Fırat Üniversitesi Müh. Bil. Dergisi
34(2), 779-794, 2022

Araştırma Makalesi
https://doi.org/10.35234/fumbd.1105174

Tüm Arama Uzayı Taranarak Kaynak Dengeleme Probleminin Optimum Çözülmesi

Önder Halis BETTEMİR[1*], Tuğba ERZURUM[2]

[1] İnşaat, Mühendislik Fakültesi, İnönü Üniversitesi, Malatya, Türkiye

[*1] onder.bettemir@inonu.edu.tr, [2] tugbaerzrm@gmail.com

(Geliş/Received: 20/04/2022; Kabul/Accepted: 03/09/2022)

Öz: İnşaat süresi boyunca kaynak kullanımında gözlemlenen dalgalanmaların proje süresinde gecikme olmadan asgari düzeye indirilmesi kaynak dengeleme problemi olarak tanımlanır. Bu problem basit kurallar içeren sezgisel, sistematik fakat rassal biçimde arama uzayını tarayan üst-sezgisel ve analitik yöntemlerle çözülmektedir. Ancak aktivite sayısının artması ile arama uzayı çok büyüyen kaynak dengeleme probleminin çözümünde oluşan sorunlara karşı belirtilen yöntemler yetersiz kalmaktadır. Bu çalışmada, aktiviteler arasındaki kısıtlar ihlal edilmeden kritik olmayan aktivitelerin ertelenmesi ile kaç farklı şekilde uygulanabilir çözüm elde edilebileceği hesaplanarak kaynak dengeleme probleminin arama uzayının büyüklüğü belirlenmiştir. Belirlenen arama uzayının tamamı taranarak uygulanabilir en iyi çözüm garantili biçimde elde edilmiştir. Kaynak dağılımının uygunluğu minimum moment ölçeği ile incelenmiştir. Yöntemin uygulanabilmesi için bir hesap tablosu uygulaması oluşturularak Visual Basic programlama dilinde makro yazılmıştır. Literatürden derlenen 27 kaynak dengeleme probleminin geliştirilen yöntemle optimum çözümü elde edilmiştir. Hesap cetveline kaynak dengeleme probleminin nasıl tanıtılacağı çözülen problemler arasından seçilen 2 örnek problem üzerinde detaylı biçimde anlatılarak yöntemin tüm araştırmacılar tarafından uygulanabilmesi sağlanmıştır. Ayrıca geliştirilen yöntem çözüm süresini kısaltmak için C++ dilinde de kodlanmıştır. Test problemleri genetik algoritmayla çözülmüş, hesaplama süreleri ve sunduğu sonuçlar önerilen yöntemin çıktıları ile karşılaştırılmıştır. Çözüm süresi ve çözüm iyiliğinin karşılaştırılması sonucunda geliştirilen yöntemin kaynak dengeleme probleminin kesin çözümünde uygulanabilir olduğu belirlenmiştir. Ayrıca çalışma kaynak dengeleme probleminin arama uzayını belirleyen bir yöntem geliştirerek literatüre katkı sağlamaktadır.

Anahtar kelimeler: Kaynak dengeleme problemi, ok diyagramı, optimizasyon.

Optimum Solution of Resource Leveling Problem by Evaluating the Whole Search Domain

Abstract: The endeavor to minimize the fluctuations of the resource assignments without postponing the project deadline is named as resource leveling problem. This problem is solved by heuristic methods, consist of simple rules, meta-heuristic methods, explore the search domain systematically but randomly, and analytical methods. Nevertheless, aforementioned approaches come to be insufficient when the number of activity of the project increases which oversize the search domain of the resource leveling problem. In this study, the number of the feasible solutions is determined by detecting the number of feasible schedules obtained by delaying the noncritical activities without violating the restrictions on the activities. The entire search domain is evaluated and the optimum feasible solution is obtained in a guaranteed manner. Suitability of the resource distribution is evaluated by minimum moment metric. A spreadsheet application is developed and macro is written by Visual Basic programming language to implement the developed method. In the literature survey, twenty seven resource leveling problems are collected and optimum solutions of them are obtained by the developed method. Definition of the parameters and the evaluation of the search domain of the resource leveling problem by the spreadsheet application are explained in detail on two of the solved case study problems so that the researchers can implement the developed method. Moreover the developed method is coded on C++ in order to expedite the computations. Test problems are also solved by genetic algorithm and the computation duration and the obtained results are compared with the proposed method. The comparison reveals that the developed method can be implemented for optimizing the resource distribution. Moreover, this research contributes to the literature by providing a methodology for the determination of the search domain of the resource leveling problem.

Key words: Resource leveling problem, activity-on-arrow, optimization.

* Sorumlu yazar: onder.bettemir@inonu.edu.tr. Yazarların ORCID Numarası: [1] 0000-0002-5692-7708, [2] 0000-0003-4788-6999

1. Giriş

Bir projenin yürütülmesi için temin edilmesi gereken kaynak miktarı proje boyunca düzgün dağılım göstermeyebilir. Proje süresince kaynak kullanımındaki artış ve azalışlar boşta kalan işçi ve iş makinesinin artmasına yol açarak önemli miktarda maddi yük getirir. Kaynak dengeleme, proje süresini geciktirmeden proje boyunca kaynak kullanımını en uygun şekle getirmeyi amaçlar. Kaynak dengelemesi yapıp etkin bir kaynak kullanımı oluşturarak kaynakların atıl kalma süreleri azaltıp günlük kaynak kullanım miktarındaki aşırı değişimin önüne geçilebilir. Bu sayede çalışanların kısa dönemler için ücretsiz izne ayrılması veya işten çıkarılıp bir süre sonra tekrar işe alınması ve benzeri çalışma süreksizliğine yol açan uygulamalardan dolayı çalışanlarda görülebilecek öğrenme, verim ve iş kalitesi kaybı ile maliyet azaltılabilir.

Kaynak Dengeleme Problemi (KDP) proje yönetiminin önemli çalışma alanlarından biridir. KDP'nin çözülmesi için kesin yöntemler, sezgisel yöntemler ve üst-sezgisel yöntemler uygulanmaktadır. Kesin yöntemler NP-Zor sınıfında olan KDP'nin küçük boyutlu problemlerinde garantili biçimde optimum çözümü verebilmektedir [1]. Aktivite sayısının artması KDP'nin karmaşık olması nedeniyle hesaplama süresi ve bellek ihtiyacını çok hızlı biçimde arttırmakta ve bunun sonucunda çözüm süresi aşırı uzamakta veya bellek yetersiz kalmaktadır [2]. Büyük boyutlu problemlerde matematiksel yöntemlerin yetersiz kalması sonucu araştırmacılar kesin yöntemlerin yanı sıra sezgisel ve üst-sezgisel yöntemlere yoğunlaşma eğilimindedir. Çünkü üst-sezgisel yöntemler, büyük problemlerin yakın optimum çözümünü makul bir çözüm süresinde bulma yeteneğine sahiptir.

Sezgisel yöntemler belirli kurallara göre geliştirilen ve düşük hesap yükü ile yakın optimum sonucun elde edilmesini sağlayan yöntemlerdir. İlk önce kaynak talebi en fazla veya en az olan aktiviteyi erteleme, bolluk süresi en fazla veya en az olan aktiviteyi erteleme ve benzeri kurallarla en iyi çözüm aranır. Kaynak kullanımındaki günlük dalgalanmaları en aza indirmek için minimum moment ölçeğini geliştirilmiştir [3]. Bu yöntem 'pack metodu' olarak adlandırılan yöntem geliştirilerek iyileştirilmiştir [4]. KDP'nin çözümünde aktivitelerin ertelenmesi sırasında tüm şebekenin çözülmesi yerine sadece başlangıç ve tamamlanma süreleri değişen aktivitelerin çözülerek sonucun elde edilmesini sağlayan bir yöntem geliştirilmiştir [5, 6]. Bu sayede hesap yükü önemli ölçüde azalmıştır.

Sezgisel yöntemlerin en önemli avantajı hesap yükünün düşük olmasıdır, fakat bunun karşılığında optimum sonucu ender biçimde elde edebilmektedirler. Bu nedenle KDP'nin çözümünde üst-sezgisel yöntemler oldukça yaygın biçimde uygulanmaktadır ve en sık kullanılan üst-sezgisel yöntem Genetik Algoritma (GA)'dır [7-12]. Ayrıca GA KDP ye çok yakın bir problem olan insansız araçların görev paylaşımı için de uygulanmıştır [13]. Ayrıca literatürde karınca koloni algoritması, parçacık sürü optimizasyonu, tavlama benzetimi ve birleştirilmiş farklı üst sezgisel yöntemlere dayalı çalışmalar bulunmaktadır [14-19]. Uygulanan üst sezgisel algoritmaların dışında Kaotik Altın Sinüs, Kaya Kartalı Optimizasyonu, Kuantum Uyarlamalı Genetik Algoritmalar ve Gauss Kaotik Haritalı Genetik Algoritma yöntemleri de akademik çalışmalarda uygulanmaktadır [20-23]. Üst-sezgisel yöntemler, sezgisel yöntemlere göre daha iyi sonuçlar vermektedir. Fakat, aktivite sayısının artması ile üst-sezgisel yöntemlerin optimum sonucu elde etme olasılığı düşmekte ve yakınsanan çözüm optimum çözümden uzaklaşmaktadır. Bu nedenle kesin çözüm sunan yöntemler de kullanılmaktadır. Bu doğrultuda dal-sınır metodunu uygulanarak KDP çözülmüştür [24-26]. Tamsayılı-doğrusal programlama da KDP'nin çözümünde küçük ölçekli projelerin çözümünde uygulanmıştır [27,28]. Karışık tamsayılı programlama kullanarak KDP küçük ve orta ölçekli KDP çözümünde uygulanmıştır [1, 29-31]. Kesin yöntemler teorik olarak KDP'nin optimum çözümünü elde edebilir, fakat aktivite sayısının artması ile matematiksel programlamaya dayalı yöntemlerde değişken sayısı çok hızlı artmakta ve çözüm süresi uzamaktadır. Dal-sınır algoritmasında ise aktivite sayısı ve şebekenin karmaşıklığına göre dalların çok fazla alt-dalları oluşmakta ve çözüm süresi çok uzamaktadır. Ayrıca dal sayısının aşırı artması sonucu görülen bellek ve hesap süresi sorunları nedeniyle dalların elenmesi sonucu garantili biçimde optimum çözüm elde edilememektedir. Bu nedenle literatürde kesin yöntemlerle optimum çözümü elde edilen en büyük KDP 50 aktivitelidir [1]. Üst-sezgisel yöntemlerde ise elde edilen sonucun iyileştirilmesi için popülasyonun büyültülmesi ve deneme sayısının arttırılması gereklidir [32]. Üst-sezgisel yöntemlerin çözüm kapasiteleri akademik çalışmalarla sürekli olarak iyileştirilse de elde edilen sonucun kesin en iyi sonuç olduğu garanti edilememektedir.

Üst-sezgisel yöntemlerin belirtilen çözüm yeteneklerine rağmen mevcut durumda kullanılan KDP çözüm yöntemlerinin kısıtlarından dolayı yeni bir çözüm yöntemine ihtiyaç duyulmaktadır. Bu çalışmada KDP'nin kritik olmayan aktivitelerini geciktirerek kaç adet farklı uygulanabilir şebeke elde edilebileceği araştırılarak KDP'nin arama uzayını belirleyen ve tüm arama uzayını tarayarak garantili biçimde optimum sonucu elde eden bir yöntem geliştirilmiştir. Makalenin ikinci kısmında KDP'nin tüm arama uzayının nasıl belirlendiği ve tarandığı açıklanmaktadır. Üçüncü bölümde geliştirilen algoritmanın performansını ölçmek için literatürden

derlenen 27 KDP çözülmüş ve bu problemlerden 2'sinin hesap tablosuna nasıl tanıtılacağı detaylı biçimde açıklanmıştır. Sonuç bölümünde geliştirilen yöntemin performansı ve sınırları tartışılmıştır.

2. Yöntem

Kaynak dengeleme problemini tüm arama uzayını belirleyip tarayarak çözen yöntem ve yöntemin uygulanması bu bölümde açıklanmaktadır. Geliştirilen yöntem kaynak dengeleme probleminin şebeke analizinin gerçekleştirilmesi, kaynak dalgalanmalarının ölçülmesi ve tarama algoritmasının uygulanması işlem adımlarına bölünüp anlatılmaktadır.

2.1. KDP arama uzayının belirlenmesi için şebeke analizi

KDP proje süresi boyunca gerçekleşen kaynak kullanımı değerlerindeki dalgalanmayı proje süresini uzatmadan azaltmayı amaçladığı için sadece bolluk süresi sıfırdan büyük olan aktiviteler geciktirilir. Kritik aktivitelerin başlangıç zamanları değiştirilmediği için arama uzayının büyüklüğü kritik olmayan aktivite sayısı ve kritik olmayan aktivitelerin bolluk sürelerine bağlıdır. Şebeke diyagramında aktiviteler birbirlerine seri veya paralel biçimde bağlanabilirler. Seri bağlı aktivitelerin bulunduğu bir hat üzerinde problemin çözümünün aranması halinde aktivitelere tanımlanan gecikme süreleri aktiviteler arasındaki ilişkileri ihlal etmemelidir. Bu nedenle seri bağlı bir hat üzerindeki aktivitelere erteleme süresi tanımlanırken öncel aktivitelerin süreleri mutlaka ardıl aktivite süresine küçük veya eşit olmalıdır. Seri bağlı bir hat üzerinde bulunan n gün toplam bolluk süresine sahip m aktivitenin kaç farklı şekilde geciktirilebileceği Eşitlik 1'de gösterildiği gibi hesaplanır [33].

$$UGS = \frac{1}{m!}\prod_{i=1}^{m}(n+i) \tag{1}$$

Eşitlik 1'de UGS aktivitelerin birbirine seri bağlı olduğu hat için uygulanabilir gecikme sayısı'nı ifade etmektedir. Şebeke üzerinde birbirine paralel bağlı aktiviteler de bulunabilir. Paralel aktiviteler arasında ardıl-öncel ilişkisi bulunmadığı için bu aktivitelere tanımlanan gecikme süreleri şebekedeki mantıksal ilişkileri etkilememektedir. Bu nedenle paralel aktivitelere tanımlanan gecikme süreleri birbirinden bağımsız biçimde tanımlanmaktadır. Birbirine paralel aktivite veya yolların kaç farklı şekilde geciktirilebileceği her bir paralel aktivite veya yola ait UGS'nin birbiri ile çarpılmasıyla hesaplanmaktadır.

Şebekeyi oluşturan aktiviteler ve bu aktivitelerin oluşturduğu hatlar birbirlerine her zaman seri veya paralel olarak tanımlanamayabilir. Şebekenin karmaşıklık düzeyi arttıkça şebekeyi oluşturan hat sayısı ile birleşen veya ayrılan hat sayısı çok hızlı biçimde artmaktadır. Bu durumda şebekenin başlangıç düğümünden bitiş düğümüne kadar uzanan hatları birbirine paralel veya seri hat olarak tanımlamak mümkün olmamaktadır. Seri ve paralel hatlar oluşturabilmek için şebeke üzerindeki hatlar hiç ayrılan veya birleşen başka bir hat olmayacak şekilde mümkün olan en uzun hatlara bölünerek seri ve paralel hatlar oluşturulmakta ve arama uzayının büyüklüğü hesaplanmaktadır.

2.2. Kaynak dalgalanmalarının ölçülmesi

Kritik olmayan aktiviteleri toplam bolluk sürelerini aşmayacak şekilde geciktirerek kaynak histogramında görülen zirve ve çukurlar arasındaki seviye farkının olabildiğince azaltılması amaçlanmaktadır. Aktiviteler arasındaki kısıtları ihlal etmeyen ve projenin tamamlanma süresini aşmayan iş programları arasından en az kaynak dalgalanmasına sahip iş programı aranır.

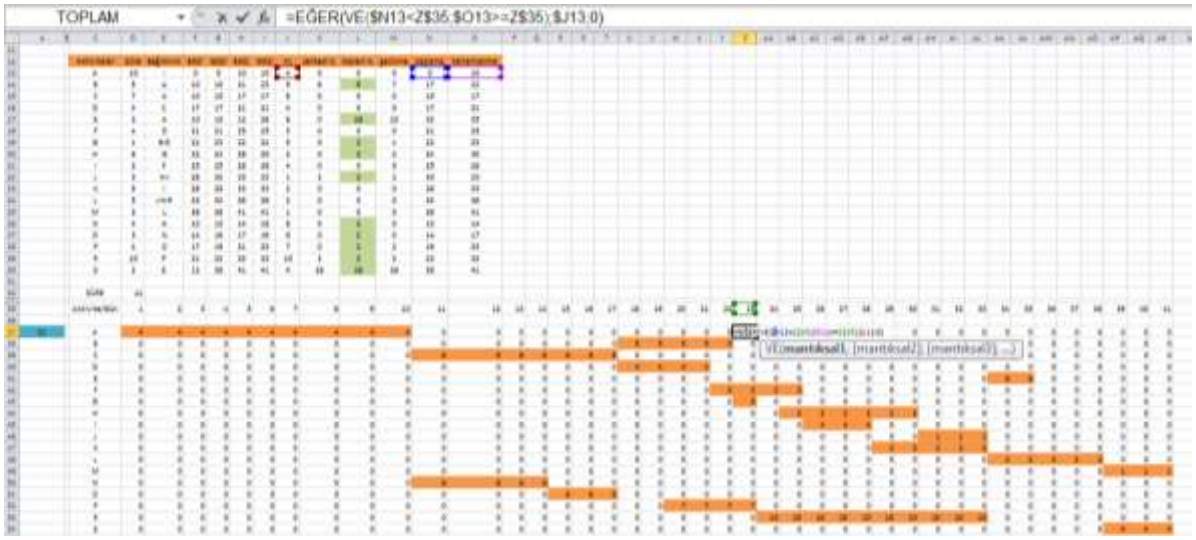
Kaynak dağılımının ölçülebilmesi için literatürde birçok ölçek bulunmaktadır. Bu ölçeklerin karşılaştırması yapılmış ve minimum moment (MM) ölçeğinin diğer ölçeklere göre daha düşük maksimum kaynak talebi ve daha az dalgalı bir kaynak histogramı sunduğu için bu çalışmada MM ölçeği kullanılmıştır [34, 35]. MM ölçeği Eşitlik 2'de sunulmuştur. Formülde S proje süresini, k farklı kaynak sayısını, ri,j i'ninci günde ihtiyaç duyulan j'ninci kaynak miktarını, ωj ise j'ninci kaynak için tayin edilen ağırlık değerini ifade etmektedir.

$$MM = \sum_{i=1}^{S}\sum_{j=1}^{k} w_j * r_{i,j}^2 \tag{2}$$

Günlük kaynak kullanımı dikkate alınarak Eşitlik 2 ile kaynak dağılımının uygunluğu belirlenir. Eşitlik 2 günlük kaynak kullanımının karelerini topladığı için yüksek kaynak kullanımına sahip günlerin çok yüksek değer eklemesine neden olur. Kaynak dağılımının düzgün dağıldığı bir proje ise küçük sayıların karelerinin toplamından oluşacağı için daha küçük MM değerini verir. Belirtilen önerme en basit biçimde örneklenecek olursa 4 günlük bir projede birinci alternatifte inşaat dört gün içinde her günde 2x kişi çalıştırılarak tamamlandığı varsayılsın. Bu durumda MM, $4x^2 + 4x^2 + 4x^2 + 4x^2 = 16x^2$ değerini alır. İkinci alternatifte ise inşaat ilk 2 günde 3x, son iki günde x sayıda işçi çalıştırılarak tamamlansın. Bu durumda MM, $9x^2 + 9x^2 + x^2 + x^2 = 20x^2$ değerini alır. Her 2 durumda da aynı miktarda yevmiye ile inşaat gerçekleştirilmiş fakat farklı MM değerleri oluşmuştur. En uygun kaynak dağılımı Eşitlik 2'nin en düşük değeri aldığı durumdur, çünkü inşaat hızının değişkenliği, inşaat sırasında işten ayrılan, işe alınan, boşta kalan personel sayısı daha azdır. Ayrıca daha küçük şantiye kurulumuna ihtiyaç duyulacağı için genel giderler daha az olacaktır.

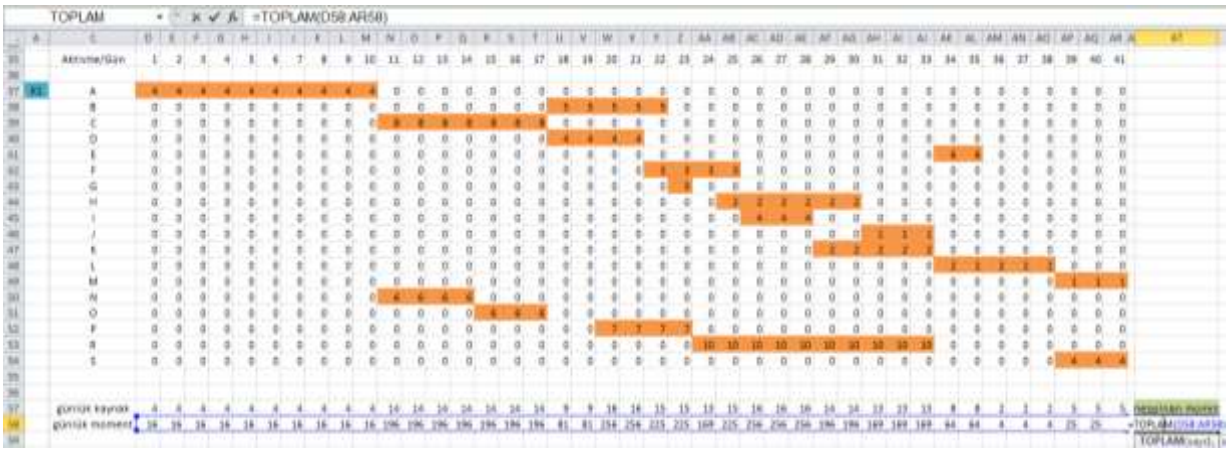
2.3. Arama uzayını tarayan algoritmanın uygulanması

Arama uzayının tamamının taranabilmesi için şebekenin bağımlılık ilişkileri göz önüne alınarak kritik yol yöntemi ile tüm aktivitelerin başlangıç ve tamamlanma zamanlarının belirlenmesi gereklidir. Sistematik biçimde arama uzayını tarayıp KDP'yi çözen bir Hesap Tablosu Uygulaması (HTU) geliştirilmiştir. HTU, projelerde kaynak kullanımını optimize etmek için tüm çözüm uzayını tarayarak en uygun iş programını elde etmektedir. Bu uygulama ofis yazılımında VBA (Visual Basic for Applications) programlama dili ile makro yazılarak hazırlanmıştır. Oluşturulan makro sayesinde kritik olmayan aktivitelerin tüm geciktirme seçeneklerini aktivitelere tanımlayıp tüm durumlar için şebekeyi çözmektedir. Projeyi oluşturan aktiviteler arasındaki kısıtları ihlal etmeden oluşturulabilecek iş programlarının tamamının analiz edilip elde edilen sonuçların değerlendirilmesi ile en iyi çözüm elde edilebilecek ve KDP'nin optimum çözümünü sunan erteleme süreleri belirlenebilecektir. KDP'nin çözüleceği şebeke hesap tablosuna Şekil 1'de gösterildiği gibi tanıtılır.



Şekil 1. Ok diyagramı ile gösterilen şebekenin hesap tablosuna girilmesi.

İleri ve geri gidiş ile belirlenen erken ve geç olay zamanları ilgili sütunlara girilir ve tüm aktivitelerin toplam ve serbest bollukları hesaplanır. Gecikme sütununa ise kritik olmayan aktivitelerin ne kadar geciktirileceği tanımlanır. Aktivitenin erken olay zamanına gecikme süresi eklenerek aktivitenin başlangıç zamanı belirlenir. Başlangıç zamanına aktivite süresi eklenerek aktivitenin tamamlanma zamanı hesaplanır. Bu işlemle birlikte KDP çözümü için gerekli tüm veriler elde edilmiş olur. Gün içinde yürütülen tüm aktivitelerin kullandığı kaynak miktarının toplamı günlük kaynak kullanımını vermektedir. Şekil 2'de gösterildiği gibi günlük kaynak kullanımı 57. satırda, kareleri ise 58. satırda hesaplanır.



Şekil 2. Minimum moment amaç fonksiyonunun hesaplanması

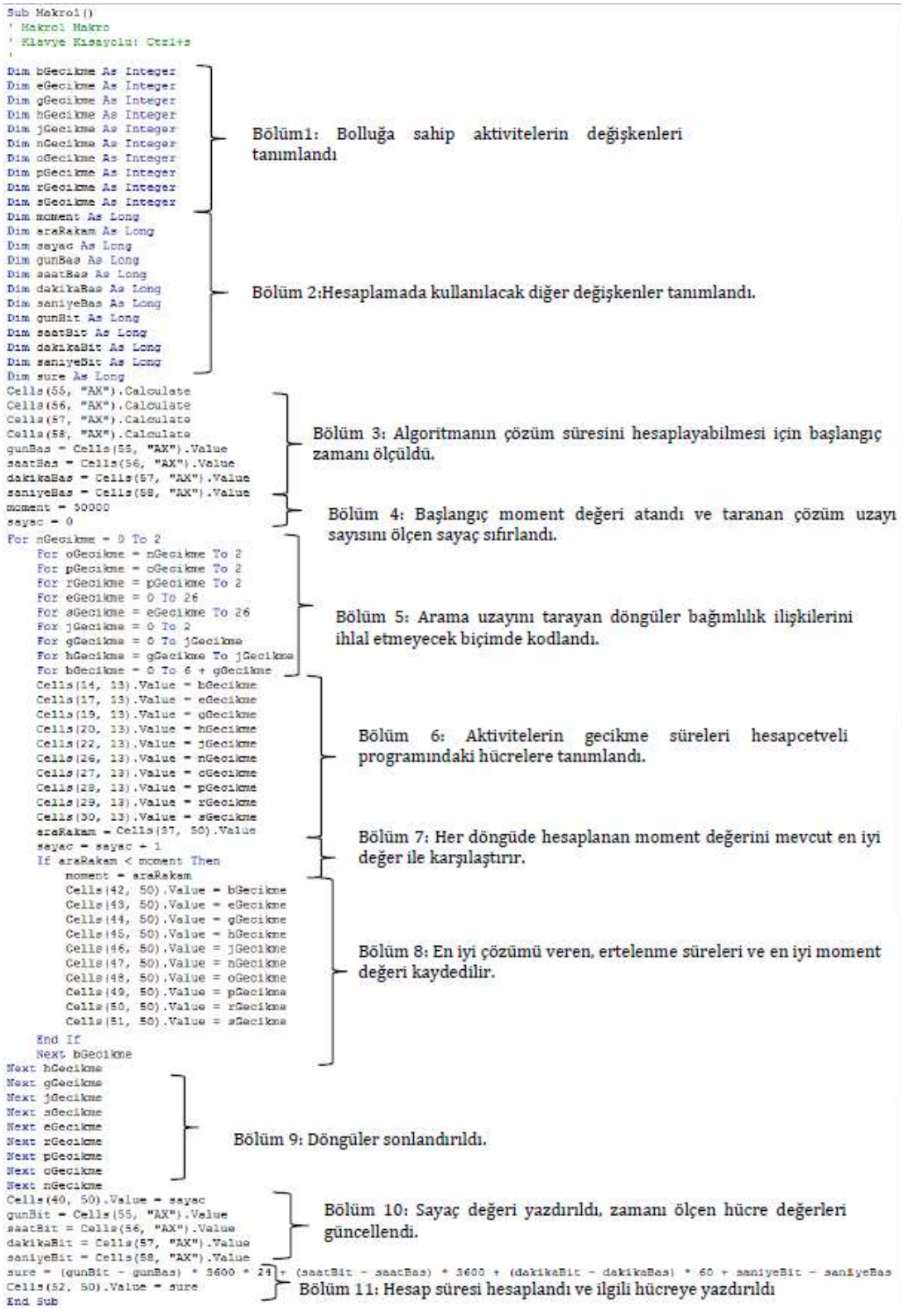
Kareler toplamı *AT58* numaralı hücrede gösterilen formülle toplanır ve Eşitlik 2'de gösterilen amaç fonksiyonu hesaplanır. Kaynak çeşidinin 1'den fazla olması durumunda her kaynak için kareler toplamı belirlenir ve kendi ağırlığı ile çarpılıp toplanarak amaç fonksiyonu hesaplanır. Bu aşamada şebeke çözülüp günlük kaynak kullanımı belirlenmiş ve MM ölçeğine göre kaynak dağılımı değerlendirilmiştir. Şebekenin tüm uygulanabilir gecikme sürelerini tanımlayıp şebeke çözümünü gerçekleştirmek için yazılan makro kodu Şekil 3'te sunulmuştur.

Makronun ilk bölümünde kritik olmayan aktivitelere atanacak gecikme sürelerini saklayan değişkenler tanımlanmıştır. Gecikme süresi tamsayı olabileceği için tüm değişkenler Integer biçimindedir. Makronun ikinci bölümünde hesaplamaya başlanılan zamanı ve hesaplamanın tamamlandığı zamanı ölçmek için saniye, dakika, saat ve gün değerlerini saklayan değişkenler ile en iyi çözüm değerlerini saklamak için gerekli değişkenler tanımlanmıştır. Üçüncü bölümde zaman bilgisini saklayan hücrelerin değerleri Calculate komutu ile güncellenmiş ve bu değerler ilgili değişkenlerde saklanmıştır.

Dördüncü bölümde amaç fonksiyonunun mevcut en iyi çözümünü saklayan değişken ile kaç çözüm yapıldığını saklayan sayaçlar sıfırlanmaktadır. Beşinci bölümde bolluğu olan aktivitelere gecikme değeri atayan döngüler çalıştırılmıştır. Birbirine seri bağlı aktivitelerin bolluk süreleri, kendinden önceki aktivitelerin bolluk sürelerinden etkilenmektedir. Öncel aktivitelerin ertelenmesi nedeniyle bir aktivite erken başlangıç zamanında başlayamaz duruma gelebilir. Bu durumda incelenen aktivite şebekenin içerdiği bağımlılık kurallarını ihlal etmeyecek şekilde ertelenmelidir. Şekil 3'te yer alan For döngüleri incelendiğinde öncel aktiviteleri olmayan veya sadece kritik aktivite olan aktivitelerde döngü sıfırdan toplam bolluk süresine kadar saydırılmaktadır. Fakat incelenen aktivitenin öncelleri arasında kritik olmayan aktivite bulunması durumunda döngü kritik olmayan aktivitenin ertelenme süresinden başlayarak toplam bolluk süresine kadar saydırılmaktadır. Bu şekilde şebekenin bağımlılık ilişkileri ihlal edilmeden, çözülmüş şebekeleri tekrar çözüp tekrara düşmeden ve hiçbir uygulanabilir erteleme alternatifini kaçırmadan tüm arama uzayı taranmaktadır.

Kaynak kodunun altıncı bölümünde For döngüleri ile elde edilen erteleme süreleri Şekil 1'de M sütununda yer alan gecikme başlığı altındaki hücrelere yazdırılmaktadır. Yapılan formülleştirme sonucunda gecikme sürelerinin atanması ile şebeke güncellenmekte ve Şekil 2'de gösterilen kaynak dağılımı elde edilmektedir. Denklem 1'de yer alan amaç fonksiyonu hesaplanarak mevcut şebekenin kaynak dağılımının uygunluğu belirlenir. Amaç fonksiyonunun değeri araRakam adlı değişkende saklanmaktadır.

Elde edilen amaç fonksiyonu mevcut en iyi çözümün saklandığı moment değişkeninin değeri ile karşılaştırılır. Hesaplanan değer mevcut en iyi çözümden daha düşükse daha iyi bir çözümün elde edildiği anlamına gelir ve mevcut çözümü veren aktivite erteleme süreleri saklanır. Hesaplanan değerin mevcut en iyi çözümle karşılaştırılması yedinci bölümde, en iyi çözümü veren erteleme sürelerinin kaydedilmesi ise sekizinci bölümde gerçekleştirilmektedir.



Şekil 3. KDP'nin tüm arama uzayını örnek problem 1 için tarayan Visual Basic makrosu

Örnek kodun dokuzuncu bölümünde For döngüleri sonlandırılmakta, onuncu bölümünde ise zamanı ölçen hücrelerden o anki zaman değeri elde edilmektedir. On birinci bölümde ise hesaplama süresi hesaplanıp ilgili hücreye yazdırılmaktadır. Bu şekilde kaynak dengeleme probleminin tüm arama uzayı taranmakta ve garantili biçimde en iyi sonuç elde edilebilmektedir.

HTU'nun bu çalışmada atıfta bulunulan tüm vaka analizlerinde uygulama adımlarının detayları Erzurum (2019)'da yer almaktadır [36]. HTU ile yapılan hesaplamalar uzun sürdüğü için Şekil 1 ve 2'de gösterilen hücrelerde yapılan günlük kaynak talebi hesaplamaları visual basic dilinde kodlanmış ve bu şekilde örnek problemler tekrar çözülmüştür. Ayrıca hesaplamaların tamamı C++'da kodlanarak hesap tablosu tamamen devre dışı bırakılarak çalıştırılabilir dosyada işlemler gerçekleştirilmiştir.

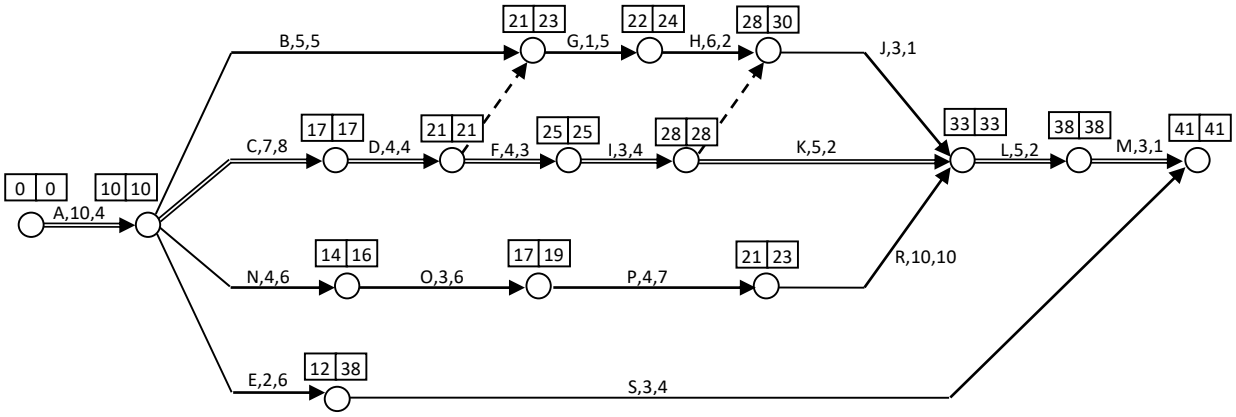
Geliştirilen yöntemin hesaplama süresinin ve elde ettiği sonuçların kıyaslanabilmesi için tüm problemler Genetik Algoritma (GA) ile de çözülmüştür. GA için çaprazlama %70, mutasyon %10 alınmıştır. Popülasyon boyutu aktivite sayısının 2 katı olarak alınmıştır. GA ile tüm projeler 2 ayrı analizde çözülmüş, birinci analizde aktivite sayısının karesinin 2 katı kadar çözüm yapılmıştır. İkinci analizde ise aktivite sayısının üçüncü kuvvetinin 2 katı kadar çözüm yapılmıştır.

3. Vaka Analizleri

Geliştirilen yöntemin denenmesi için literatürden elde edilen 27 kaynak dengeleme problemi çözülmüştür. Hesaplamalar Intel Core i5-9400F 6 Çekirdekli 2,9 GHz işlemci hızına sahip 8 GB RAM, 9 MB Cache bir bilgisayarda tek çekirdek çalıştırılarak yapılmıştır. Örnek problemler arasından seçilen 2 problemin çözümü ve arama uzayının taranması detaylı biçimde açıklanmıştır.

3.1. Örnek Problem 1

Geliştirilen algoritmanın denendiği ilk problem Bandelloni vd. tarafından oluşturulan 18 aktiviteli projedir [37]. Örnek problem 1'in şebeke diyagramı Şekil 4'te sunulmuştur. Aktiviteler şebekede oklarla gösterilmektedir. Aktivite ismini takip eden ilk rakam aktivite süresini, diğer rakam ise kaynak kullanım miktarını göstermektedir. Şebekenin çözülmesi ile elde edilen erken olay ve geç olay zamanları düğüm noktaları üzerinde gösterilmiştir. Şebekedeki kritik aktiviteler çift çizgili okla, bolluğu olan aktiviteler ise tek çizgili okla gösterilmektedir.



Şekil 4. Örnek problem 1'in şebeke diyagramı, aktivite süreleri ve kaynak kullanımı

Şebeke birbirinden bağımsız 4 hatta sahiptir. Bu hatlardan çift çizgiyle gösterilen kritik, kalan 3 hat ise bolluk sahibi aktivitelerin oluşturduğu hatlardır. Kritik olmayan aktivitelerden oluşan hatlar B-G-H-J, N-O-P-R ve E-S hatlarıdır. Örnek problem 1'in arama uzayının büyüklüğü kritik olmayan hatların kaç farklı biçimde ertelenebileceğinin hesaplanması ile belirlenmektedir. Şekil 4'te sunulan şebeke diyagramında erken ve geç olay zamanları göz önüne alınarak aktivitelerin toplam bolluk süreleri hesaplanabilir.

E-S hattı için E ve S aktivitelerinin 26 gün toplam bolluğa sahip oldukları görülmektedir. E aktivitesinin serbest bolluğu olmadığı için E aktivitesinin ertelenmesi S aktivitesini de aynı sürede erteleyecektir. E

aktivitesinin ertelendiği her gün, S aktivitesinin erteleme süresini azaltacaktır. E aktivitesi 0 gün gecikiyorsa S aktivitesine 0 ila 26 gün arasında toplam 27 farklı gecikme süresi atanabilir. Benzer şekilde E aktivitesi 1 gün gecikiyorsa S aktivitesine 1 ila 26 gün arasında toplam 26 farklı gecikme süresi tayin edilebilir. Çözüme bu şekilde devam edildiğinde E aktivitesi 25 gün geciktirildiğinde S aktivitesi 25 veya 26 gün geciktirilebilir, E aktivitesi 26 gün geciktirildiğinde S aktivitesi sadece 26 gün geciktirilebilir. Bu durum incelendiğinde uygulanabilir erteleme sayısı 26'dan 1'e kadar olan sayıların toplamına eşit olacaktır. Bu durumda erteleme sayısı 27*28/2=378 olarak hesaplanmıştır. Bir başka deyişle E ve S aktivitelerine bir birinden farklı ertelenme süreleri atayarak 378 adet birbirinden farklı uygulanabilir iş programı elde edilebilmektedir. Eşitlik 1'e m=2, n=26 değerleri girildiğinde 378 sonucu elde edilir.

Kritik olmayan aktivitelerden oluşan ikinci hat olan N-O-P-R hattında aktivite adlarının ifade edildiği sırada aktiviteler arasında seri biçimde bitince başlar ilişkisi bulunmaktadır. Hattaki ilk öncel aktivite olmasından dolayı N-O-P-R hattındaki aktivitelerin gecikme süreleri N aktivitesinden az olamaz. Benzer şekilde O aktivitesinin ardıl aktivitelerinin gecikme süreleri de O aktivitesinden az olamaz. Hattın toplam bolluk süresi 2 gündür. Belirtilen koşullar altında aktivitelere atanabilecek erteleme süreleri Tablo 1'de sunulmuştur.

Tablo 1. N-O-P-R hattının ertelenme kombinasyonu.

N Erteleme Süresi	O Erteleme Süresi	P Erteleme Süresi	R Erteleme Süresi	Toplam Erteleme Sayısı
0	0	0	0-1-2	3
0	0	1	1-2	2
0	0	2	2	1
0	1	1	1-2	2
0	1	2	2	1
0	2	2	2	1
1	1	1	1-2	2
1	1	2	2	1
1	2	2	2	1
2	2	2	2	1
				Toplam=15

Tablo1'de erteleme süreleri en soldaki aktivitelere başlangıçta 0 gün atayıp sağdaki aktivitelere atanabilecek gecikme sürelerinin hesaplanması gösterilmektedir. N, O ve P aktivitelerinin 0 gün ertelenmesi ile P aktivitesi için 0, 1 ve 2 gün erteleme olmak üzere 3 farklı erteleme ihtimali olmaktadır. N ve O aktivitelerinin ertelenme sürelerinin 0 günde sabit olduğu, P aktivitesinin ise 1 gün ertelendiği durum için 2 farklı erteleme, P aktivitesinin 2 gün ertelendiği durum için 1 farklı erteleme olmak üzere toplam 6 farklı erteleme seçeneği oluşur. Değişen N aktivite ertelenme süreleri sonucu Tablo 1'de gösterildiği üzere 6-3-1/3-1/1 farklı ertelenme süreleri oluşmuştur. Aktivite erteleme kombinasyonları değerlendirildiğinde 15 farklı iş programı elde edilmektedir. Eşitlik 1'e m=4, n=2 değerleri girildiğinde 15 değeri elde edilir.

Kritik olmayan aktivitelerden oluşan son hat ise B-G-H-J hattıdır. Bu hattın erteleme kombinasyon hesaplaması Tablo 2'de gösterilmiştir. Hatta bağlanan kukla aktiviteler nedeniyle B hattının 6 gün serbest bolluğu bulunmakta ve diğer aktivitelerin toplam bolluğu 2 günken, B aktivitesinin 8 gün toplam bolluğu bulunmaktadır.

Tablo 2. B-G-H-J hattının ertelenme kombinasyonu**.**

J.Ert Süresi	G.Ert Süresi	H Ert Süresi	B Ert Süresi	Farklı Erteleme Sayısı
0	0	0	0-1-2-3-4-5-6	7
1	0	0-1	0-1-2-3-4-5-6	2*7=14
1	1	1	0-1-2-3-4-5-6-7	1*8=8
2	0	0-1-2	0-1-2-3-4-5-6	3*7=21
2	1	1-2	0-1-2-3-4-5-6-7	2*8=16
2	2	2	0-1-2-3-4-5-6-7-8	1*9=9
				Toplam=75

Üçüncü hattın erteleme hesaplamasında önce hattın son aktivitesi olan J aktivitesinin 0 gün ertelenme durumu incelenmiştir. Bu durumda serbest bollukları 0 gün olan G ve H aktiviteleri zorunlu olarak 0 gün ertelenecek, 6 gün serbest bolluğu olan B aktivitesi ise 0 günden 6 güne kadar 7 farklı ertelenme seçeneğine sahip olacaktır. Belirtilen koşul Tablo 2'nin ilk satırında gösterilmektedir.

J aktivitesinin 1 gün ertelendiği durum için G ve H aktiviteleri hiç ertelenmeyebilir, sadece H aktivitesi 1 gün ertelenebilir veya G ve H aktiviteleri 1'er gün ertelenebilir. Bu durumda B aktivitesinin kaç gün ertelenebileceği Tablo 2'de 2 ve 3. satırlarda gösterilmiştir. J aktivitesi 2 gün, G aktivitesi 0 gün ertelenirse H aktivitesi 0, 1 ve 2 gün olmak üzere 3 farklı biçimde ertelenebilir. Sırası ile J ve G aktiviteleri 2 ve 1 gün ertelendiğinde H aktivitesi 1 veya 2 gün olmak üzere 2 farklı biçimde ertelenebilir. J ve G aktivitelerinin 2 gün ertelendiği durumda H aktivitesi sadece 2 gün ertelenebilir. B aktivitesinin ertelenmeleri ise J-G-H aktivitelerinin ertelenmeleri için Tablo 2'de gösterildiği gibi gerçekleştirilebilir. Belirtilen ertelemeler Tablo 2'nin 4, 5 ve 6. satırlarında gösterilmiştir. B aktivitesinin ertelenme süreleri J-G-H aktiviteleri ile oluşturulabilecek geciktirme senaryolarının tamamı dikkate alındığında B-G-H-J hattı toplam 75 farklı biçimde ertelenebilir.

Kritik olmayan aktivitelerden oluşan ve birbirine paralel N-O-P-R, E-S ve B-G-H-J hatları üzerindeki aktivitelere atanan gecikme süreleri diğer hatlara atanacak gecikme sürelerini etkileyecektir. Bu nedenle şebekeye atanabilecek farklı gecikme sürelerinin sayısı, şebekedeki birbirinden bağımsız hatların sahip olduğu gecikme sayılarının çarpımına eşit olacaktır. Belirtilen gecikme sayısı 15 * 378 * 75 = 425250 olarak hesaplanmıştır.

Arama uzayının tamamının taranması için arama uzayının büyüklüğünün hesaplanması zorunlu değildir. Kodlama ile belirlenen şebeke çözüm sayısının doğruluğunu teyit etmek için hesaplamalar gerçekleştirilmiştir. Şekil 3'te sunulan makro kodunun çalıştırılması ile problemin tüm arama uzayı taranmış ve Tablo 3'te sunulan değerler elde edilmiştir.

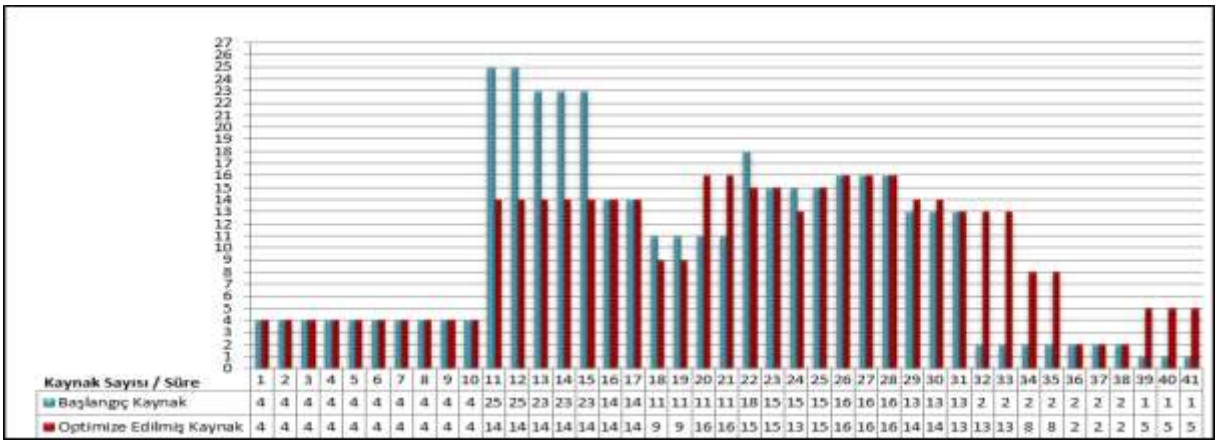
Tablo 3'ün ilk 10 satırında en iyi kaynak dağılımını veren aktivite erteleme süreleri gösterilmiştir. Arama uzayı taranırken 425250 adet iş programı çözümü yapılmıştır. Bulunan değer arama uzayı büyüklüğü hesaplaması ile aynı olduğu için sunulan makro kodunun tüm arama uzayını taradığı görülmektedir. Hesaplama süresi yaklaşık 1.25 saat olarak ölçülmüştür. Hesap süresi üst sezgisel yöntemlere göre uzundur. Bunun en önemli sebepleri arama uzayı taranırken çok fazla sayıda çözüm yapılması ve hesap tablosu uygulamasının çalıştırılabilir dosyalara göre daha yavaş olmasıdır. Fakat garantili biçimde en iyi çözüm elde edildiği için hesap süresi makuldür. Tüm aktivitelerin erken başlangıç zamanında başladığı durumda amaç fonksiyonunun değeri 6178 iken, optimum çözümde amaç fonksiyonu 4932'ye düşürülmüştür.

Tablo 3. Örnek problem 1'in analiz sonuçları.

Parametre	**Değer**
B-erteleme süresi	7
E-erteleme süresi	23
G-erteleme süresi	1
H-erteleme süresi	2
J-erteleme süresi	2
N-erteleme süresi	0
O-erteleme süresi	0
P-erteleme süresi	2
R-erteleme süresi	2
S-erteleme süresi	26
Çözüm sayısı	425250
Hesaplama süresi (dk)	73.75
Başlangıç moment değeri	6178
Optimum moment değeri	4932

Tüm aktivitelerin erken başlangıç zamanında başladığı durumda oluşan kaynak kullanımı ile dengelenmiş kaynak kullanımı Şekil 5'te sunulmuştur. Şekil 5'te mavi çubuklar tüm aktivitelerin erken başlangıç zamanında başladığı durumda oluşan kaynak dağılımını, kırmızı çubuklar ise dengelenmiş kaynak dağılımını, x ekseni günü, y ekseni ise ilgili günde kullanılan duyulan kaynak sayısını belirtmektedir. Çözüm sonrasında kaynak dağılımındaki dalgalanmalar önemli ölçüde azalmıştır. Ayrıca projenin yürütülmesi sırasında ihtiyaç duyulan

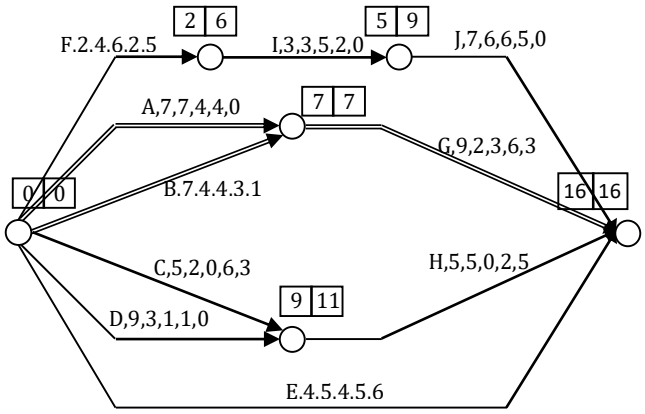
maksimum kaynak miktarı 25'ten 16'ya indirilmiştir. Kaynak dengeleme probleminin çözümü doğrudan kaynak ihtiyacını düşürmeyi amaçlamaz, fakat çözüm sonrasında kaynak ihtiyacında düşüşler de olabilir ve bu durum kaynak dengelemenin sağladığı ikincil faydadır. Analizde kullanılan minimum moment ölçeği maksimum kaynak talebini düşürme eğiliminde olan bir ölçektir [33-36], [38, 39].



Şekil 5. Örnek problem 1'in optimizasyon öncesi ve sonrası kaynak histogramları.

3.2. Örnek Problem 2

Geliştirilen algoritmanın denendiği ikinci problem 10 aktiviteli bir projedir [24]. Problemin şebeke diyagramı ve aktivitelerin kaynak ihtiyaçları aktivite sürelerinin arkasında dört rakam olarak Şekil 6'da sunulmuştur.



Şekil 6. Örnek problem 2'nin şebeke diyagramı, aktivite süreleri ve kaynak kullanımı.

Örnek problemde 4 farklı kaynak türü olduğu için problem çok-kaynaklı kaynak dengeleme problemidir. Amaç fonksiyonunun hesaplanması her kaynak için günlük kaynak kullanımının karelerinin toplanması ile gerçekleşmektedir. Her kaynak için elde edilen değerler kaynak türleri için tayin edilen ağırlıkla çarpılarak toplanmaktadır. Literatürde ağırlık olarak işçilerin yevmiyeleri veya makine kiraları alınmaları yaygındır, fakat problem hipotetik olduğu için tüm kaynak türleri için ağırlıklar 1 alınmıştır.

Problem 2'de kritik hattın dışında birbirinden bağımsız 3 hat bulunmaktadır. Kritik olmayan hatlar, F-I-J, C-D-H ve E hatlarıdır. Belirtilen hatlardan C-D-H hattı C-H ve D-H hatları biçimde 2 ayrı hat olarak da

incelenebilirdi, fakat tek hat olarak incelenmesi sonucu değiştirmemektedir. İlk hat olan F-I-J hattı toplam 12 gün sürmektedir ve bu hat üzerinde bulunan aktivitelerin 4 gün toplam bolluğu bulunmaktadır.

F ve I aktiviteleri 0 gün ertelendiğinde J aktivitesi 0 ila 4 gün olmak üzere 5 farklı biçimde ertelenebilir. I aktivitesi 1 gün ertelendiğinde J aktivitesi 4, I aktivitesi 2 gün ertelendiğinde J aktivitesi 3 farklı şekilde ertelenebilmektedir. F aktivitesinin 0 gün ertelendiği durumda farklı ertelenme sayısı 5'ten 1'e kadar olan sayıların toplamı olan 15'e eşit olacaktır. F aktivitesi 1 gün ertelendiğinde I ve O aktivitelerinin toplam ertelenme sayıları 4'ten 1'e kadar olan sayıların toplamı olan 10'a eşit olacaktır. Benzer şekilde F aktivitesinin 2 gün ertelendiği durum için farklı ertelenme sayısı 3'ten 1'e kadar olan sayıların toplamı olan 6 olacaktır. F-I-J hattı 15 + 10 + 6 + 3 + 2 + 1 =35 farklı erteleme seçeneğine sahiptir. C-D-H hattı 38 farklı biçimde ertelenebilirken, E hattı ise 13 farklı ertelenme seçeneğine sahiptir. İncelenen 3 hattın ertelenme sayılarının çarpımı 17290 sonucunu vermektedir. Problem 2'nin tüm arama uzayı 575 saniyede taranmıştır. Tüm aktivitelerin ertelenme süreleri 0 gün iken MM ölçeği 11290 değerindedir. Optimizasyon sonrası C, D, E, F, H, I ve J aktiviteleri sırası ile 6, 2, 0, 4, 2 ve 4 gün ertelendiğinde MM ölçeği 9438'e düşürülmüştür.

3.3. Diğer örnek problemler

Literatürden derlenen toplam 27 KDP geliştirilen yöntem ile hesap tablosu üzerinde çözülerek optimum sonuçlar elde edilmiştir. Bu sayede yöntemin küçük ve orta büyüklükteki KDP'yi çözebileceği gösterilmiştir. Çözümü gerçekleştirilen problemlerin arama uzayı büyüklüğü, aktivite sayısı ve çözüm süresi Tablo 4'te sunulmuştur.

Tablo 4'te yer alan Analiz 1 sütunu, kaynak kullanım hesaplamalarının hesap tablosu hücrelerinde gerçekleştirildiği küçük ölçekli problemleri içermektedir. Çözülen problemlerin önemli bir kısmının çözüm süresi 20 dakikanın altında kalmaktadır. Aktivite sayısı 6 ve 7 olan küçük problemler ise 1 saniye içerisinde kesin çözülmektedir. Tablo 4'te A1 sütunu altında problemin çözüm süresini kaynak kullanımının HTU ile hesaplanması halinde gerçekleşen hesaplama süresi saniye biriminde gösterilmektedir. Proje süresine ve aktivite sayısına bağlı olarak kaynak kullanımı hesaplamaları çok fazla sayıda hücre işgal etmekte ve hesaplamaları yavaşlatmaktadır. A2 sütunu altında sunulan sonuçlarda ise günlük kaynak kullanımı hesap tablosu hücreleri yerine makro ile hesaplanarak HTU üzerindeki işleme dâhil olan hücre sayısı azaltılmıştır. Bu şekilde yapılan çözümde hesaplama sürelerinin ilk analize göre önemli ölçüde kısaldığı görülmektedir. Ayrıca hesaplama süresi çok uzun olduğu için ilk analizde yarıda kesilen 23 ve 24 numaralı problemler ikinci analizde çözülebilmiştir.

Tablo 4. Literatürden derlenen örnek problemlerin aktivite sayısı, arama uzayı büyüklüğü ilişkisi ve çözüm süreleri

ID	Akt. S.	Arama Uzayı	En İyi Sonuç	A1 Süre	A2 Süre	A3 Süre	Ref
1	9	272	2700	14	1	0,016	[15]
2	6	6	24	<1	<1	0,015	[40]
3	6	3	22	<1	<1	0,015	[40]
4	7	36	641	5	1	0,016	[41]
5	5	11	1357	1	<1	0,015	[42]
6	15	600	3005	8	2	0,015	[46]
7	12	180	5691	6	1	0,016	[45]
8	5	11	428	1	<1	0,016	[27]
9	8	10	853	<1	<1	0,016	[19]
10	9	1953	8050	138	5	0,016	[12]
11	13	98784	6225	5683	477	0,063	[18]
12	8	16	5466	1	<1	0,016	[46]
13	11	5684	993	245	27	0,016	[48]
14	10	17290	9438	1002	51	0,235	[2]
15	11	462	6723	51	2	0,016	[47]
16	18	425250	4932	20649	2995	0,097	[37]
17	11	7134	915	369	21	0,016	[18]
18	11	14364	821	597	52	0,016	[4]

19	16	5126772	11960	195763	22563	6,405	[44]
20	16	45900	3720	533	161	0,062	[45]
21	18	8678	6539	453	55	0,031	[44]
22	9	300	2112	5	<1	0,015	[39]
23	13	7408830	12355	YOK	27327	2,827	[12]
24	14	6030269	10996	YOK	28995	7,592	[46]
25	19	153513360	5608	YOK	YOK	174,99	[49]
26	15	662299680	8971	YOK	YOK	854,30	[49]
27	20	13531140	3059	YOK	YOK	5,627	[50]

Elde edilen iyileştirmeye rağmen daha büyük problemler için HTU ile tüm arama uzayının taranarak KDP çözümünün uygun olmadığı görülmektedir. Bu nedenle geliştirilen yöntem hesaplamaların daha hızlı yürütüldüğü C++ dilinde kodlanmıştır. C++ dilinde geliştirilen yazılımla gerçekleştirilen hesaplama süreleri Tablo 4'te A3 Süre sütununda sunulmuştur. Birinci ve ikinci analizlerde hesaplama süresinin çok uzun olması nedeniyle çözülemeyen 25, 26 ve 27 numaralı problemler bu analizde çözülmüştür.

Hesaplama süreleri C++ yazılımında bulunan ctime kütüphanesi fonksiyonu olan clock() fonksiyonu ile ölçülmüştür. Geliştirilen yöntemin hesaplama süresinin ve elde edilen sonuçların karşılaştırılabilmesi için örnek problemler GA ile 10 defa çözülmüş ve elde edilen sonuçların ortalaması Tablo 5'te sunulmuştur. Analiz 1'de aktivite sayısının karesinin 2 katı kadar, Analiz 2'de ise aktivite sayısının üçüncü kuvvetinin 2 katı kadar iş çizelgesi çözüldüğünde hesaplamalar durdurulmuştur.

Kaynak dengeleme probleminin arama uzayı kritik olmayan aktivite sayısına ve bolluk sürelerine bağlıdır. Arama uzayı projenin büyüyüp karmaşıklaşması ile çok hızlı artmaktadır. Arama uzayının artması ile çözüm süresi de artmaktadır. Bu engel kaynak dengeleme probleminin çözümü için kullanılan tüm çözüm yöntemlerinde bulunmaktadır. Bu çalışmada geliştirilen yöntem de aynı koşullardan etkilenmektedir. Fakat hesaplama süresi uzatılsa da üst-sezgisel yöntemler kesin sonuca ulaşmayı garanti edemezken, önerilen yöntem kesin sonucu garantileyebilmektedir.

Karşılaştırmada kullanılan GA üst-sezgisel optimizasyon algoritması mayoz bölünme ve doğal seleksiyondan esinlenerek geliştirilmiştir [51]. Başlangıçta uygulanabilir çözümleri temsil eden rastgele sayılar kullanılarak üretilen bireylerden oluşan popülasyonla çözüme başlanır. Kaynak dengeleme probleminin çözümü bolluk süresi bulunan aktiviteleri sahip oldukları bolluk sürelerini aşmadan farklı erteleme süreleri atanarak aranır. Bu nedenle popülasyonu oluşturan bireylerin her biri kritik olmayan aktivite sayısı kadar değişken taşır. Değişkenlere ifade ettikleri aktivitenin erteleme süresini belirten değer rastgele atanır. Şebeke koşullarının ihlal edilmemesi için rastgele sayılar 0 ile aktivitenin bolluk süresi arasında olacak şekilde üretilmiştir. Bazı durumlarda kritik olmayan aktivitelere atanan bolluk süreleri tutarlı olmayabilir. Örneğin birbirine seri bağlı öncel ve ardıl aktiviteler için atanan bolluk sürelerinde öncelin gecikmesi ardıl aktiviteye göre daha fazla olabilir ve atanan değerlere göre ardıl aktivite için hesaplanan başlangıç zamanında öncel aktivite tamamlanmamış olabilir. Bu durum şebeke kısıtlarını ihlal etmektedir. Belirtilen şebeke kısıtlarını engellemeyecek şekilde rastgele rakam üretmek oldukça karmaşıktır. Bu nedenle şebeke çözülürken ardıl aktivitenin başlangıç zamanı belirlenirken tüm öncellerinin tamamlanmış olma şartını sağlayan zaman ile ilgili aktivitenin erken başlangıç zamanına tayin edilen gecikme eklenerek elde edilen başlangıç zamanlarından hangisi en geç ise o süre başlangıç zamanı olarak atanarak çözüm yapılmıştır. Bu durumda ardıl aktiviteye atanan gecikme süresi öncel aktiviteye göre daha az olsa dahi öncel aktiviteye göre atanan erteleme süresine göre çözüm yapılır fakat bireylerin taşıdığı erteleme süresi değerleri değiştirilmez.

Mevcut çözümlerin iyileştirilmesi için çaprazlama ve mutasyon operatörleri uygulanmaktadır. Çaprazlama için bireyler rastgele eşleştirilerek yeni bireyler oluşturulur. Çaprazlama için eşitlik 3'te sunulan ifadeyle bireylerin çaprazlama uyumluluk değerleri hesaplanmıştır.

$$CU_i = \frac{(M_i - P\dot{I})}{(PK - P\dot{I})} + \frac{rand(1)}{5} \qquad (3)$$

Eşitlik 3'te CUi i'ninci bireyin çaprazlama uyum indeksini, Mi ise i'ninci bireyin eşitlik 2 kullanılarak hesaplanan moment değerini, PK popülasyonun en kötü değerini, Pİ popülasyonun en iyi değerini rand(1) ise 0 ile 1 arasında oluşturulan rastgele sayısı ifade etmektedir. Bireyler eşitlik 3'e göre küçükten büyüğe sıralanırlar ve ilk "çaprazlama oranı * popülasyon boyutu" kadar birey çaprazlamaya alınır ve çaprazlama için ardışık bireyler eşleştirilir. Denklem 3'te yer alan rastgele sayı eklemenin amacı ilerleyen GA çevrimlerinde sürekli aynı ebeveyn eşleşmelerinin oluşmasını engellemektir. Bu çalışmada tek noktalı çaprazlama uygulanmıştır. Her

çaprazlama için eşleşen birey çiftleri için 1 ile kritik olmayan aktivite sayısı arasında tamsayı rastgele sayı üretilir ve çaprazlama bu konumdan gerçekleştirilir.

Mutasyon için her bireye rastgele sayı atanır ve atanan sayılara göre bireyler sıralanır. Sıralama sonucuna göre ilk "mutasyon oranı * popülasyon boyutu" kadar birey mutasyona tabi tutulur. Mutasyon sırasında 1 ile kritik olmayan aktivite sayısı arasında rastgele tamsayı üretilir ve mutasyonun yapılacağı aktivite belirlenir. Ardından -2 ile +2 arasında bir tamsayı üretilerek aktivitenin erteleme süresine eklenir. Aktivitenin ertelenme süresinin 0 günden az olması halinde erteleme süresi 0 gün, aktivitenin toplam bolluk süresinden fazla olması halinde ise erteleme süresi toplam bolluk süresi olarak düzeltilir. Son olarak mevcut bireyler, çaprazlama ve mutasyonla oluşturulan tüm bireyler eşitlik 3 kullanılarak tekrar sıralanır ve ilk popülasyon sayısı kadar birey saklanır diğer bireyler ise yok edilerek popülasyon sayısı sabit tutulur. Çaprazlama, mutasyon ve doğal seleksiyon işlemleri GA'nın 1 çevrimini oluşturur. GA durma şartları oluşuncaya kadar tekrarlanmıştır.

Tablo 5'te sunulan değerler incelendiğinde Analiz 2 sütununda sunulan hesaplama sayılarının Analiz 1 deki hesaplama sayılarından ciddi miktarda fazla olmasına rağmen yakınsanan çözümde aynı oranda bir iyileşmenin gözlenemediği ortaya çıkmaktadır. Bu durumun en önemli nedenlerinden birisi optimizasyon sürecinde popülasyon bireylerinin çoğunluğunun yerel minimuma yakınsaması ve GA'nın içerdiği çaprazlama ve mutasyon operatörlerinin yakınsanan yerel minimuma yakın çözümler üretmesinden kaynaklanmaktadır. Ayrıca bu çalışmada uygulanan tüm arama uzayının tamamının taranmasının hesaplama süresi en büyük problem için GA'dan birkaç dakika daha uzundur fakat GA yerel minimuma takılan çözüm üretirken önerilen yöntem en iyi çözümü garantili biçimde sunmaktadır.

Tablo 5. Literatürden derlenen örnek problemlerin GA ile çözümünün sonuçları

		Analiz 1			**Analiz 2**		
ID	**Akt. S.**	**Hesaplama Sayısı**	**Hesap Süresi**	**Ortalama Sonuç**	**Hesaplama Sayısı**	**Hesap Süresi**	**Ortalama Sonuç**
1	9	162	0,016	2704	1458	0,046	2704
2	6	72	0,015	24,8	432	0,025	24,4
3	6	72	0,015	22,2	432	0,025	22
4	7	98	0,018	641	686	0,063	641
5	5	50	0,015	1357	250	0,024	1357
6	15	450	0,017	3005,4	6750	0,092	3005
7	12	288	0,017	5707	3456	0,076	5700,2
8	5	50	0,016	431,2	250	0,032	429,6
9	8	128	0,030	866,6	1024	0,233	854,6
10	9	162	0,015	8097,4	1458	0,030	8065,8
11	13	338	0,017	6430,6	4394	0,081	6418,6
12	8	128	0,018	5466	1024	0,069	5466
13	11	242	0,018	1000,8	2662	0,090	994,6
14	10	200	0,018	9447,8	2000	0,083	9453,6
15	11	242	0,016	6728,6	2662	0,052	6723
16	18	648	0,019	4988,6	11664	0,093	4951,8
17	11	242	0,019	923	2662	0,075	918,6
18	11	242	0,015	839	2662	0,034	829
19	16	512	0,020	12094	8192	0,178	12004
20	16	512	0,020	3765,8	8192	0,169	3728,6
21	18	648	0,021	6544,4	11664	0,229	6543,2
22	9	162	0,015	2136,8	1458	0,030	2133,2
23	13	338	0,017	12410,8	4394	0,081	12390,4
24	14	392	0,018	11052,4	5488	0,110	11036,4
25	19	722	0,020	5660,2	13718	0,209	5622,2
26	15	450	0,019	9145	6750	0,143	9063,4
27	20	800	0,021	3110,6	16000	0,253	3092

4. Sonuçlar

Bu çalışmada küçük ölçekli projelerde kaynak dengeleme probleminin tam sonucunu arama uzayının tamamını tarayarak makul hesaplama süresinde elde eden bir yöntem geliştirilmiştir. Yöntem hesap tablosu üzerinde çalışan makro ile uygulanmış ve toplam 27 adet kaynak dengeleme problemi çözülerek yöntemin kesin çözüm elde edebileceği gösterilmiştir.

HTU'nun geliştirilmesi ve makro kodu detaylı biçimde verilerek geliştirilen yöntemin araştırmacılar ve yükleniciler tarafından uygulanabilmesi sağlanmıştır. Bu çalışmada geliştirilen yöntem sayesinde küçük ve orta ölçekli müteahhitler tarafından uygulanması güç olan optimizasyon algoritmalarına ihtiyaç duymadan KDP'nin kesin çözümünü HTU ile elde etmek mümkün olabilecektir. Şebekenin ok diyagramının çizilip hesap tablosuna aktarılıp makro kodunun çalıştırılması ile KDP'nin kesin çözümü elde edilebilecektir. Bu sayede inşaat ve imalatın tüm aşamalarında kaynakların mümkün olan en üst verimde kullanılması mümkün olacak ve maliyetlerde kayda değer düşüşler sağlanabilecektir.

Yöntem hesap tablosu üzerinde uygulanarak başlangıç düzeyinde programlama bilgisine sahip mühendislerin kolaylıkla uygulayabilmesi amaçlanmıştır. Ancak uygulama kolaylığı hesap yavaşlığı engelini getirmiştir. Çalıştırılabilir dosya biçiminde C++ dilinde oluşturulan programlarda bir saniyede 1 milyondan daha fazla iş programı çözebilirken hesap tablosu üzerinde bir saniyede yaklaşık 100 iş programı çözülmektedir. Bu durum hesap tablosunun daha büyük problemlerin çözümünde kullanılabilirliğini düşürmüştür.

Arama uzayının taranmasının hesap yükü ile Genetik Algoritmanın hesap yükü karşılaştırılmış ve tüm arama uzayının taranmasının hesap yükünün GA'nın hesap yükünden çok daha az olduğu belirlenmiştir. Tablo 4 ve 5'te algoritmaların iş çizelgesi hesaplama sayıları verilmiştir. Aktivite sayısının 10'dan az olduğu problemlerde arama uzayının tamamının taranması daha az şebeke analizi gerektirmekte ve daha az hesaplama ile garantili biçimde kesin sonucu vermektedir.

Üst-sezgisel yöntemlerde popülasyonun taşıdığı genler ve amaç fonksiyonunun bellekte saklanması gerekmektedir. Büyük projeler için popülasyonun bellek ihtiyacı işlemcinin tampon belleğinin boyutunu aşıp Random Access Memory'de (RAM) saklanmasına yol açmaktadır. Arama uzayının taranması yönteminde saklanması gereken veriler şebekedeki olay sayısı kadar erken olay ve geç olay zamanları, aktivitelerin bolluk süreleri ve günlük kaynak kullanım değerleridir. Belirtilen verilen tampon bellekte saklanabilecek kadar az yer kaplamaktadır. Tampon bellek RAM'e göre çok daha hızlı erişilebilen bir bellek türü olduğu için arama uzayının taranması yönteminde daha fazla şebeke analizi yapılmasına rağmen hesaplama süresi GA'dan daha kısa olmaktadır.

Problemin zorluğu, yerel minima ile optimum arasındaki parametre değerlerinin benzerliği ve popülasyon büyüklüğüne bağlı olarak GA'nın arama uzayını tam olarak tarayamama ve yerel minimaya takılma riski bulunmaktadır. Yerel minimaya takılma durumunda birkaç yinelemeden sonra popülasyonun büyük çoğunluğu yerel minimaya eşit olmakta ve elde edilen sonucun iyileştirilme olasılığı düşmektedir. Kaynak dengeleme problemi doğası gereği çok fazla yerel minima içermektedir ve yerel minimadan kurtulmak için aynı anda birden fazla aktivitenin bolluk sürelerinin değiştirilmesi gerekebilmektedir. Bu durum yerel minimadan kurtulabilme olasılığını iyice düşürmektedir. Ayrıca ilerleyen yinelemelerde bireyler birbirlerine çok yakın genler taşır duruma geldikleri için daha önce denenen şebeke çözümlerinin tekrar denenme olasılığını arttırmaktadır. Tabu arama benzeri yöntemler ise büyük problemler için önemli miktarda bellek kullanımı gerektirmekte ve hesaplamaları yavaşlatmaktadır.

Bu çalışmada kaynak dengeleme probleminin arama uzayının belirlenmesi için bir yöntem önerilmektedir. Bu sayede ilgili problemde kaç farklı uygulanabilir iş programının oluşturulabileceği hesaplanabilecektir. Bu değer problemin zorluk derecesi hakkında fikir vermektedir. Kaynak dengeleme probleminin çözümünde yaygın olarak üst-sezgisel algoritmalar kullanılmaktadır. Çözüm için yapılan çözüm sayısı ile arama uzayının büyüklüğü kıyaslanarak uygulanan çözüm algoritmasının başarısı daha iyi biçimde ölçülebilecektir.

Ayrıca problemin arama uzayı büyüdükçe üst-sezgisel yöntemler kullanılarak optimum veya yakın-optimum çözüm elde edebilmek için popülasyon boyutunun ve çevrim sayısının arttırılması gerekmektedir. Arama uzayının büyüklüğü belirlendikten sonra popülasyon boyutu ve hesaplama sayısı daha hassas biçimde belirlenebilecek ve yetersiz hesaplama sayısı atayarak optimum çözümden daha uzak bir çözüm elde etme riski önlenebilecektir. Ayrıca aşırı hesaplama sayısı atanarak hesaplama süresinin gereğinden fazla uzaması da önlenebilecektir.

Bu çalışmanın literatüre katkısı, sezgisel ve matematiksel yöntemlere kıyasla uygulaması daha kolay ve hızlı bir yöntem geliştirilerek kaynak dengeleme probleminin optimum çözülmesidir. Geliştirilen yöntem literatürden derlenen en büyüğü 20 aktiviteli olan 27 KDP üzerinde denenmiş ve tam çözüme ulaşılmıştır. Aynı

yöntemle kaynak kısıtlı iş programı problemi de çözülebilmektedir [52]. Geliştirilen yöntem mevcut haliyle küçük ölçekli problemleri başarı ile çözebilmektedir. Fakat Tablo 4'te gösterildiği üzere problemin arama uzayı aktivite sayısına bağlı olarak üstel artmaktadır. Çözüm süresi arama uzayına bağlı olarak çok hızlı artacak ve tüm arama uzayı taranamadan hesaplamaların durdurulması gerekecek ve optimum sonucun elde edilebilmesi kesin olamayacaktır. Belirtilen sorunun önüne geçmek için hesaplamaların hızlandırılması ve paralel hesaplamaların gerçekleştirilmesi gereklidir. Belirtilen hususların sağlanması ile önerilen yöntemin kapasitesi ve uygulanabilirliği arttırılabilecektir. Ayrıca bu çalışmada şebekeyi oluşturan hatların küçük parçalara ayrılması manuel olarak gerçekleştirilmiştir. Bu işlemi insan müdahalesi olmadan gerçekleştirebilen bir algoritmanın geliştirilmesi de yöntemin uygulanabilirliğini arttıracaktır.

Etik Standartların Beyanı

Bu makalenin yazarları çalışmalarında kullandıkları materyal ve yöntemlerin etik kurul izni ve/veya yasal-özel bir izin gerektirmediğini beyan ederler.

Çıkar Çatışması

Bu çalışmada herhangi bir çıkar çatışması yoktur.

Kaynaklar

[1] Rieck J, Zimmermann J. Handbook on Project Management and Scheduling. Exact methods for resource leveling problems, Springer International Publishing: 2015; 1, 361-387.
[2] Demeulemeester E, Herroelen W. Project Scheduling: A Research Handbook, Kluwer Academic Publishers, Boston, USA, (2002).
[3] Harris RB. Resource and arrow networking techniques for construction, Wiley, New York, 1978.
[4] Harris RB. Packing method for resource leveling (PACK), J. Constr. Eng. Manage., 1990; 116(2), 331-350.
[5] Hiyassat MAS. Modification of minimum moment approach in resource leveling, J. Constr. Eng. Manage., 2000; 126(4), 278-284.
[6] Hiyassat MAS. Applying modified minimum moment method to multiple resource leveling, J. Constr. Eng. Manage., 2001; 127(3), 192-198.
[7] Hegazy T. "Optimization of resource allocation and leveling using genetic algorithms, J. Constr. Eng. Manage., 1999; 125(3), 167-175.
[8] Hegazy T, Kassab M. Resource optimization using combined simulation and genetic algorithms, J. Constr. Eng. Manage., 2003; 129(6): 698-705.
[9] Hossein HD, SeifiSA, Shariat SY. Efficient hybrid genetic algorithm for resource leveling via activity splitting, J. Constr. Eng. Manage., 2010; 137(2): 137-146.
[10] Ponz-Tienda JL, Yepes V, Pellicer E, Moreno-Flores J. "The resource leveling problem with multiple resources using an adaptive genetic algorithm", Automation in Construction, 2013; 29: 161-172.
[11] Zheng DX, Ng, ST, Kumaraswamy MM. "GA-based multiobjective technique for multi-resource leveling", Bridges, 2003; 10(40671): 29.
[12] Leu SS, Yang CH, Huang JC. Resource leveling in construction by genetic algorithm-based optimization and its decision support system application, Automation in construction, 2000; 10(1): 27-41.
[13] Karaköse, E. Sürü İnsansız Hava Araçlarının Görev Paylaşımı için Genetik Algoritma Tabanlı Bir Yaklaşım. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 2022; 34(1), 351-360.
[14] Li Z, Wuliang P, Zhongliang Z. An ant colony system for solving resource leveling problem, In IEEE International Conference on Intelligent Computation Technology and Automation (ICICTA), 2010; 1: 489-492.
[15] Geng JQ, Weng LP, Liu SH. An improved ant colony optimization algorithm for nonlinear resource-leveling problems, Computers & Mathematics with Applications, 2011; 61(8), 2300-2305.
[16] Wang Q, Qi JX. Research on resource leveling problem under resource constrained condition, IEEE International Conference on Machine Learning and Cybernetics, 2009; 2: 901-906.

[17] Li H, Dong X. "Multi-mode resource leveling in projects with mode-dependent generalized precedence relations", Expert Systems with Applications, 2018; 97, 193-204.
[18] Son J, Skibniewski MJ. Multiheuristic approach for resource leveling problem in construction engineering: Hybrid approach, J. Constr. Eng. Manage., 1999; 125(1), 23-31.
[19] Qi, JX, Wang Q, Guo XZ. Improved particle swarm optimization for resource leveling problem, IEEE International Conference on Machine Learning and Cybernetics, 2007; 2: 896-901.
[20] Tanyıldızı, E., ve Demir, G. Nümerik Optimizasyon için Kaotik Altın Sinüs Algoritması. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 2019; 31(1), 91-97.
[21] Akyol, S. Global Optimizasyon için Yeni Bir Hibrit Yöntem: Kaya Kartalı Optimizasyonu-Tanjant Arama Algoritması. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 2021; 33(2), 721-733.
[22] Yetiş, H., ve Karaköse, M. Kuantum Uyarlamalı Genetik Algoritmalar için Çözüm Kalitesini Artıracak Yeni Bir Yaklaşım. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 2021; 33(1), 71-79.
[23] Aydemir, S. B. Küresel Optimizasyon için Gauss Kaotik Haritası ile Kartal Optimizasyonu. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 2022; 34(1), 85-104.
[24] Demeulemeester E, Herroelen W. A branch-and-bound procedure for the multiple resource-constrained project scheduling problem, Management science, 1992; 38(12): 1803-1818.
[25] Neumann K, Zimmermann J. Procedures for resource leveling and net present value problems in project scheduling with general temporal and resource constraints, European Journal of Operational Research, 2000; 127(2): 425-443.
[26] Nübel H. The resource renting problem subject to temporal constraints, OR-Spektrum, 2001; 23(3): 359-381.
[27] Easa SM. Resource leveling in construction by optimization, J. Constr. Eng. Manage., 1989; 115(2): 302-316.
[28] Hariga M, El-Sayegh SM. Cost optimization model for the multiresource leveling problem with allowed activity splitting, J. Constr. Eng. Manage., 2010; 137(1): 56-64.
[29] Karaa FA, Nasr AY. Resource management in construction", J. Constr. Eng. Manage., 1986; 112(3), 346-357.
[30] Gather T, Zimmermann J, Bartels JH. Exact methods for the resource levelling problem, Journal of Scheduling, 2011; 14(6): 557-569.
[31] Mattila KG, Abraham DM. Resource leveling of linear schedules using integer linear programming, J. Constr. Eng. Manage., 1998; 124(3): 232-244.
[32] Özcan, H. Comparison of particle swarm and differential evolution optimization algorithms considering various benchmark functions, Politeknik Dergisi, 2017; 20(4): 899-905.
[33] Erzurum, T., ve Bettemir, Ö. H. Kaynak Dengeleme Probleminin Arama Uzayını Paralel Programlama ile Tarayarak Kesin Çözümü, Teknik Dergi, 2021; 32(3): 10767 – 10805.
[34] Bettemir ÖH, Erzurum T. Comparison of resource distribution metrics on multi-resource projects, Journal of Construction Engineering, Management & Innovation, 2019; 2(2): 93-102, ().
[35] Bettemir ÖH, Erzurum T. Comparison of Resource Distribution Metrics on Small Projects, International Civil Engineering and Architecture Conference, Trabzon, Turkey, (2019).
[36] Erzurum T. Kaynak Dengeleme Probleminin Optimum veya Yakın Optimum Çözülmesi, Yüksek Lisans Tezi, T.C. İnönü Üniversitesi, Malatya, Türkiye, (2019).
[37] Bandelloni M, Tucci M, Rinaldi R. Optimal resource leveling using non-serial dyanamic programming, European Journal of Operational Research, 1994; 78(2): 162-177.
[38] Erzurum T. Bettemir, ÖH. Kaynak Dengeleme Problemlerinin Arama Uzayının Belirlenmesi Determination of Search Domain of Resource Leveling Problem, Uluslararası Katılımlı 7. İnşaat Yönetimi Kongresi, 437-453, Samsun, Türkiye, (2017).
[39] Erzurum T, Bettemir ÖH. Optimum or Near-Optimum Resolution of Resource Leveling Problems with Spreadsheet Application, 5th International Project and Construction Management Conference (IPCMC 2018):1285-1299, Northern Cyprus, (2018).
[40] Gordon, J., ve Tulip, A. Resource scheduling, Int. J. Proj. Manage., 1997; 15(6): 359-370.
[41] Rui, L., and Xiao-ya, W., Using elitist particle swarm optimization to facilitate resources leveling optimization analysis, In IEEE 3rd IEEE Conference on Industrial Electronics and Applications, ICIEA: 90-95, (2008).
[42] Abeyasinghe, M. C. L., Greenwood, D. J., ve Johansen, D. E., An efficient method for scheduling construction projects with resource constraints, International Journal of Project Management, 2001; 19(1), 29-45.
[43] Younis, M. A., ve Saad, B., Optimal resource leveling of multi-resource projects, Computers and industrial engineering, 1996; 31(1): 1-4.
[44] Mutlu, M. Ç. A branch and bound algorithm for resource leveling problem, MSc Thesis, METU, (2010).
[45] Newitt, J. S. Construction Scheduling: Principles and Practices, Pearson Prentice Hall, NJ, (2004).
[46] Hinze, J. W. "Construction Planning and Scheduling, 3rd Edition, Pearson Prentice Hall, Upper Saddle River, NJ, (2006).
[47] Mubarak, S. A., Construction Project Scheduling and Control, 2nd Ed. Wiley, India (2010).
[48] Akpan, E. O. P., Resource smoothing: a cost minimization approach, Production Planning & Control, 2000; 11(8): 775 – 780.
[49] Stevens, J.D., Techniques for Construction Network Scheduling, McGraw-Hill, New York, (1990).
[50] El-Rayes, K., Jun, D.H. Optimizing resource leveling in construction projects, J. Constr. Eng. Manage., 2009; 135(11): 1172-1180.

[51] Holland J., Adaptation in Natural and Artificial Systems, University of Michigan Press, Michigan, (1975).
[52] Bettemir, Ö. H., ve Cakmak, D. (2021). Tüm Arama Uzayını Tarayarak Küçük Ölçekli Kaynak Kısıtlı Proje Çizelgeleme Problemlerinin Çözümü. Aksaray University Journal of Science and Engineering, 5(2), 92-112.