de test sonuçları ise

Çizelge 14'de belirtilmiştir. Test edilen $H_0: \eta = 0$ sıfır hipotezi, iki algoritmanın ilgili değişken medyan değerleri arasında anlamlı bir fark olmadığını ortaya koymaktadır.

Çizelge 13. Değişken kısaltmaları (Variable abbreviations)

Sembol*	Değişken
Süre_1	Çizelge 7- Süre Optimizasyonu: Süre
Süre_2	Çizelge 8- Süre Optimizasyonu: Süre
Bütçe_1	Çizelge 7- Bütçe Optimizasyonu: Bütçe
Bütçe_2	Çizelge 8- Bütçe Optimizasyonu: Bütçe
B.Süresi_1	Çizelge 7- Boşta Bekleme Süresi Optimizasyonu: B.Süresi
B.Süresi_2	Çizelge 8- Boşta Bekleme Süresi Optimizasyonu: B.Süresi
Süre_3	Çizelge **9** - Süre Optimizasyonu: Süre
Süre_4	Çizelge **10** - Süre Optimizasyonu: Süre
Bütçe_3	Çizelge **9** - Bütçe Optimizasyonu: Bütçe
Bütçe_4	Çizelge **10** - Bütçe Optimizasyonu: Bütçe
B.Süresi_3	Çizelge **9** - Boşta Bekleme Süresi Optimizasyonu: B.Süresi
B.Süresi_4	Çizelge **10** - Boşta Bekleme Süresi Optimizasyonu: B.Süresi

*TGA ve THibrit arasındaki fark

Çizelge 14. Mann-Whitney test sonuçları* (Mann-Whitney test results)

	Medyan	Mann-Whitney U	P-Değeri	Kara
Süre_1	171	30.50	0.143	Anlamlı fark yok
Süre_2	168			
Bütçe_1	71866.6	.000	0.000	Anlamlı fark
Bütçe_2	80398.1			
B.Süresi_1	5326.5	25.500	0.063	Anlamlı fark yok
B.Süresi_2	5284.0			

Çizelge 15. (devam) Mann-Whitney test sonuçları* ((continued)Mann-Whitney test results)

Süre_3	198.545	.000	0.000	Anlamlı fark
Süre_4	133.238			
Bütçe_3	71491.2	-	-	Fark**
Bütçe_4	70061.1			
B.Süresi_3	5745.52	.000	0.000	Anlamlı fark
B.Süresi_4	3718.60			

*$\alpha = 0{,}05$

** Her iki değişkende de değişkenlik olmadığından (Bütçe_3, Bütçe_4), test gerçekleştirilememiştir.

Çizelge 12'deki 1. satırın anlamı şudur: $\mathrm{P-Değeri} > \alpha$ (0.143>0.05) olduğundan TGA ve THibrit yöntemleri ile elde edilen proje süreleri arasından anlamlı bir fark bulunmamaktadır. Medyan sütununda her iki algoritmanın proje sürelerinin medyan değerleri, Whitney U sütununda ise, Whitney Uistatistik değeri yer almaktadır. Sıfır hipotezinin reddedildiği $\mathrm{P-Değeri} < \alpha$ durumunda, sonuçlar medyan değeri daha küçük olan algoritma lehinde yorumlanmıştır. Diğer satırların yorumlanması da benzer mantık kullanılarak yapılmalıdır.

Parametrik olmayan Mann-Whitney test sonuçlarına göre, bütçe optimizasyonunda TGA yönteminin THibrit yönteminden daha başarılı sonuçlar ürettiği belirlenmiştir. Diğer taraftan, ÇHibrit yönteminin proje süresi ve boşta bekleme süresi optimizasyonlarında ÇGA yöntemine göre daha başarılı sonuçlar ürettiği tespit edilmiştir.

6. SONUÇLAR (CONCLUSIONS)

Yazılım proje çizelgesi, projelerin başarı ile tamamlanmasında önemli etkiye sahip bir araçtır. Bu çalışmada, proje süresi, proje bütçesi ve insan kaynaklarının görev bekleme süresini azaltmak amacıyla kaynak kısıtlı yazılım proje çizelgeleme probleminin çözümü için hibrit bir yaklaşım önerilmiştir. Standart GA, BO ve YAKA algoritmalarının belirli özellikleri kullanılarak iyileştirilmiştir. Böylelikle popülasyon çeşitliliği arttırılmış, standart GA ve BO algoritmalarının yerel minimuma takılma ve durgunluk sorunları iyileştirilmiş oldu. Daha başarılı arama işlemi için, kaotik rastgele sayı üreteci tercih edilmiştir. Geliştirilen yöntem, GA ve literatürdeki GreedyDo, HAntCO, GRASP, CSM ve DEM yöntemleri ile karşılaştırılmıştır. Elde edilen sonuçlar TÖİT ve Mann-Whitney testi ile istatistiksel olarak analiz edilmiştir.

- Test sonuçlarına göre, tekli insan kaynağı atama yönteminde %7, çoklu insan kaynağı atama yönteminde ise %15 oranına varan iyileşme gözlenmiştir.
- Hibrit yöntemin optimum sonuçlara ulaşma ve tutarlı sonuç üretmede başarılı olduğu görülmüştür.
- Çoklu insan kaynağı modeli entegre edildiğinde, hibrit yöntemin daha da etkin sonuçlar ürettiği belirlenmiştir.
- Çoklu insan kaynağı modeline oranla, tekli insan kaynağı atama modeli ile elde edilen sonuçların standart sapması daha küçük olmuştur.
- GA'daki yerel minimuma takılma sorunu ve BO'daki durgunluk probleminin hibrit yöntem ile önemli düzeyde azaldığı görülmüştür.

Proje süresini daha da iyileştiren ve gerçek hayat projeleri için daha kullanılabilir olan çoklu insan atama yönteminde durum uzayının büyük olması nedeniyle, çözüm süresi uzamakta ve yerel minimuma takılma sorunundaki iyileşme minimum seviyede kalmaktadır. Gelecek dönem çalışmalarında, bu alanda iyileştirmelerin yapılması ve bütçede, insan kaynaklarında ve görevlerin tamamlanma sürelerinde oluşabilecek güncellemeler sonrasında proje çizelgesinin dinamik olarak güncellenmesi üzerine çalışmalar gerçekleştirilmesi planlanmaktadır.

ETİK STANDARTLARIN BEYANI (DECLARATION OF ETHICAL STANDARDS)

Bu makalenin yazar(lar)ı çalışmalarında kullandıkları materyal ve yöntemlerin etik kurul izni ve/veya yasal-özel bir izin gerektirmediğini beyan ederler.

YAZARLARIN KATKILARI (AUTHORS' CONTRIBUTIONS)

Nurhan GÜL: Yöntemi tasarlamış, yazılımı hazırlamış, deneyleri gerçekleştirmiş, verileri analiz etmiş ve makaleyi yazmıştır.

Nursal ARICI: Çalışmayı denetlemiş, proje yönetimini gerçekleştirmiş ve makaleyi gözden geçirmiştir.

Her iki yazar da çalışmanın tamamlanması, kavramsallaştırma, metodoloji ve makale yazımı konusunda birlikte çalışmıştır.

ÇIKAR ÇATIŞMASI (CONFLICT OF INTEREST)

Bu çalışmada herhangi bir çıkar çatışması yoktur.

KAYNAKLAR (REFERENCES)

[1] J. P. Lewis, "Project Planning, Scheduling, and Control: A Hands-On Guide to Bringing Projects in on Time and on Budget", 4th edition. ***McGraw-Hill***, (2004).

[2] Project Management Institute, "A guide to the project management body of knowledge (PMBOK guide)", Fifth edition. Newtown Square, Pennsylvania: ***Project Management Institute***, Inc, (2013).

[3] A. Başar, "A novel scheduling methodology for resource constrained projects by a new mathematical model and a hybrid metaheuristic: A case study", ***Journal of the Faculty of Engineering and Architecture of Gazi University***, 37(3), (2022).

[4] A. Reza, V. Zeighami, and K. Ziarati, "Artificial Bee colony for resource constrained project scheduling problem", ***International Journal of Industrial Engineering Computations***, 2, (2011).

[5] V. Nayak, H. A. Suthar, and J. Gadit, "Implementation of Artificial Bee Colony Algorithm", ***IJ-AI***, 1(3),(2012).

[6] N. F. B. M. Pauzi and salleh ahmad Bareduan, "Scheduling analysis for flowship using artifical bee colony (ABC) algorithm with varying onlooker approaches", ***ARPN Journal of Engineering and Applied Sciences***, 11: 6472–6477, (2016).

[7] H. M. H. Saad, R. K. Chakrabortty, S. Elsayed, and M. J. Ryan, "Quantum-Inspired Genetic Algorithm for Resource-Constrained Project-Scheduling", ***IEEE Access***, 9: 38488–38502, (2021).

[8] H. F. Rahman, R. K. Chakrabortty, and M. J. Ryan, "Memetic algorithm for solving resource constrained project scheduling problems", ***Automation in Construction***, 111:103052, (2020).

[9] X. Shen, Y. Guo, and A. Li, "Cooperative coevolution with an improved resource allocation for large-scale multi-objective software project scheduling", ***Applied Soft Computing***, 88: 106059, (2020).

[10] M. Laszczyk and P. B. Myszkowski, "Improved selection in evolutionary multi–objective optimization of multi–skill resource–constrained project scheduling problem", ***Information Sciences***, 481: pp. 412–431, (2019).

[11] M. W. Guo, J. S. Wang, L. F. Zhu, S. S. Guo, and W. Xie, "An Improved Grey Wolf Optimizer Based on Tracking and Seeking Modes to Solve Function Optimization Problems", ***IEEE Access***, 8: 69861–69893, (2020).

[12] W. Xiao, H. Deng, Y. Sheng, and L. Hu, "Factored grey wolf optimizer with application to resource-constrained project scheduling", ***International Journal of Innovative Computing, Information and Control***, 14: 881–897, (2018).

[13] T. Jiang and C. Zhang, "Application of Grey Wolf Optimization for Solving Combinatorial Problems: Job Shop and Flexible Job Shop Scheduling Cases", ***IEEE Access***, 6: 26231–26240, (2018).

[14] H. D. Quoc, L. Nguyen The, C. N. Doan, and T. Phan Thanh, "Solving Resource Constrained Project Scheduling Problem by a Discrete Version of Cuckoo Search Algorithm", ***2019 6th NAFOSTED Conference on Information and Computer Science (NICS)***, Hanoi, Vietnam, 2019: 73–76, (2019).

[15] K. Bibiks, Y.-F. Hu, J.-P. Li, P. Pillai, and A. Smith, "Improved discrete cuckoo search for the resource-constrained project scheduling problem", ***Applied Soft Computing***, 69: 493–503, (2018).

[16] B. Afshar-Nadjafi, "A solution procedure for preemptive multi-mode project scheduling problem with mode changeability to resumption", ***Applied Computing and Informatics***, 14(2), (2018).

[17] M. Ali, S. Ullah, and M. Jahanzaib, "A Resource Optimisation Based Heuristic For Resource Constrained Project Scheduling Problem", ***NEDJR***, XVI(4),(2019).

[18] M. Rauf, Z. Guan, L. Yue, Z. Guo, J. Mumtaz, and S. Ullah, "Integrated Planning and Scheduling of Multiple Manufacturing Projects Under Resource Constraints Using Raccoon Family Optimization Algorithm", ***IEEE Access***, 8: 151279–151295, (2020).

[19] M. Á. Vega-Velázquez, A. García-Nájera, and H. Cervantes, "A survey on the Software Project Scheduling Problem", ***International Journal of Production Economics***, 202: 145–161, (2018).

[20] J. Snauwaert and M. Vanhoucke, "A new algorithm for resource-constrained project scheduling with breadth and depth of skills", ***European Journal of Operational Research***, 292(1), (2021).

[21] A. Ghamginzadeh, A. A. Najafi, and M. Khalilzadeh, "Multi-Objective Multi-Skill Resource-Constrained Project Scheduling Problem Under Time Uncertainty", ***Int. J. Fuzzy Syst***., 23(2), (2021).

[22] H. Dai and W. Cheng, "A Memetic Algorithm for Multiskill Resource-Constrained Project Scheduling Problem under Linear Deterioration", ***Mathematical Problems in Engineering***, 2019: 1–16, (2019).

[23] Y. Guo, J. Ji, J. Ji, D. Gong, J. Cheng, and X. Shen, "Firework-based software project scheduling method considering the learning and forgetting effect", ***Soft Computing***, 23(13): 5019–5034, (2019).

[24] J. Blazewicz, J. K. Lenstra, and A. H. G. R. Kan, "Scheduling subject to resource constraints: classification and complexity", ***Discrete Applied Mathematics***, 5(1), (1983).

[25] F. Uysal, "Hybrid meta-heuristic algorithms for the resource constrained multi-project scheduling problem", Doctoral dissertation, ***Middle East Technical University***, Ankara, (2014).

[26] **http://imopse.ii.pwr.wroc.pl**, Wrocław University of Science and Technology, "Intelligent Multi Objective Project Scheduling Environment", (2021).

[27] P. B. Myszkowski, M. Laszczyk, I. Nikulin, and M. Skowroński, "iMOPSE: a library for bicriteria optimization in Multi-Skill Resource-Constrained Project Scheduling Problem", ***Soft Computing***, 23(10), (2019).

[28] H. D. Quoc, L. N. The, C. N. Doan, and T. P. Thanh, "New Effective Differential Evolution Algorithm for the Project Scheduling Problem", ***2nd International Conference on Computer Communication and the Internet (ICCCI)***, Nagoya, Japan, 2020: 150–157, (2020).

[29] P. B. Myszkowski, Ł. P. Olech, M. Laszczyk, and M. E. Skowroński, "Hybrid Differential Evolution and Greedy Algorithm (DEGR) for solving Multi-Skill Resource-Constrained Project Scheduling Problem", ***Applied Soft Computing***, 62: 1–14, (2018).

[30] P. B. Myszkowski, M. Skowroński, and K. Sikora, "A new benchmark dataset for Multi-Skill Resource-Constrained Project Scheduling Problem", ***Proceedings of the Federated Conference on Computer Science and Information Systems***, 5: 129–138, (2015).

[31] A. M. Fahmy, "Optimization Algorithms in Project Scheduling", ***Optimization Algorithms - Methods and Applications***, O. Baskan, Ed. InTech, (2016).

[32] F. Meziane and S. Vadera, Eds., Artificial Intelligence Applications for Improved Software Engineering Development: New Prospects. ***IGI Global***, (2010).

[33] N. Yavuz, "Relible data transfer in chaotic systems", M.S. Thesis, ***Karadeniz Technical University***, Trabzon, Turkey, (2006).

[34] A. R. Ozoren, "Random Number Generation Using Chaotic Dynamical Maps", M.S. Thesis, ***Boğaziçi University***, Istanbul, Turkey, (2011).

[35] S. Wu, H.-D. Wan, S. K. Shukla, and B. Li, "Chaos-based improved immune algorithm (CBIIA) for resource-constrained project scheduling problems", ***Expert Systems with Applications***, 38(4), (2011).

[36] D. Tian, "Particle Swarm Optimization with Chaotic Maps and Gaussian Mutation for Function Optimization", ***IJGDC***, 8(4), (2015).

[37] G. Yan and C. Li, "An Effective Refinement Artificial Bee Colony Optimization Algorithm Based On Chaotic Search and Application for PID Control Tuning", ***Journal of Computational Information Systems***,7(9) , (2010).

[38] F. Gargiulo and D. Quagliarella, "Genetic Algorithms for the Resource Constrained Project Scheduling Problem", ***2012 IEEE 13th International Symposium on Computational Intelligence and Informatics (CINTI)***, Budapest, Hungary, 2012: 39–47, (2012).

[39] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer", ***Advances in Engineering Software***, 69: 46–61, (2014).

[40] S. Mirjalili, S. Saremi, S. M. Mirjalili, and L. dos S. Coelho, "Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization", ***Expert Systems with Applications***, 47: 106–119, (2016).

[41] Q. Tu, X. Chen, and X. Liu, "Hierarchy Strengthened Grey Wolf Optimizer for Numerical Optimization and Feature Selection", ***IEEE Access***, 7: 78012–78028, (2019).

[42] X. Zhou, F. Miao, and H. Ma, "Genetic Algorithm with an Improved Initial Population Technique for Automatic Clustering of Low-Dimensional Data", ***Information***, 9(4): 101, (2018).