KAYNAK KISITLI ÇOKLU PROJE PROGRAMLAMA PROBLEMİ İÇİN TAVLAMA BENZETİMİ ALGORİTMASI

Tuba Yakıcı AYAN(*)

Özet: Bu makalede kaynak kısıtlı çoklu proje programlama problemini (KÇPP) çözmek için geliştirilen için bir tavlama benzetimi (TB) algoritması sunulmaktadır. Programlamanın amacı, proje gecikmeleri ve faaliyet beklemelerinden kaynaklanan maliyetler toplamını en küçüklemektir. Bu yönü ile çalışma benzer çalışmalardan ayrılmaktadır. Kısıtlara uygun çözüm gösterimi, öncelik ilişkilerine uygun bir faaliyet listesine ve kaynak tahsislerine dayanmaktadır. TB algoritmalarında büyük çoğunlukla kullanılandan farklı olarak bu çalışmada daha yavaş bir soğutma planı ve iki ayrı durdurma ölçütü kullanılmaktadır. Geliştirilen algoritma rastgele oluşturulan büyük bir problem üzerinde test edilmekte ve elde edilen sonuçlar önerilen algoritmanın etkinliğini doğrulamaktadır.

Anahtar Kelimeler: Proje programlama, Kaynak kısıtı, , Tavlama benzetimi, Soğutma planı.

Abstract: In this paper, a simulated annealig (SA) algorithm developed to solve resource constrained multi-project problem is presented. Object of programming is to minimize project delays plus standby times of activities. From this perspective, this study differs from similar studies. The feasible solution representation is based on a precedence feasible list of activities and resurce assignments. In this study unlike traditional SA algorithms , a very slow cooling schedule and two different stop criteria are used. The algorithm is tested on a randomly generated large problem. The results confirm the effectiveness of the proposed algorithm.

Keywords: Project programming, Resource constraint, Simulated annealing, Cooling Schedule.

I. Giriş

Projeler belirli bir sıra ile icra edilmesi gereken çok sayıda faaliyetten oluşurlar. Bu sıralama, faaliyetler arası öncelik ilişkileri olarak adlandırılır. Kaynak kısıtlı proje programlama (KPP), belirli kaynakların projelere tahsis edilmelerini ve projede yer alan faaliyetlerin başlangıç ve bitiş zamanlarının belirlenmesini içeren karmaşık bir karar verme sürecidir. Kaynakların sınırlı fakat periyottan periyota yenilenebilir olmaları ve aynı anda gerçekleştirilmesi gereken birden fazla projenin varlığı durumunda problem çok daha karmaşık hale gelir. Kaynak kısıtlı çoklu proje programlama problemi (KÇPP), kendi içinde tek modlu ve çok modlu olmak üzere iki kategoriye ayrılır: Çok modlu problemde her bir faaliyetin birden fazla icra tarzı vardır. Tek modlu problemde ise her faaliyet ancak bir şekilde icra edilebilir. Söz konusu icra tarzları ise faalyet süreleri ve kaynak ihtiyaçları ile karakterize edilir. Bu çalışmada tek modlu durum dikkate alınmaktadır.

(*) Yrd. Doç. Dr., Karadeniz Teknik Üniversitesi İİBF Ekonometri Bölümü

KÇPP probleminde pek çok farklı amaç sözkonusu olabilmektedir. Bu amaçlardan en yaygın kullanılanlar proje tamamlanma süreleri toplamının en küçüklenmesi, en uzun proje tamamlanma süresinin en küçüklenmesi ve proje maliyetleri toplamının en küçüklenmesidir (Boctor, 1996, s.2335). Bu çalışmada, proje gecikme maliyetlerinin yanı sıra faaliyetlerin erken bitmelerinden kaynaklanan atıl süre maliyetleri de dikkate alınarak toplam maliyet en küçüklenmeye çalışılmaktadır.

Proje programlamada kullanılan Gantt şeması, CPM ve PERT gibi geleneksel araçlar, kaynakların sınırsız olduklarını varsayarlar ve sadece tek projeye uygulanabilirler. Bu nedenle kaynak kısıtlı çok projeli problemlerin çözümü için çok daha gelişmiş araçlara ihtiyaç vardır. Söz konusu araçlardan biri bu çalışmada kullanılan tavlama benzetimi (TB) algoritmasıdır. TB, kombinasyonel en iyileme problemlerini katıların tavlama sürecinden yararlanarak çözmek için geliştirilmiş bir yerel arama algoritmasıdır. Algoritma temel olarak, Metropolis vd. (1953) tarafından bir fiziksel tavlama sürecini simüle etmek için geliştirilmiş olan monte carlo metoduna dayanmaktadır. TB ilk olarak Kirkpatrick vd. (1983) ve Cerny (1985) tarafından kombinasyonel en iyileme problemlerine uygulanmıştır. Daha sonra Van Laarhoven ve Aarts (1987) ve Aarts ve Korst (1989) tarafından iyice geliştirilmiştir. TB algoritması bir başlangıç çözümünden başlayarak pek çok komşu çözüm arasında arama yapmaktadır. Algoritmayı aynı sınıftaki diğer algoritmalardan ayıran ana özellik, iyi çözümleri kesin olarak kabul ederken kötü çözümleri kesin olarak reddetmemesidir. Bir başka deyişle amaç fonksiyonunun değerini kötüleştiren bir çözümün kabul edilme olasılığı daima sıfırdan büyük olmaktadır. Bunun gerekçesi, algoritmanın bir yerel optimuma takılmasını engellemektir.

Çalışmanın ilk bölümünde KÇPP problemi ile ilgili literatüre değinilmekte izleyen bölümde ise problemin özellikleri açıklanmakta ve bir matematiksel formülasyon geliştirilmektedir. Dördüncü bölüm TB algoritmasının genel adımlarını ve algoritma parametrelerinin belirlenmesine dair bilgiler içermektedir. Beşinci bölümde başlangıç ve komşu çözümlerin oluşturulması için iki algoritma geliştirilerek TB algoritmasına dahil edilmektedir. Altıncı bölümde algoritma işlerliğini göstermek amacı ile örnek problemler sunulmaktadır. Son bölüm çalışmadan elde edilen bulguları ve önerileri içermektedir.

II. Literatür Araştırması

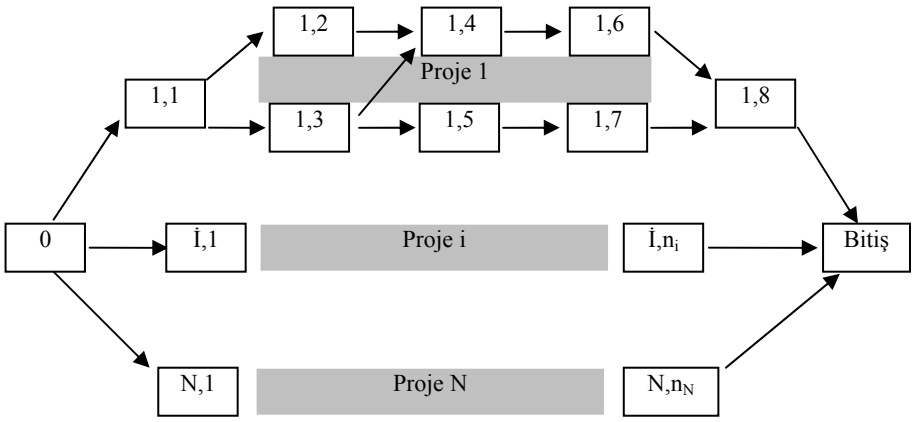
KÇPP problemi, KPP probleminin bir genellemesidir. KPP probleminin çözümü için pek çok yaklaşım geliştirilmiştir. Ancak birden fazla projeyi aynı anda programlamayı içeren KÇPP problemi için az sayıda çalışma yapılmıştır. Literatürde çoğu sezgisel algoritmalar olmakla birlikte, nispeten küçük KÇPP problemlerini çözmek için önerilmiş birkaç kesin metot mevcuttur. Pritsker vd. (1969) problemin 0-1 doğrusal programlama formülasyonunu vermiş iken Mohanty ve Siddiq (1989) problemi çözmek için tamsayılı hedef programlama

yaklaşımı önermiştir. Bock ve Patterson (1990) da dal-sınır yaklaşımı uygulanmıştır. Drexl (1991) de dal-sınır ve dinamik programlamadan oluşan melez bir algoritma geliştirilmiştir. Deckro vd. (1991) ve Vercellis (1994) de ise problemi çözmek için parçalama algoritmaları önerilmiştir. Problemin çözümü için Linyi ve Yan (2007) tarafından yapılan çok yeni bir çalışmada ise, bir "sürü en iyilemesi" yaklaşımı kullanılmıştır.

KÇPP problemi ile ilgili literatür incelendiğinde karşılaşılan sezgisel yöntemlerin çoğunluğu öncelik kuralı esaslı metotlardır. Çünkü bu tür metotlar her zaman en iyi sonucu veremeseler de büyük hacimli problemler için hızlı ve iyi sonuçlar vermektedirler (Kolish ve Hatmann, 1998). Çoklu proje programlama alanında ilk ve önemli çalışmalardan biri olan Fendley (1968) de "en küçük atıl zamanlı faaliyeti önce programlama" şeklinde bir öncelik kuralı kullanılmıştır. Ancak seçilen öncelik kuralının performansı, elbette gerçekleştirilmek istenilen amaca bağlı olarak değişecektir. Kurtuluş ve Davis, (1982) her biri 2–6 birim kaynak gereksinimi olan 34–63 faaliyetli birden fazla projeden oluşan problemler üzerinde, ortalama proje gecikme süresini en küçükleme amacı ile altı ayrı öncelik kuralını test etmişlerdir. Ayrıca Kurtuluş ve Narula (1985) de proje gecikmelerine cezalar yükleme yaklaşımı kullanılmıştır. Dumond ve Mabert (1998), 6–49 faaliyet ve 1- 3 tür kaynak bulunan çok projeli problemleri incelemişler ve proje teslimat tarihlerinin yönlendirdiği bir programlamanın iyi işlediği sonucuna varmışlardır. Tsubakitani ve Decro (1990) tarafından, her biri 100 den fazla faaliyetten ibaret olan 50 den fazla proje içeren problemler için, projeleri güncelleme olanağı veren bir proje kontrol metodu geliştirilmiştir. Lawrence ve Morton (1993) proje teslimat tarihleri esaslı bir algoritma geliştirmiştir. Dikkat çekici bir diğer çözüm yöntemi ise Wiley vd. (1998) deki, çok sayıda projeyi programlamak için Dantzig-Wolfe (Dantzig ve Wolfe, 1960) parçalaması kullanan algoritmadır. Söz konusu algoritma, en iyi çözümün yanı sıra karar vericiye kendi deneyim ve risk toleransına göre alternatif çözümler sunabilme yeteneğine sahiptir. Bunlara ilaveten Blazewicz vd. (1993), KÇPP probleminde kaynak kısıtları için bir sınıflama planı önermiştir. Ash ve Korst (1999) da faaliyet programlama sezgiselini seçmek için mevcut proje verilerini kullanan bir deterministik simulasyon önerilmiştir. Lova ve Tormos. (2000) de herhangi bir sezgisel algoritma ile bulunan çözümü iyileştirmek için çok kriterli sezgisel bir algoritma geliştirilmiştir. Shankar ve Nagi (1996), Boctor (1996) ve Jozefowska vd. (2001) tavlama benzetimi algoritması kullanan çalışmalardır. Gonçalves vd. (2004) özel bir amaç fonksiyonlu KÇPP problemi için bir genetik algoritma geliştirmişlerdir. Kim vd. (2005), Bouleimen ve Lecocq (2000), Chiu ve Tsai (2002), Okada vd. (2008), Homberger (2007) dikkate değer diğer bazı çalışmalar olarak verilebilir.

III. Problem Açıklaması ve Formülasyonu

Şekil 1 de gösterilen problem, kısaca her biri n_j adet faaliyetten oluşan N adet projenin belli bir amacı gerçekleştirmek üzere programlanmaları problemidir. Her bir projenin önceden belirli olan bir teslimat tarihi bulunmaktadır ve teslimat tarihlerindeki gecikmeler firmaya birim zaman başına sabit bir maliyet (a_i) yüklemektedir. Söz konusu sabit maliyet gecikme tazminatları ve proje bedelinin tahsilatındaki gecikmelerden kaynaklanmaktadır. Problemde dikkate alınan ikinci bir maliyet öğesi (b_{ij}) ise herhangi bir faaliyetin erken bitip kaynak yokluğundan dolayı beklemesi durumunda, o faaliyeti gerçekleştiren işçi ve makinenin atıl kalması ile ortaya çıkmaktadır. Bu çalışmada gerçekleştirilmek istenen amaç, toplam maliyetlerin en küçüklenmesidir. Problemde öncelik kısıtları ve kaynak kısıtları olmak üzere iki tür kısıt söz konusudur. Öncelik kısıtları, bir projedeki her bir faaliyetin ancak kendisinden önce bitmiş olması gereken bütün faaliyetler tamamlandıktan sonra başlayabilmesi şeklinde ifade edilebilir. N adet proje, K adet farklı kaynak kullanılarak gerçekleştirilebilmekte ve her bir faaliyet k tipi kaynaktan r_{ijk} birime gereksinim duymaktadır. Söz konusu kaynaklar hammadde ve yardımcı malzemeden oluşmakta ve tamamı firma dışından sabit periyotlarla (t_k) ve sabit miktarlarda (R_k) satın alınabilmektedir. Aynı türden kaynak farklı projelerde kullanılabileceğinden dolayı projelerin kaynak kısıtları açısından bir birlerine sıkı şekilde bağımlı ancak öncelik kısıtları açısından tamamen bağımsız oldukları açıktır. Bu açıklamalar ışığında problemin formülasyonu aşağıda verilmektedir.



Şekil 1: *Proje Şebeke Örneği*

(i,j) :Proje$_i$ deki j yinci faaliyet

P_{ij} : Proje$_i$ deki j yinci faaliyete başlamadan önce bitmiş olması gereken faaliyetler kümesi

DD_i : Proje$_i$ nin teslimat tarihi

CD_i : Oluşturulan programda proje$_i$ nin tamamlanma tarihi

S_{ij} : (i,j) faaliyetinin başlama zamanı

R_k : k tipi kaynağın firmaya sabit geliş miktarı

t_k : k tipi kaynağın firmaya sabit geliş periyotu

r_{ijk} : (i,j) faaliyetinin k tipi kaynaktan gereksim duyduğu miktar

R_{kt} : t zamanında eldeki k tipi kaynak miktarı

i = 1,2,….,N j = 1,2…..,n_i k = 1,2,…,K

a_i = i yinci projenin birim gecikme maliyeti

b_{ij} = i yinci projedeki j yinci faaliyette boş geçen sürenin birim maliyeti

$[\ ]$:tamsayılı bölme işlemi

$$Z_{\min} = \sum_{i=1}^{N} a_i Max\left(CD_i - DD_i, 0\right) + \sum_{i=1}^{N}\sum_{j=1}^{n_i} b_{ij} Max\left(S_{ij} - Max\left(S_{im} + t_{im}\right), 0\right),\ im \in P_{ij} \quad (1)$$

Öncelik kısıtları:

$$S_{ij} \geq S_{im} + t_{im}\ , im \in P_{ij} \quad (2)$$

Kaynak kısıtları:

$$r_{ijk} \leq R_k - \sum_{S_{lm} \leq S_{ij}} r_{lmk} + \left[1 - \frac{S_{ij}\left(Mod\ t_k\right)}{t_k}\right] R_k \quad (3)$$

Amaç fonksiyonu (1), her biri problemin bir karar değişkenini ifade eden iki ayrı maliyet öğesinin ağırlıklı toplamından ibarettir. Burada ağırlıklar birim maliyetlerdir. Karar değişkenlerinden birincisi projelerin bitiş zamanlarındaki gecikme sürelerinden oluşurken diğeri ise kaynak yokluğundan kaynaklanan bekleme sürelerini ifade etmektedir.

IV. Tavlama Benzetimi Algoritması

Bir TB algoritması genel olarak bir başlangıç çözümü, komşu çözümler oluşturma yöntemi ve bir tavlama programından oluşmaktadır. Tavlama sürecinin işlevi, yeterince yüksek bir sıcaklıktaki bir çözümden başlayıp sıcaklığı aşamalı olarak azaltarak iyi ve kötü çözümler arasında dolaşmak ve en

sonunda en iyi çözüme ulaşmaktır. C (.) bir en küçükleme probleminde herhangi bir çözümün amaç fonksiyonu değerini göstermek üzere, TB nin genel adımları aşağıdaki gibi özetlenebilir.

Adım 1: Herhangi bir sezgisel algoritma ile bir başlangıç çözümü oluştur (S)
Adım 2: Mevcut çözüm X=S ve en iyi çözüm M = S dir.
Adım 2: Başlangıç sıcaklığını ayarla (T_0 >0), devre sayacını 1 e ayarla(k = 1)
Adım 3: Aşağıdaki döngüyü L defa tekrarla (L devre uzunluğudur)
Adım3.1: Herhangi bir sezgisel algoritma ile bir komşu çözüm oluştur ($S^{'}$)
Adım3.2: $\Delta_1 = C(S^{'}) - C(X)$ hesapla.
Adım 3.3: Eğer $\Delta_1 \geq 0$ ise adım 4 e git. Aksi halde komşu çözümü kabul et, $X = S^{'}$
Adım 3.3.1: $\Delta_2 = C(S^{'}) - C(M)$ hesapla
Adım 3.3.2: Eğer $\Delta_2 \geq 0$ ise, adım 5 e git. Aksi halde $M = S^{'}$, adım 5 e git.
Adım 4: Düzgün dağılımdan $[0,1]$ aralığında bir rasgele sayı (RS) çek. Eğer $RS \geq e^{\frac{-\Delta_1}{T_k}}$ ise adım 5 e git. Aksi halde komşu çözümü kabul et, $X = S^{'}$
Adım 5: Eğer belli bir durdurma koşulu gerçekleşmişse, dur. Aksi halde $T_{k+1} = F(T_k)$ ve k = k+1. Adım 3 e git.

Yukarıda adım 3 ve adım 5 arasında özetlenen bir tavlama süreci dört ana bileşene sahiptir ve TB algoritmasının etkinliğini ve hesaplama süresini esasen bu bileşenlere ilişkin seçimler belirlemektedir. Söz konusu bileşenler, başlangıç sıcaklığı (T_0), soğutma fonksiyonu (F(.),) devre uzunluğu (L) ve durdurma ölçütleridir. Tavlama süreci her biri birçok çözümün değerlendirilmesini içeren ve devre olarak adlandırılan sıcaklık periyotlarından oluşmaktadır. Sıcaklık (T) algoritmanın kötü çözümleri kabul etme olasılığını kontrol eden bir parametredir. Yüksek bir sıcaklık tamamen rasgele bir yürüyüşe neden olarak çözüm süresini arttırabilir ya da kötü bir performansa yol açabilir. Düşük sıcaklıklar ise en iyiye giden yolun gözden kaçırılmasına yol açabilir. Bu nedenle sıcaklık başlangıçta yüksek tutulup araştırma ilerledikçe aşamalı olarak azaltılmalıdır.

A. Başlangıç Sıcaklığı (T_0)

Kirkpatrick vd. (1983), başlangıç sıcaklığının (T_0) seçimi için tavlamaya başlamadan önce bir deneme koşumu yapmayı önermişlerdir. Onların önerisine göre, bu denemede kabul edilmiş olan kötü çözümlerin toplam çözümlere oranı önceden belirlenmiş olan başlangıç kabul olasılığına (P_0) eşit olmalıdır. Burada kötü çözüm, amaç fonksiyonu değeri açısından mevcut çözümünden daha kötü olma anlamında kullanılmaktadır. Bu yönteme göre, komşu çözümler arasında pek çok geçiş yapılmakta ancak sadece kötü geçişlerin amaç fonksiyonunda yarattıkları ortalama artış (Δ^-) dikkate alınmakta ve $T_0 = \Delta^-/\ln P_0$ eşitliğine göre başlangıç sıcaklığı hesaplanmaktadır. Bu yöntem çok yaygın olarak kullanılmakla birlikte başka yöntemler de önerilmiştir.

Bunlara iki örnek Huang vd. (1986), Connoly (1990) daki yaklaşımlardır. Huang vd. (1986) başlangıç sıcaklığını belirlemek için Δ^- yerine çözüm uzayında rasgele bir yürüyüşle elde edilen amaç fonksiyonu değerlerinin standart sapmasını (σ) kullanmayı önermişlerdir. Connoly (1990) bir deneme koşumunda komşu çözümler arasında maliyet fonksiyonu değişmelerinden (Δ) yararlanarak başlangıç sıcaklığını $T_0 = \Delta_{min} + 0{,}1(\Delta_{max} - \Delta_{min})$ şeklinde belirlemektedir.

B. Soğutma Fonksiyonu ($F(T_k)$)

Sıcaklık birçok çalışmada Kirkpatrick vd. (1983) tarafından geliştirilmiş olan sabit oranlı soğutma fonksiyonu ile azaltılmaktadır. Bu yöntemde sıcaklıklar $T_{k+1} = \alpha T_k$ şeklinde belirlenmektedir. Burada $0<\alpha<1$ önceden belirlenmesi gereken ve soğutma oranı olarak adlandırılan bir parametredir. Potts ve Van Wassenhove (1991) $\alpha = (T_M/T_0)^{1/(M-1}$ şeklinde bir bağıntı önermiştir. Burada M toplam devre sayısını ve T_M son devredeki sıcaklığı göstermektedir Ancak α yı belirlemede bu yöntemi uygulayabilmek için formüldeki parametrelerin önceden belirlenmiş olmaları gerekmektedir. Lundy ve Mees (1986) araştırma ilerledikçe soğutmayı yavaşlatan bir yaklaşım geliştirmişlerdir. $T_k = T_{k-1}/(1+\beta T_{k-1})$, $\beta>0$. Connoly(1990) da β nın değeri sabit kabul edilmekte ve $\beta = (T_0-T_M)/((M-1)T_0T_M)$ ile belirlenmektedir. Van Laavarhoven vd. (1992) de ise β nın değeri değişkendir ve k yıncı devrede $\beta = \ln(1+ \delta)3\sigma_k$ ile verilmektedir. Burada σ_k k yıncı devrede oluşturulan komşu çözümlerin amaç değerlerinin standart sapması ve δ uzaklık parametresidir.

C. Devre Uzunluğu (L)

Tavlama sürecinde her bir sıcaklıkta çok sayıda komşu çözüm oluşturulmakta ve bunlardan bazıları yeni çözüm olarak kabul edilmektedir. Mevcut çözümün kendisine komşu bir çözümle yer değiştirmesi geçiş olarak adlandırılmaktadır. Herhangi bir sıcaklıkta geçişlere karar vermek için yapılacak ikili karşılaştırmaların sayısı sınırlandırılmalıdır. Aksi halde tavlama süreci sonsuz bir döngüye dönüşebilecektir. Söz konusu sınırlandırma devre uzunluğu ile yapılmaktadır. Daha açık ifade ile devre uzunluğu (L), her bir sıcaklıkta izin verilen en büyük deneme sayısıdır. L nin değeri problem hacminin bir oranı olarak belirlenebilir. Bir başka yol, L yi belli bir çözümün komşu çözümlerinin sayısının bir oranı olarak belirlemektir. Oluşturulabilecek tüm komşu çözümlerin sayısının önceden biliniyor olması durumunda L toplam deneme sayısının bir oranı olarak da belirlenebilir.

D. Durdurma Ölçütü

TB algoritması, devre sayacı önceden belirlenmiş bir değere ulaştığında veya toplam deneme sayısı önceden belirlenmiş bir değere ulaştığında sona erdirilebilir. Algoritmayı durdurmak için diğer bir kural hesaplama süresini sınırlamaktır. Van Laarhoven vd. (1992), sıcaklık başlangıçta seçilmiş olan bir

son sıcaklığın (T_{min}) altına düştüğünde algoritmanın durdurulmasını önermektedir. Bir başka etkin durdurma kuralı Johnson vd. (1989) tarafından geliştirilmiştir. Bu kural için farklı bir sayaca gerek vardır. Her devrenin sonunda kabul edilmiş olan çözüm sayısının oluşturulan toplam çözüm sayısına oranı hesaplanmakta ve bu oranın belli bir limit değerin altında olması durumunda sayaç 1 artırılmaktadır. En iyi çözümden daha iyi bir çözüm bulunması durumunda ise sayaç sıfırlanmaktadır. Bu yöntemde sayacın değeri belli bir değere ulaştığında algoritma sonlandırılmaktadır.

İyi bir TB geliştirmek için yukarıda özetlenen yöntemlere ilaveten Park ve Kim (1998) ve Wang ve Wu (1999b) parametre değerlerini ayarlamak için birer sistematik prosedür geliştirmişlerdir. Ancak literatür incelendiğinde TB kullanan çoğu çalışmanın daha önce benzer çalışmalarda kullanılmış olan parametre değerlerini benimsemiş oldukları görülmektedir. Böyle yaparak çözüm süresi ve hesaplama çabası önemli miktarda azaltılabilmektedir.

V. Önerilen Yaklaşım

Bu çalışmada KÇPP probleminin çözülmesi için önerilen yöntem bir TB nin kullanılmasıdır. En iyileme alanında TB kullanıldığında ortaya çıkan algoritmik yapının görünüşü daima birbirine benzer özellikler sergilemektedir. Ancak burada benimsenen yaklaşımdaki farklılıklar, başlangıç ve komşu çözümlerin oluşturulmasında, sıcaklığın nasıl ve ne zaman azaltılacağında ve durdurma ölçütlerinde ortaya çıkmaktadır.

A. Çözüm Algoritması

Adım 1: $T_k = T_0$, TS = 0, KS = 0, MS = 0, DS = 0
Adım 2: Bölüm IV (B) deki algoritma ile bir başlangıç çözümü oluştur, S
Adım 3: Eşitlik (1) ile S nin amaç değerini belirle, C(S)
Adım 4: X = S, C(X) = C(S), M = S, C(M) = C(S)
Adım 5: Eğer DS = DS_{max} ise veya $T_k = T_{min}$ ise adım 16 ya, aksi halde adım 6 ya git.
Adım 6: Bölüm IV (C) deki algoritma ile komşu çözüm oluştur, S', TS = TS+1
Adım 7:Komşunun amaç değerini hesapla, $C(S')$
Adım 8: $\Delta_1 = C(S') - C(X)$
Adım 9: Eğer $\Delta_1 \geq 0$ ise adım 12 ye git. Aksi halde $X = S'$, $C(X) = C(S')$ ve KS = KS+1
Adım 10: $\Delta_2 = C(S') - C(M)$
Adım 11: Eğer $\Delta_2 \geq 0$ ise adım 13 e git. Aksi halde $M = S'$, $C(M) = C(S')$, DS = 0, MS = MS+1 ve adım 13 e git.
Adım 12: Düzgün dağılımdan $[0,1]$ aralığında bir rasgele sayı çek, RS. Eğer $RS \geq e^{\frac{-\Delta_1}{T_k}}$ ise adım 13 e git. Aksi halde $X = S'$, $C(X) = C(S')$ ve KS = KS+1
Adım 13: Eğer TS > L ise adım 14 e git. Aksi halde adım 6 ya git.
Adım 14: Eğer MS = 0 ise $T_{k+1} = \alpha T_k$ ve adım 15 e git. Aksi halde adım 15 e git.
Adım 15: Eğer KS/TS < O_{min} ise DS = DS+1, TS = 0, KS = 0, MS = 0 ve adım 5 e git. Aksi halde TS = 0, KS = 0, MS = 0 ve adım 5 e git
Adım 16: Algoritma donmuştur. En iyi çözüm M ve en düşük toplam maliyet C(M) dir.

Yukarıdaki TB algoritmasında, soğutma her devrenin sonunda değil, yalnız en iyi çözümden daha iyisinin bulunamadığı devrelerin sonunda yapılmaktadır. Sayaç MS ile kontrol edilen bu uygulama sıcaklığın daha yavaş azaltılmasına ve aramanın iyi çözümlerin etrafında yoğunlaştırılmasına yol açmaktadır Bir başka dikkate değer nokta ise iki ayrı durdurma ölçütünün kullanılmasıdır. Söz konusu ölçütlerden biri, izin verilebilir bir en düşük sıcaklığa (T_{min}) ulaşılması durumudur. Sıcaklık T_{min} değerine düştüğünde kötü çözümlerin kabul olasılığı çok küçülmüş olacaktır. TB nin çıkış noktası zaten bu özelliğe dayanmaktadır. İkinci durdurma ölçütü, donma sayacının önceden belirlenmiş bir en büyük değere (DS_{max}) ulaşması durumudur. DS bir devre içinde kabul edilmiş çözüm yüzdesinin belli bir değerin altında kalması ile artmaktadır. KS VE TS sayaçları ile kontrol edilen DS nin yüksek bir değeri, algoritmanın artık sürekli kötü çözümler üretmeye başladığının ve donmaya doğru yöneldiğinin bir göstergesidir. DS değeri yüksek iken en iyi çözümden daha iyi bir çözümün bulunması, donma ihtimalinin o an için ortadan kalmış olması anlamına geleceği için bu durumda DS sıfırlanmaktadır.

B. Başlangıç Çözümünün Oluşturulması

Bu bölümde öncelik ve kaynak kısıtlarına tamamen uygun bir rasgele başlangıç çözümü oluşturmak için bir algoritma geliştirilmektedir. Algoritmanın adımları:

Adım 1: Zaman sayacını başlat, $t = 0$

Adım 2: Eldeki kaynak miktarlarını belirle, $R_{kt} = R_k$

Adım 3: Programlanacak faaliyetler listesini kaydet, *PR*

Adım 4: Öncelik kısıtı açısından başlatılabilecek faaliyetleri belirle,

$$\ddot{O}KB = \left\{ i, j \in PR \middle| P(i, j) = \phi \right\}$$

Adım 5: Kaynak kısıtları açısından başlatılabilecek faaliyetleri belirle

$$KKB = \left\{ i, j \in PR \middle| \forall k \text{ için } r_{ijk} \le R_{kt} \right\}$$

Adım 6: Başlatılabilecek faaliyetleri belirle $B = \ddot{O}KB \cap KKB$

Adım 7: B kümesinden düzgün dağılımlı rasgele sayı kullanarak bir faaliyet seç ve başlat, *BF= herhangi bir i,j∈ B.* BF nin başlama zamanını kaydet, $S_{ij} = t$

Adım 8: BF yi ÖKB ve KKB kümelerinden sil, *ÖKB = ÖKB/BF ve KKB = KKB/BF*

Adım 9: Başlatılabilecek durumda iken başlatılmamış olan faaliyetleri komşu çözüm oluşturmada kullanılmak üzere kaydet, *BMF = B/BF*

Adım 10: BF nin bitiş zamanını belirle, $F_{ij} = S_{ij} + t_{ij}$

Adım 11: Eldeki kaynak miktarlarını belirle, $R_{kt} = R_k - r_{BF,k}$ $\forall k$ için

Adım 12: Eğer $BMF \neq \phi$ ise adım 5 e git. Aksi halde bulunulan zaman ile en küçük bitiş zamanı arasında herhangi bir kaynak girişi olup olmayacağını kontrol etmek üzere adım 13 e devam et.

Adım 13: Eğer herhangi bir t_k için $\min F_{ij} - t \ge \min F_{ij}\ (\text{Mod}(t_k))$ ise $t = \min F_{ij} - \min F_{ij}\ (\text{Mod}(t_k))$. k tipi kaynak için elde kalan miktarı hesapla, $R_{kt} = R_{kt} + R_k$. Adım 5 e git. Aksi halde adım 14 e devam et.

Adım 14: t = min F_{ij}., i,j faaliyetini PR den sil ve adım 4 e git.

Adım 15: $PR = \phi$ ise başlangıç çözümü elde edilmiştir. Projelerin bitiş zamanlarını belirle, CD_i = max F_{ij} (j = 1,2,…n_i), DUR.

C. Komşu Çözümün Oluşturulması

Yukarıdaki algoritma ile bir çözüm oluşturulduğunda zamana (t) göre küçükten büyüğe doğru sıralanmış bir liste elde edilecektir. Mevcut çözümden bir komşu çözüm oluşturmak için bu çalışmada, başlatılmamış faaliyetler kümesi BMF nin kullanılması önerilmektedir. Bütün kısıtlara uygun komşular oluşturan algoritmanın adımları aşağıda özetlenmektedir.

Adım 1: Eğer $BMF = \phi$ ise alternatifsiz çözüm durumu vardır, komşu çözüm oluşturulamaz, programlama sona ermiştir, DUR.

Adım 2: t_{min} = BMF kümesindeki ilk zamanı ve t_{max} = BMF kümesindeki en son zamanı göstermek üzere, eğer $t_{max} < t_{min}$ ise komşu çözüm oluşmuştur, DUR. Aksi halde adım 3 e devam et.

Adım 3: Eğer t_{max} zamanına ilişkin faaliyetler varsa, bunlar arasından düzgün dağılımlı rasgele sayı kullanarak bir faaliyet (i,j) seç, (i,j) faaliyetinin başlama zamanını (S_{ij}) al ve adım 4 e git. Aksi halde t_{max} ı BMF kümesinden sil ve adım 2 ye git.

Adım 4: Eğer $S_{ij} = t_{max}$ ise (i,j) faaliyetini BMF deki t_{max} a ilişkin faaliyetler arasından sil ve adım 3 e git. Aksi halde adım 5 e devam et.

Adım 5: (i,j) faaliyetini t_{max} zamanında başlat, $S_{ij} = t_{max}$

Adım 6: t_{max} zamanından sonrasını programlamak üzere başlangıç çözümü oluşturma algoritmasının sekizinci adımına git.

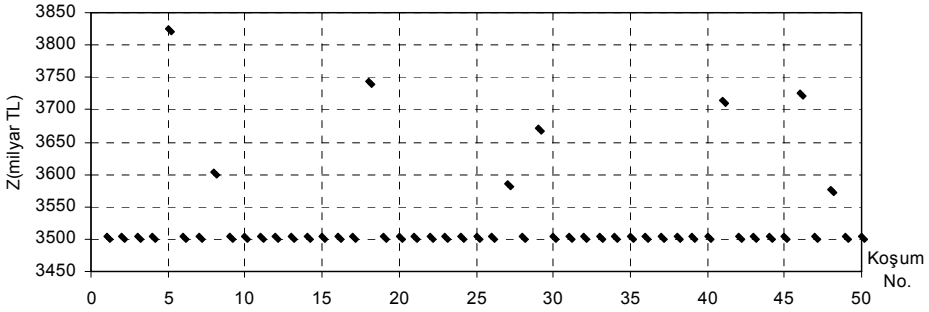
VI. Uygulama

Sezgisel çözüm algoritmalarının etkinlik testleri, literatürden standart problemler üzerinden yapılabilmektedir. Ancak literatürde en iyi çözüm değeri bilinen bir KÇPP problemine rastlanamamıştır. Ayrıca bu çalışmada ele alınan problemin ilave bir özelliği kaynakların yenilenme zamanlarının kesikli bir yapıya sahip olmasıdır. Bu nedenle Mendes (2003) de geliştirilmiş olan problem üretecinde bazı zorunlu değişiklerle varsayımsal bir problem geliştirilmiştir. Problem üretecine göre her bir proje Kolish ve. Sprecher (1996) da verilen J30 tipindeki, 488 tek proje arasından rasgele seçilmiştir . Her bir proje için faaliyet öncelikleri, faaliyet süreleri, kaynak gereksinimleri ve ideal tamamlanma süreleri ile kaynak kapasiteleri PSPLIB kütüphanesinde mevcuttur . Kaynakların sabit yenilenme periyotları ise keyfi olarak, ideal tamamlanma süreleri ortalamasının %25 i olarak alınmıştır.

Tablo 1: *Deneme Problemi Çözüm Sonuçları*

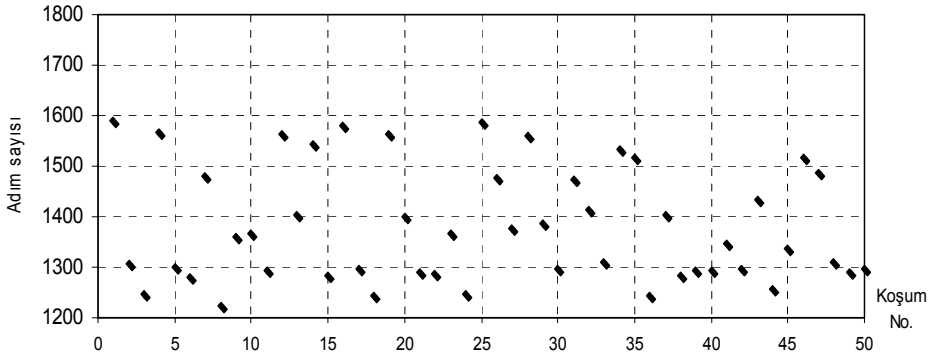
T_0	T_{min}	α	L	DS_{max}	Z_{min}	Ort. hesaplama süresi(sn)
700	2,37	0,85	5	5	4152	608,3
475	271	0,98	4	5	4774	50,9
300	0,04	0,80	7	5	4164	1210,9
250	41,52	0,95	6	5	4903	873,4
200	61,98	0,97	7	5	4918	142,8
180	43,12	0,96	5	5	3621	1084,6
150	3,75	0,90	6	5	3640	1098,5
100	4,24	0,90	5	5	3504	124,3
50	2,12	0,90	4	5	4881	349,8
20	0,1	0,90	5	5	4087	619,4

Problemin maliyet fonksiyonu katsayıları varsayımsal olmakla birlikte uzman kişilere danışılarak belirlenmiştir. Özetlenen üreteç ile oluşturulan her biri 30 faaliyetten oluşan 20 projeli problem, C++ da kodlanan TB algoritması ile Pentium M, 1.60 GHz hızında 752 MB hafızaya sahip bilgisayarda çözülmüş ve problemin başlangıç çözümünde amaç değeri 6436 olarak bulunmuştur. Farklı parametre değerleri ile 10 ar kez koşturulan algoritmanın sonuçları Tablo 1 de sunulmaktadır. Deneme koşumunda elde edilen en iyi sonuca ilişkin parametre değerleri ile algoritma 50 kez koşturularak bulunan en iyi çözümlerin frekansları gözlemlenmiştir. Aşağıda uygulama sürecine dair bazı grafikler sunulmaktadır.



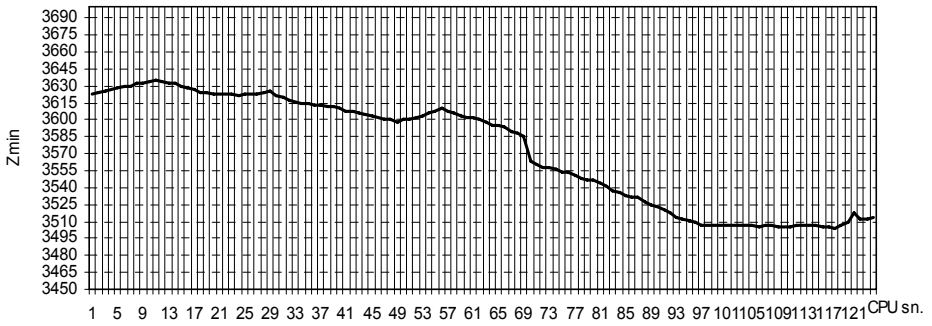
Şekil 2: *Koşum No.-Optimum Çözüm Dağılımı*

Şekil 2 birbirlerinden bağımsız olarak her bir koşumda elde edilen en iyi çözüm değerinin dağılımını göstermektedir. Sonuçların çok büyük bir kısmının 3500 civarında yoğunlaşmış oldukları görülmektedir. Bu durum algoritmanın en iyi sonucu elde etmekteki kararlılığını göstermesi açısından son derece anlamlıdır.



Şekil 3: *Koşum No.-Toplam Adım Sayısı Dağılımı*

Şekil 3 de koşumların kaçar adımda sona ermiş oldukları gösterilmektedir. Doğal olarak bazı koşumlar en iyi sonuca ulaşamadan sona ermişlerdir. Koşumların adım sayıları incelendiğinde 1225 ile 1589 arasında değiştikleri ve ortalamalarının 1382 olduğu görülmektedir.



Şekil 4: *Örnek Koşum İçin Amacın Zamana Göre Değişim Grafiği*

Uygulama sürecinde her bir koşum, birbirinden bağımsız olmasına rağmen defalarca tekrarlanmaktadır. Bunun nedeni, algoritmanın en iyi çözümü bulmaktaki kararlılığını test etmektir. Bundan dolayı bütün koşumları bir tek grafik üzerinde izlemek hem mümkün değildir hem de anlamlı değildir. Ancak koşumların nasıl bir seyir izlediklerini görebilmek amacı ile Şekil 4 de bir örnek olarak 1450 adım ve 125 sn. de sona ermiş olan 38. koşumda amaç değerinin zamana göre değişim grafiği sunulmaktadır.

Tablo 2: *Küçük Problemdeki Projelere Ait Faaliyet Verileri*

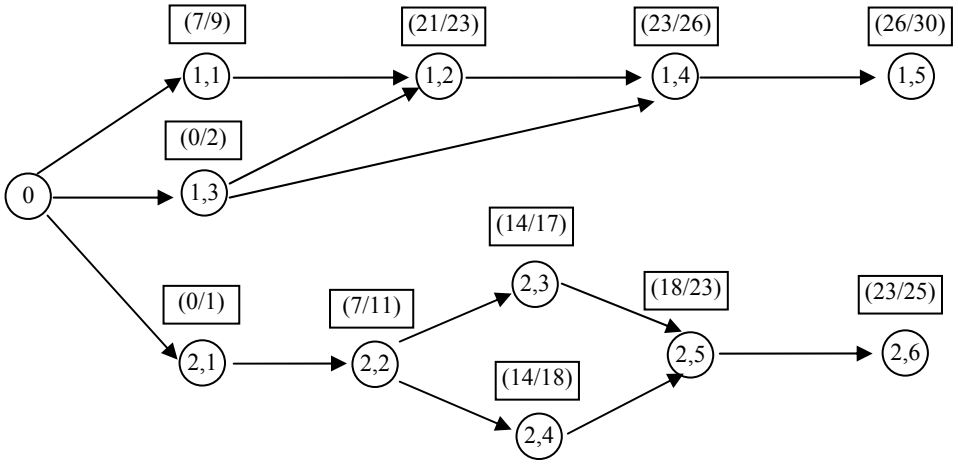
Faaliyet	Önceki Faaliyet	Süre	Kaynak gereksinimi
(1,1)	-	2	5
(1,2)	(1,1), (1,3)	2	2
(1,3)	-	2	4
(1,4)	(1,2), (1,3)	3	3
(1,5)	(1,4)	4	2
(2,1)	-	1	4
(2,2)	(2,1)	4	3
(2,3)	(2,2)	3	3
(2,4)	(2,2)	4	4
(2,5)	(2,3), (2,4)	5	1
(2,6)	(2,5)	2	2

Grafikten görülebileceği gibi koşum sırasında zaman sürekli evrimleşerek değil iyi ve kötü çözümlerle ilerlemekte ve belli bir noktadan sonra sürekli kötü çözümler üretmeye başlamaktadır. İşte bu nokta algoritmanın donma noktasıdır.

Tablo 3: *Başlangıç Çözümünün Oluşturulması*

t	R_{kt}	ÖKB	KKB	B	BF	BMF	F_{ij}
0	8	(1,1),(1,3),(2,1)	(1,1),(1,2),(1,3),(1,4),(1,5),(2,1),(2,2),(2,3),(2,4),(2,5),(2,6)	(1,1),(1,3),(2,1)	(2,1)	(1,1),(1,3)	1
0	4	(1,1),(1,3)	(1,2),(1,3),(1,4),(1,5),(2,2),(2,3),(2,4),(2,5),(2,6)	(1,3)	(1,3)	-	2
1	0	(1,1),(2,2)	-	-	-	-	-
2	0	(1,1),(2,2)	-	-	-	-	-
7	8	(1,1),(2,2)	(1,1),(1,2),(1,4),(1,5),(2,2),(2,3),(2,4),(2,5),(2,6)	(2,2)	(2,2)	1	11
7	5	(1,1)	(1,1),(1,2),(1,4),(1,5),(2,3),(2,4),(2,5),(2,6)	(1,1)	(1,1)	-	9
9	0	(1,2)	-	-	-	-	-
11	0	(1,2),(2,3),(2,4)	-	-	-	-	-
14	8	(1,2),(2,3),(2,4)	(1,2),(1,4),(1,5),(2,3),(2,4),(2,5),(2,6)	(2,3)	(2,3)	(1,2),(2,4)	17
14	5	(1,2),(2,4)	(1,2),(1,4),(1,5),(2,4),(2,5),(2,6)	(2,4)	(2,4)	(1,2)	18
17	1	(1,2)	-	-	-	-	-
18	1	(1,2),(2,5)	(2,5)	(2,5)	(2,5)	-	23
21	9	(1,2)	(1,2)	(1,2)	(1,2)	-	23
23	7	(1,4),(2,6)	(1,4),(1,5),(2,6)	(2,6)	(2,6)	(1,4)	25
23	5	(1,4)	(1,4),(1,5)	(1,4)	(1,4)	-	26
25	5	-	(1,5)	-	-	-	-
26	5	(1,5)	(1,5)	(1,5)	(1,5)	-	30
30	3	-	-	-	-	-	-

Yer problemi nedeni ile 30*20 boyutundaki çok büyük bir problemi bu çalışma kapsamında göstermek mümkün değildir. Bu nedenle Tablo 2 de ilgili verileri sunulan ve teslimat tarihleri (DD_i) belirli 2 proje ile tek kaynaktan oluşan küçük bir KÇPP problemi için başlangıç ve komşu çözüm oluşturma aşamaları maliyet hesaplamaları ile birlikte aşağıda açıklanmaktadır. Tablo 2 de parantezler içindeki ilk rakamlar proje numarasını ikinci rakamlar ise faaliyet numarasını göstermektedir. Başlangıç çözümü oluşturma aşamalarını özetleyen Tablo 3 deki sütun başlıklarına ait kısaltmaların anlamları bölüm 4 (B) de sunulan algoritmada açıklanmış olduğu gibidir. Öncelik ilişkilerine göre proje akışları ve başlangıç çözümünde oluşturulmuş olan faaliyet programı Şekil 5 de sunulmaktadır. Şekilde yuvarlaklar içindeki rakamlar proje ve faaliyet numaralarını gösterirken dörtgenler içindeki ilk rakam ait olduğu faaliyetin başlangıç, ikinci rakam ise bitiş zamanını ifade etmektedir. Mevcut ve en iyi çözüm olarak kabul edilecek olan başlangıç çözümünün maliyeti, eşitlik (1) den $Z = a_1\max(30\text{-}DD_1,0)+a_2\max(25\text{-}DD_2,0)+7b_{11}+12b_{12}+6b_{22}+3b_{23}+3b_{24}$ şeklinde hesaplanmaktadır. Bir başlangıç çözümünün çok sayıda komşusu vardır. Bulunabilecek komşuların sayısı bazı durumlarda kombinasyon formülleri ile hesaplanabilir. Ancak kaynak kısıtlarının varlığı durumunda kısıtlara uygun komşu sayısını önceden belirlemek artık mümkün değildir. Sayısı belirsiz olan komşulardan birinin oluşturulma adımları Tablo 4 de sunulmaktadır. T = 14 den önceki program Tablo 3 ün aynısıdır.



Şekil 5: *Başlangıç Çözümüne Dair Faaliyet*

Komşu çözüm $Z = a_1 \max(25\text{-}DD_1, 0)+a_2 \max(30\text{-}DD_2, 0)+7b_{11}+5b_{12}+2b_{15}+6b_{22}+3b_{23}+10b_{24}$ olarak hesaplanan maliyetinin mevcut çözümün maliyetinden küçük olması durumunda kabul edilmekte aksi halde $e^{-\Delta_1/T_k}$

olasılığına göre kabul veya reddedilmektedir. Bu nedenle en iyi sonucu bulmak için hangi yoldan ilerleneceği rasgele belirlenmiş olmaktadır.

Tablo 4: *Komşu Çözümün Oluşturulması*

t	R_{kt}	ÖKB	KKB	B	BF	BMF	F_{ij}
14	5	(1,2),(2,4)	(1,2),(1,4),(1,5),(2,4),(2,5),(2,6)	(1,2),(2,4)	(1,2)	(2,4)	16
16	3	(1,4),(2,4)	(1,4),(1,5),(2,5),(2,6)	(1,4)	(1,4)	-	19
17	0	(2,4)	-	-	-	-	-
19	0	(1,5),(2,4)	-	-	-	-	-
21	8	(1,5),(2,4)	(1,5),(2,4),(2,5),(2,6)	(1,5),(2,4)	(2,4)	(1,5)	25
21	4	(1,5),(2,5)	(1,5),(2,5),(2,6)	(1,5),(2,5)	(1,5)	(2,5)	25
25	2	(2,5)	(2,5),(2,6)	(2,5)	(2,5)	-	30
28	10	-	(2,6)	-	-	-	-
30	8	(2,6)	(2,6)	(2,6)	(2,6)	-	32
32	8	-	-	-	-	-	-

VII. Sonuç ve Öneriler

Bu çalışmada kaynak kısıtlı çoklu proje programlama problemleri (KÇPP) için bir tavlama benzetimi algoritması sunulmuştur. Geliştirilen algoritmanın etkinliği 30 ar faaliyetli 20 projeden oluşan bir problem üzerinde test edilmiştir. Algoritma en uygun parametre değerlerinin belirlenmesi için önce farklı parametre değerleri ile 10 ar kez koşturulmuştur. Deneme koşumlarında en iyi amaç değeri (Z_{min} = 3504) 124.3 sn. ortalama hesaplama süresi içinde elde edilmiştir. Daha sonra en iyi sonucu vermiş olan parametre değerleri ile algoritma 50 kez koşturulmuş ve 42 kez (koşumların %84 ünde) aynı amaç değerine ulaşılmıştır. Bu sonuçlara göre problemin en iyi çözüm sonucunun %84 olasılıkla 3504 olduğu söylenebilir. 42 koşumdaki ortalama hesaplama süresi 117 sn. dir. Bu süre daha önce yapılmış çalışmalarda aynı boyuttaki benzer problemlere dair sürelerle karşılaştırıldığında en az onlar kadar iyi olduğu görülmektedir.

Elde edilen bir başka bulgu, algoritmanın performansının büyük ölçüde parametre seçimine bağlı olmasıdır. Tablo 1 de deneme sonuçlarında görülebileceği gibi bazı parametre değerleri algoritmayı daha kısa sürede ancak daha kötü çözümlerle sonlandırabilmektedir. Burada amaç, kabul edilebilir bir süre içinde en iyi çözüme ulaşmaktır. Bu nedenle parametre değerlerinin belirlenmesinde son derece dikkatli olunmalıdır.

Gelecek araştırma konularından biri, burada önerilen algoritmanın diğer meta-sezgisel yöntemler kullanılarak performans karşılaştırmasının yapılması, bir diğeri ise farklı amaç fonksiyonlarına ve/veya farklı kaynak yenileme özelliklerine sahip olan problemlere uyarlanması olabilir.

Kaynaklar

Aarts, E.H.L and Korst, J.H.M. (1989), Simulated Annealing and Boltzmann Machines: A stochastic Approach to Combinatorial Optimization and Neural Computing, Wiley, Chichester,.ss. 284.

Ash, R. (1989) "Activity Scheduling in the Dynamic, Multiple-Project setting: Choosing Heuristics Through Deterministic Simulation", Proceedings of the 1999 Winter Simulation Conference, ss. 937–941.

Blazewicz, J., Lenstra, J.K, Kan, A., Rinnooy, H.G. (1983). "Scheduling Subject to Resource Constraints: Classification and Complexity ", Discrete Applied Mathematics, 5, ss. 11-24.

Bock, D.B. and Patterson, J.H. (1990), "A Comparison of Due Date Setting, Resource Assignment and Job Preemption Heuristics for the Multiproject Scheduling Problem", Decision Sciences, 21, ss. 387–402.

Boctor, F.F. (1990) "Some Efficent Multi-Heuristic Procedures for Resource Constrained Project Scheduling", European Journal of Operational Research, 49, ss. 3–13.

Bouleimen, K. and Lecocq, H. (2003) "A New Efficent Simulated Annealing Algorithm for the Resource Constrained Project Scheduling Problem and Its Multiple Mode Version", European Journal of Operational Research, 149, ss. 268–281.

Cerny, V. (1985) "Thermodynamical Approach to the Traveling Saesman Problem: An Efficent Simulation Algorithm", Journal of Optimization Theory and Applications, 45, ss. 41–5.

Chiu, H.N.and Tsai, D.M. (2002) "An Efficent Search Procedure for the Resource Constrained Multi-Project Scheduling Problem with Discounted Cash Flows", Construction Management and Economics, 20, ss. 55–66.

Connoly, D. (1990) "An Improved Annealing Scheme for the QAP", European Journal of Operational Research, 46, ss. 93–100.

Dantzig, G.B, Wolfe, P. Decomposition principle for linear programs, Operations Research, 8, ss. 101–111.

Deckro, R.F., Winkofsky , E.P., Hebert , J.E., Gagnon , R. (1991) "A Decomposition Approach to Multi-project Scheduling", European Journal of Operational Research, 51, ss. 110–118.

Drexl, A. (1991) "Scheduling of Project Networks by Job Assignment", Management Science, 37(12), ss. 1590–1602.

Dumond, J.and Mabert, V.A. (1988) "Evaluating Project Scheduling and Due Date Assignment Procedures: An Experimental Analysis", Management Science, 34(1), ss. 101–118.

Fendley, L.G. (1968) "Towards the Development of A Complete Multiproject Scheduling System", Journal of Industrial Engineering,October, ss. 505–515.

Gonçalves, J.F., Mendes, J.J.M., Resende, M.G.C. (2004) "A Genetic Algorithm for the Resource Constrained Multi-Project Scheduling Problem", AT&T Labs Technical Report, ss.1-19.

Homberger, J. (2007) "A Multi Agent System for the Decentralized Resource-Constrained Multi-Project Scheduling Problem", Int. Transactions in Operational Research, 14 (6), ss. 565–589.

Huang, M., Romeo, F., Sangiovanni-Vincentelli, A. (1986) "An Efficent General Cooling Scedule for Simulated Annealing", IEEE Transact. on Computer Aided Design, 5 (1), ss. 381–384.

Johnson, D.S,.Aragon, C.R., McGeoch, L.A, Schevon, C.(1989) "Optimization by Simulated Annealing: An Experimental Evaluation; Part 1, Graph Partitioning", Operations Research, 37, ss 865-892.

Jozefowska, J., Mika, M., Rozycki, R., Waligora, G., Weglarz, J. (2001) "Simulated Annealing for Multi-Mode Resource-Constrained Project Scheduling", Annals of Operations Research, 102,ss. 137–155.

Kim, K.W., Yun, Y., Yoon, J., Gen, M., Yamazaki, G. (2005) "Hybrid Genetic Algorithm with Adaptive Abilities for Resource-Consrained Multiple Project Scheduling", Computers in Industry, 56 (2), ss. 143–160.

Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P. (1983) "Optimization by Simulated Annealing", Science, 220, ss. 671–680.

Kolish, R.and Hartmann, S. (1998) "Heuristic Algorithms for the Resource-constrained Project Scheduling Problem: Classification and Computational analysis", J. Weglarz, ed., Kluwer Academic, Amsterdam, ss. 147–178.

Kolish R. and Sprecher, A. (1996) "PSLIB-a Project Scheduling Problem Library", European Journal of Operational Research, 96, ss. 205–216.

Kurtuluş, I.S. and Davis, E.W. (1982) "Multi-project Scheduling: Categorization of Heuristic Rules Performances", Management Science, 28 (2), ss. 161–172.

Kurtuluş, I.S. and Narula, S.C. (1985) "Multi-project Scheduling: Analysis of Project Performance", IIE Transactions, 17 (1), ss 58–66.

Lawrence, S.R.and Morton, T.E. (1993) "Resource-Constrained Multi-project Scheduling with Tardy Costs: Comparing Myopic, Bottleneck and Resource Pricing Heuristics", European Journal of Operational Research, 64, ss. 168–187.

Linyi, D. and Yan, L. (2007) "A Particle Swarm Optimization or Resource-Constrained Multi-Project Scheduling Problem", International Computational Intelligence and Security Conference, 15(9), ss. 1010–1014.

Lova, A., Maroto, C., Tormos, P. (2000) "A Multicriteria Heuristic Method to Improve Resource Allocation in Multiproject Scheduling", European Journal of Operational Research, 127, ss. 40–424.

Lundy, M.and Mees, A. (1986) "Convergence of Annealing Algorithm", Mathematical Programming, 34, ss. 111–124.

Mendes, JJM. (2003) "Sistema de apoio â decisão para planeamento de sistemas de produção do tipo projecto" PhD thesis, Departamento de Engenharia Mecânica e Gestão Industrial, Faculdade de Engenharia da Universidade do Porto, Portugal.

Metropolis, N., Rosembluth, A., Rosenbluth, M., Teller, A. (1953) "Equation of State Calculationsby Fast Computing Machines", Jornal of Chemical Physics, 21, ss. 1087–1092.

Mohanty, R.P.and Siddiq, M.K. (1989) "Multiple Projects Multiple Resources-Constrained Scheduling", International Journal of Production Research, 27 (2), ss. 261–280.

Okada, I., Lin, L., Gen, M. (2008) "Solving Reource Constrained Multiple Project Scheduling Problems by Random Key-Based Genetic Algorithm", IEEJEISS, 128 (3), ss. 441–449.

Park, M.W., and Kim, Y.D.(1998) "A Systematic Procedure for Setting Paameters in Simulated Annealing Algorithms", Computers and Operations Research, 3, ss. 207–217.

Potts, C.N.and Van Wassenhove, L.N. (1991) "Single Machine Tardiness Sequencing Heuristics", IIE Transactions, 23, ss. 346–354.

Pritsker, A.B., Wattres, L.J., Wolfe, P.M. (1969) "Multi-Project Scheduling with Limited Resources: A Zero-One Programming Approach", Management Sciene, 16 (1), ss. 93–109.

Shankar, V. and Nagi, R. (1996) "A Flexible Optimization Approach to Multi-Resource, Multi-Project Planning and Scheduling", Proceedings of 5th Industrial Engineering Research Conference, Minneapolis, May, USA.

Tsubakitani, S. and Decro, R.F. (1990) "A Heuristic for Multi-Project Scheduling with Limited Resources in Housing Industry", European Journal of Operational Research, 49, ss. 80–91.

Van Laarhoven, P.J.M.and Aarts, E.H.L. (1987) Simulated Annealing: Theory and Applications, Kluwer Academic Publisher,.ss. 204.

Van Laarhoven, P.J.M,. Aarts, E.H.L., Lenstra, J.K. (1992) "Job Shop Scheduling by Simulated Annealing", Operations Research, 40, ss. 113–126.

Vercellis, C. (1994) "Constrained Multi-Project Planning Problems: A Lagrangean Decomposition Approach", European Journal of Operational Research, 78, ss. 267–275.

Wang, T.Y.and Wu, K.B. (1999) "A Parameter Set Design Procedure for the Simulated Annealing Algorihm under the Computational Time Constrained", Computers and Operations Research, 26, ss. 665–678.

Wiley, V.D., Deckro, R.F., Jackson, J.A. (1998) "Optimization Analysis for Design and Planning of Multi-Project Programs", European Journal of Operational Research, 107, ss. 492–506.