$n/1/L_{enb}$ veya T_{enb} İŞ SIRALAMA PROBLEMLERİNİN ÇÖZÜMÜNDE SEZGİSEL BİR YÖNTEM : (En Kısa Teslim)

Doç.Dr. Mehmet ÇINAR*

TERMİNOLOJİ

Çizelgeleme kavramı her bir alanda kullanılmasına karşılık, terminolojik temeli endüstrisinden gelmekte olup üretim yönetiminin önemli bir karar alanını oluşturmaktadır. Çizelgelemeye ilişkin temel terimler izleyen biçimde tanımlanmaktadır [Çınar, 1990]:

Toplam işlem : Bir çok işin veya parçanın işlenmesi ile ilgili olup izleyen sorunları içerir :

- İşler nasıl işlenmelidir?
- Hangi işler işlenmelidir?

İş atölyesi : Bir işin atölyesi, işlerin üzerinde işlenebileceği tezgahlar kümesidir. İşler, düzgün (doğrusal) bir akışı izleyerek tamamlanabileceği gibi, değişik güzergahları da izleyebilirler. Ürün çeşidi ve tezgah sayısı arttıkça, düzgün bir iş akışına nadir olarak rastlanabilir.

Akış atölyesi : İşlerin düzgün bir akışı izlediği, belirli bir işlem temelli birimlerdir. Her bir işlem üç temel eleman ile belirlenir.

- İş,
- Tezgah veya Makina, (burada makina, her zaman fiziksel bir tezgah anlamında olmayıp, zaman ölçüsünde elde bulunan bir aralık ta olabilir),
- İşleme süresi.

Çizelgeleme : İşlerin, fabrika veya atölyedeki üretim akışında süreçlenmesi için izlenmesi gerekli sıradır. Çizelgeleme ve sıralama kavramları eş anlamda kullanılmalarına karşılık, sıralama daha dar anlam taşımaktadır. İşlerin belirlenen öncelik değerlerine göre sıraya sokulması çalışmalarına **iş sıralaması** adı verilir. Böylelikle, bir tezgah boşaldığı zaman tezgaha yüklenecek iş bu sıra temelinde yapılır. İş çizelgelemesinde ise işlerin bir tezgahta hangi zamanlarda başlaması ve bitirilmesi gerektiği planlanır. Diğer bir anlatımla elde edilen iş sıralamasına zaman boyutunun da eklenmesiyle iş çizelgesi hazırlanmış

* Erciyes Üniversitesi, İktisadi ve İdari Bilimler Fakültesi, Üretim Yönetimi A.B.D. Öğretim Üyesi.

olur. Genel çizelgeleme problemleri çok karmaşık olup, yapıları temelinde özel türleri ve bunlara yönelik çözüm yöntemleri bulunmaktadır. 10 tezgahlı bir atölyede, bu tezgahlardan geçecek 100 işlem varsa, toplam $(100!)^{10}$ kadar çizelgeleme seçeneği bulunmaktadır. Sadece 8 işi bir tezgahta sıralamak bile 8! = 40320 seçenek ortaya çıkarmaktadır. Kaldı ki, iş parçalarının tezgahlara atanması ile ilgili kısıtlar, iş öncelikleri, teslim süreleri gibi ölçütlerin de dikkate alınması ile problem yapısı iyice karmaşıklaşır, ancak seçenek sayısı da azalır.

Statik sıralama problemlerinde atölyeye gelen işler önceden bilinir ve işler atölyeye belirli bir düzen içinde gelirler. Problemin bu statik özelliği cebirsel birtakım kurallarla çözüme ulaşmayı sağlamaktır. Ancak iş listesinin sürekli ve rasgele değiştiği dinamik (değişken) sıralama problemlerinde çözüme ulaşmak için tamamen farklı teknikler kullanmak gerekir. Bu tekniklerin temeli, öncelik kurallarının kullanılmasına dayanır. Öncelik kuralları, işlerin özellikleri ya da sistemin durumuna göre işleri öncelik değerleri vermede kullanılır. Statik öncelik kuralları, üretim sürecinde işlerin değişmeyen özelliklerini kullanan kurallardır. Örneğin, bir parçanın üretimi için gerekli işlem sayısı gibi. Dinamik öncelik kuralları ise üretim süreci içinde değişen durumlara göre öncelik hesaplamada kullanılan kurallardır. Örneğin, parçanın kalan işlem sürüsi ya da önceden belirlenmiş olan teslim tarihleri gibi.

KULLANILACAK YAZILIM VE DEĞERLEME ÖLÇÜTLERİ

Çizelgeleme problemlerinin genel gösterimi **A/B/C/D** olup, burada;

A = İşlerin geliş biçimi; gelişler toplu (n), belirli zamanlarda ya da tek tek belirsiz (rassal),

B = Tezgah sayısı (m_,

C = İşleme biçimi; işlerin belirli bir akışı izlediği akış atölyesi (F) veya rassal bir akışı izlediği iş atölyesi (G),

D = Değerlendirme (Performans) ölçütü.

Örneğin, n/3/F/F anlatımı ile; n işin 3 tezgahtan oluşan bir akış atölyesinde işlenebileceği ve değerleme ölçütünün ortalama akış süresinin (F) en küçüklenmesi olduğu anlaşılır.

Parametreler :

r_i = i işinin işlenmesi için hazır olduğu zaman,

d_i = i işinin teslim tarihi,

k = 1, 2,....., m makina sayısı,

i = 1, 2,....., n iş sayısı,

j = 1, 2,....., gi i işinin işlem sayısı,

a_i = di - ri i işinin yapılması için eldeki süre,

k_{ij} = i işinin j işleminin atandığı tezgah,

P_{ij} = i işinin j işleminin işlenme süresi,

$P_i = \Sigma P_{ij}$ = i işinin toplam işlenme süresi,

w_{ij} = i işinin j işleminden önceki bekleme süresi,

$W_i = \Sigma W_{ij}$ = i işinin toplam bekleme zamanı,

$C_i = r_i + w_i + P_i$, i işinin tamamlanma zamanı,

$F_i = C_i + r_i = W_i + P_i$, i işinin akış süresi, (Atölye veya üretim süresi)

$L_i = C_i - d_i$, i işinin gecikme durumu,

$L_i = F_i + r_i - d_i = F_i - a_i$; $L_i > 0$ iş erken,

T_i = Enb [0, L_i], i işinin gecikmeli olması,

E_i = Enb [0, L_i], i işinin erken olması,

F_{enb} = En büyük akış süresi,

$C = \Sigma C_i / n$,

O_{ij} = i işinin j işlemi, (kij, pij)

$P_{(i)}$ = Çizelgede i. nci sıradaki işin işlem süresi,

$C_{(i)}$ = Çizelgede i. nci sıradaki işin tamamlanma zamanı,

olarak tanımlandığında, ortalama ya da en büyük C, F, L_i, W_i ve T_i birer değerleme ölçütü olarak alınır ve enküçüklenmeye çalışılır [Hofeman, 1968: Çınar, 1990].

ÇİZELGELEME KURALLARI

Üretim çizelgelemesinde bir takım kurallar, teorem biçiminde ortaya konulmaktadır. Önemli kurallardan bazıları izleyen biçimdedir [Baker, 1976 : Acar, 1985].

Kural 1. Bir n/1/F probleminin çizelgelemesinde F_{enk}, izleyen anlatım ile elde edilir;

$$P_{(1)} <= P_{(2)} <= P_{(3)}, ..., <= P_{(n)}$$

Bu kurala "En Kısa Süreç Zamanı Sıralaması (EKZ)", adı verilir. Bunun tersi, "En Uzun Süreç Zamanı Sıralaması (EUZ)" olup, F_{enb} ortalama akış süresinin enbüyük değerini verir.

Kural 2. Bir n/1/L_{enb} veya n/1/T_{enb} probleminde L_{enb} (enbüyük gecikme durumu ve T_{enb} (enbüyük gecikme), "En Erken Teslim Tarihi (ETT)" sıralaması olan;

$$d_{(1)} <= d_{(2)} <= d_{(3)}, \ldots, <= d_{(n)}$$

ile enküçüklenir.

Kural 3. Bir n/1/L probleminde, L (ortalama gecikme durumu);

$$P_{(1)} <= P_{(2)} <= P_{(3)}, \ldots, <= P_{(n)}$$

ile enküçüklenir.

Kural 4. Bir $n/1/L_{enk}$ veya $n/1/T_{enk}$ probleminde L_{enk} ve T_{enk} izleyen sıralama "En Kısa Aylık Sürü (EAS)", ile enbüyüklenir.

$$d_{(1)} - P_{(1)} <= d_{(2)} - P_{(2)} <= d_{(3)} - P_{(3)}, \ldots, <= d_{(n)} - P_{(n)}$$

Kural 5. Bir n/1/L probleminde L, EKZ ile enküçüklenir.

Örnek : [Çınar, 1990, s. 247].

İş	P_i	d_i	$d_i - P_i$
1	6	4	-2
2	2	7	5
3	3	7	4

Bu problemde olası 6 çizelge seçeneği bulunmaktadır. Her bir seçeneğe ve uygulanan ölçütlere göre sonuçlar izleyen tablo'da özetlenmektedir.

Çizelge	İş	$F_i = C_i$	F	L_i	L_{enb}	L_{enk}	T_i	T_{enb}	T_{enk}	T
2,3,1	1	11		7	7	-5	7	7	0	2.3
	2	2	6	-5			0			
EKZ	3	5		-2			0			
1,2,3	1	6		2			2			
	2	8	8.3	1	4	1	1	4	1	2.3
ETT	3	11		4			4			
1,3,2	1	6		2			2			
EAS	2	11	8.7	4	4	2	4	4	2	2.7
ETT	3	9		2			2			
3,2,1	1	11		7			7			
EAS	2	5	6.3	-2	7	-4	0	7	0	2.3
ETT	3	3		-4			0			
2,1,3	1	8		4			4			
	2	2	7	-5	4	-5	0	4	0	2.7
	3	11		4			11			
3,1,2	1	9		5			5			
	2	11	7.7	4	5	-4	4	5	0	3
	3	3		-4			0			

Tablodan izleneceği gibi F = 6 olarak, EKZ ile enküçüklenmiştir. Lenk ve Tenk, EAS ile enbüyüklenmiş ve Lenb ile Tenb, ETT ile enküçüklenmiştir.

n/1/L_{enb} veya T_{enb} PROBLEMLERİNDE TESLİM TARİHİ KISITLARI ALTINDA EN KISA SÜREÇ ZAMANI ÇİZELGESİ

Bu tür çizelgelerde toplam akış süresi F, teslim süre kısıtları temelinde enküçüklenmeye çalışılır. Bu amaçla geliştirilen bir sezgisel algoritma izleyen adımlardan oluşturulmuştur :

Adım 1. $T_{enb} = 0$ olan bir çizelgenin varlığı ETT ile araştırılır. Eğer $T_{enb} > 0$ ise 5. adıma geçilir.

Adım 2. $d_j >= \Sigma P_i$ olan tüm j ler arasından enbüyük P_i değeri içeren bir k işi seçilir. k işi n. pozisyona yerleştirilir. $P_k = P_n$ ataması yapılır.

Adım 3. Kalan n-1 işi için; $C_{n-1} = P_i - P_n$ ve $d_j >= C_{n-1}$ temelinde kalan j'ler arasından P_i değeri enbüyük olan bir I işi seçilerek (n-1). pozisyonuna yerleştirilir. $P_m = P_{n-1}$ dönüşümü gerçekleştirilir.

Adım 4. Adım n-1 kere yinelenerek istenen çizelge elde edilir.

Adım 5. $T_{enb} > 0$ ise problemde geciken işler bulunmaktadır. Bu durumda geciken iş sayısını enküçükleyen çizelge oluşturulabilir. Eldeki ETT sıralamasında $F_i - d_i = L_i$ değerleri arasından pozitif değer sayısı geciken iş sayısıdır (Nt). Eğer Nt = 0 ise eldeki sıralamaya yeni kümeyi ekle (çözüm), çizelgeyi ve Nt değerini yaz ve dur, Nt > 0 ise ilk geç işi k olarak tanımla.

Adım 6. İlk k işin sıralamasında öyle bir j işi bulunuz ki, P_j enbüyük olsun. j işini başka bir kümeye atayarak, Adım 5'e gidiniz.

Örnek : [Çınar, 1990, s. 247].

(* Teslim süresi*) d1 ={24,21,8,5,10,23};

(*İş süresi*) p1 = {4,7,1,3,2,5};

biçiminde 6 iş teslim kısıtlarının karşılanması temelinde sıralanmak isteniyor.

Bilgisayarda kullanılacak komut;

In [1] : =

d1 = {24,21,8,5,10,23}

p1 = {4,7,1,3,2,5};

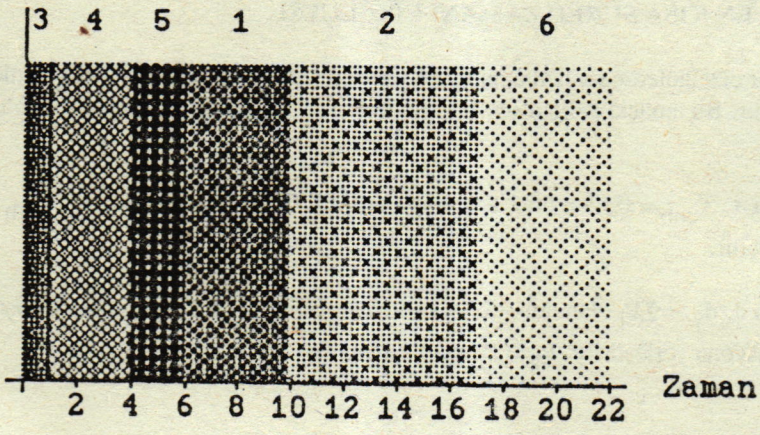
En kısa Teslim [d1, p1]

olup, çıktısı;

Out [1] : =

Enb F = 22 *** Ortalama F = 10

Optimum Sıralama = { 3, 4, 5, 1, 2, 6}



İş	3	4	5	1	2	6
Başla	0	1	4	6	10	17
Bit	1	4	6	10	17	22
Teslim	8	5	10	24	21	23

biçiminde olup, algoritma temelinde programın açılımı izleyen biçimdedir :

Adım 1. $T_{enb} = 0$ olan bir çizelgenin varlığı ETT ile araştırılır. Eğer $T_{enb} > 0$ ise 5. Adıma geçilir.

```
In [2] : =
    i = Range [1, Length [d]];
    issıra = Table [0, {Length [i]}];
    set = {d, p, i}; tci = { }; set1 = set;
    yersıra = Range [1, Length [i]];
    kul = Sort [Transpose [set]]; kull = kul;

Out [2] : =
    set = {{5, 3, 4}, {8, 1, 3}, {10, 2, 5}, {21, 7, 2}, {23, 5, 6}, {24, 4, 1}}

In [3] : =
    f1 = TArastır [kul]];
    li = f1 , Sort [d];

Out [3] : =
    f1 = {3, 4, 6, 13, 18, 22}
    sd = {5, 8, 10, 21, 23, 24}
    li = {-2, -4, -4, -8, -5, -2}
```

Tüm değerler negatif olduğundan, geciken iş bulunmamaktadır. Adım 2'ye geçilir.

Adım 2. $d_j >= \Sigma P_i$ olan tüm j ler arasından enbüyük P_i değeri içeren bir k işi seçilir. k işi n. pozisyona yerleştirilir. $P_k = P_n$ ataması yapılır.

```
In [4] : =
    f1 = TArastır [kul]];
    Ci = Max [f1]; tci = Append [tci, ci];
    poz = Length [yersıra];
    islem = Transpose [kul]];
    sec = Select [islem [[1]], Apply [And, Positive [# - Ci]]&];

Out [4] : =
    Ci = 22
```

```
islem1 = {5, 8, 10, 21, 23, 24}
sec = {23, 24}
```

Toplam akış süresi olan 22 değerinden büyük iki Pi değeri bulunmaktadır.

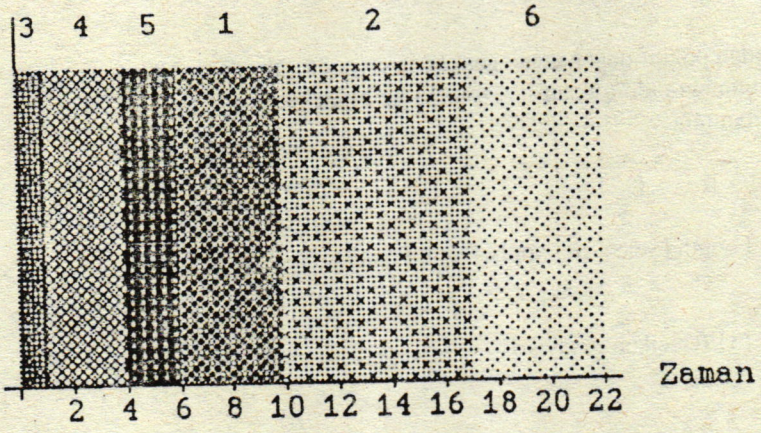
```
In [5] : =
    top = { };
    Do [bi = Take [islem [[2]];
    Position [islem [[1]], sec [[yy]] [[1]]] [[1]];
    top = Append [top, bi], {yy, 1, Length [sec]}];
    yer1 = Position [islem [[2]], Max [top]] [[1, 1]];
    yer = Last [kul [[yer1]]];
    issıra [[poz]] = yer;
    yersıra = Drop [yersıra, {poz}]; poz = poz-1;
    Ci = Ci -Max [top];
    kul = Drop [kul, {yer1}]];

Out [5] : =
    poz = 6
    islem2 = {3, 1, 2, 7, 5, 4}
    top = {5, 4}
    yer1 = 5
    yersıra = {1, 2, 3, 4, 5}
    issıra = {0, 0, 0, 0, 0, 6}
    kul = {{5, 3, 4}, {8, 1, 3}, {10, 2, 5}, {21, 7, 2}, {24, 4, 1}}
```

İlk döngü sonucunda 6 no. lu pozisyona 6 kodlu iş atanır. En son değerler temelinde program n-2 kez yinelenerek; sıralama düzeni (issıra), işlerin başlama, bitiş ve teslim zamanlarını gösteren (tablo) ve Gantt benzeri gösterim çıktısına ulaşılır.

```
Enb  F=  22  ***        Ortalama  F  =  10
Optimum  Sıralama  =  {  3,  4,  5,  1,  2,  6}
```

İş	3	4	5	1	2	6
Başla	0	1	4	6	10	17
Bit	1	4	6	10	17	22
Teslim	8	5	10	24	21	23



Örnek 2 : [Çınar, 1990, s. 248].

```
In [1] : =
    (*Teslim süresi*) d1 = {2, 7, 8, 13, 11};
    [*İşlem süresi*) p1 = {1, 5, 3, 9, 7};
    En Kısa Teslim [d1, p1];

Out [1] : =
    Geciken İş Sayısı = 2*** En iyi Sıralama ->{1, 3, 5, 2, 4}
```

Adım 1. $T_{enb} = 0$ olan bir çizelgenin varlığı ETT ile araştırılır. Eğer $T_{enb} > 0$ ise 5. Adıma geçirilir.

```
In [1] : =
    islem = Transpose [kul]];
    f1 = NuArastır [islem];
    Ci = Max [f1];
    Li = f1 - islem [[1]];

Out [1] : =
    islem = {{2, 7, 8, 11, 13}, {1, 5, 3, 7, 9}, {1, 2, 3, 5, 4}}
    f1 = {1, 6, 9, 16, 25}
    Ci = 25
    Li = {-1, -1, 1, 5, 12}
```

Tüm değerler negatif olmadığından, geciken iş bulunmaktadır. Adım 5'e geçilir.

Adım 5. $T_{enb} > 0$ ise problemde geciken işler bulunmaktadır. Bu durumda geciken iş sayısını enküçükleyen çizelge oluşturulabilir. Eldeki ETT sıralamasında $F_i - d_i = L_i$

değerleri arasından pozitif değer sayısı gecken iş sayısıdır (Nt). Eğer Nt = 0 ise eldeki sıralamaya yeni kümeyi ekle (çözüm), çizelgeyi ve Nt değerini yaz ve dur, Nt > 0 ise ilk geç işi k olarak tanımla.

```
Tenb = Max [Li] = {12} > 0;
In [2] : =
    Nt = Length [Select [li, Apply [And, Positive [# -0 ]]&]]];

Out [2] : =
    Nt = {3} (Geciken iş sayısı)

In [3] : =
    If [secsı ==1,
    sec = Select [li, Apply [And, Negative [# - 0]]&],
    sec = Select [li, Apply [And, Positive [# - 0]]&]];
    If [sec == { }, islem [[3]] = Append [islem [[3]], sota];
    Print ["Geciken İş Sayısı =", Length [sota],
    "*** Eniyi Sıralama -> ", Flatten [islem [[3]]]];
Break [ ]];

Out [3] : =
    sec = {-1, -1}
```

1 ve 2 kodlu geciken işlerden hangisinin k olarak tanımlanacağı izleyen adım ile belirlenir.

```
In [4] : =
    top = { };
    Do [
    ai = Position [li, sec [[yy]]] [[1]] [[1]];
    bi = islem [[2]] [[ai]];
    li [[ai]] = 0;
    top = Append [top, bi], {yy, 1, Length [sec]}];
    yer1 = Position [islem [[2]], Max [top]] [[1, 1]];
    yer2 = islem [[3]] [[yer1]];
    sota = Append [sota, yer2];
    kul = Drop [kul, {yer1}];

Out [4] : =
    ai = 1
    bi = 1
    Li = {0, -1, 1, 5, 12}
    ai = 2
    bi = 5
    Li = {0, 0, 1, 5, 12}
```

yer1 = 2
yer2 = 2
kul = {{2, 1, 1}, {8, 3, 3}, {11, 7, 5}, {13, 9, 4}

2 kodlu işin P değeri daha büyük olduğundan (5), 2 kodlu iş k olarak tanımlanarak yeni bir kümeye (sota) atanır.

sota = {2}
In [5] : =
islem [[3]] (Kalan işler).

Out [5] : =
{1, 3, 4, 5}

2. Öteleme :

islem = {{2, 8, 11, 13}, {1, 3, 7, 9}, {1, 3, 5, 4}}
f1 = {1, 4, 11, 20}
Ci = 20
Li = {-1, -4, 0, 7}

sec = {7}
ai = 4
bi = 9
Li = {-1, -4, 0, 0}
yer1 = 4
yer2 = 4
kul = {{2, 1, 1}, {8, 3, 3}, {11, 7, 5}}

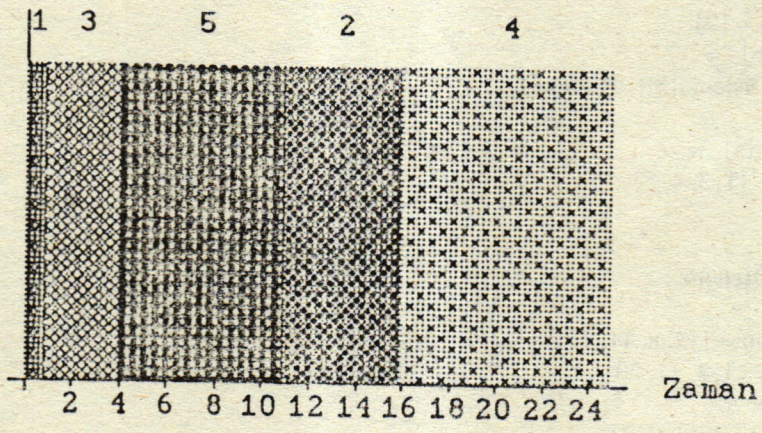
Kalan işler = {1, 3, 5}
sota = {2, 4}

Algoritma sonunda izleyen çıktı oluşur :

In [6] : =
d1 = {2, 7, 8, 13, 11};
p1 = {1, 5, 3, 9, 7};
EnKısaTeslim [d1, p1];

Out [6] : =
Geciken İş Sayısı = 2*** Eniyi Sıralama -> {1, 3, 5, 2, 4}

İş	1	3	5	2	4
Başla	0	1	4	11	16
Bit	1	4	11	16	25
Teslim	2	8	11	7	13



SONUÇ

Sıralama problemlerinin statik yapıda olan n/1 türü problemlerde, n adet işin tek bir tezgahta en iyi sırada işlenmesi amaçlanır. Bu problemlerde işler hangi sırada yapılırsa yapılsın bütün işleri bitirmek için gerekli toplam süre hiçbir zaman değişmez. Diğer taraftan işlerin ortalama akış zamanları (F) ve ortalama bekleme zamanları azaltılmak isteniyorsa, işler en kısa süreç zamanı (EKZ) ile başlatılmalıdır. Bu kural yardımıyla işler artan zamanlarına göre sıralanırlar.

Bu çalışmada, iş sıralama ve çizelgeleme problemlerinin genel teorik çerçevesi sunularak, genel kabul görmüş çizelgeleme kuralları tanıtılmış ve bu alanda kullanım yaygınlığı kazanmış EKZ ölçütüne, teslim tarihi öncelik kuralı eklenmesiyle oluşturulan bir sezgisel algoritma verilmiştir. Algoritmada, teslim tarihi kısıtları karşılanmadığında ise değerleme ölçütü "geciken iş sayısı" ölçütüne dönüştürülerek eniyi sıralama önerilmektedir.

En kısa Teslim adı verilen bu algoritma Mathematica dili ile kodlanmış olup, çalışmada sunulan örneklere ilişkin çözüler 10-11 saniyede türetilmiştir. Test çalışmaları sürdürülen ve bir başka yazıda tanıtılacak olan OPTSIRA n/3 ve değişen iş sıralı üretim çizelgeleme algoritmasında ise eniyi sıralama ve Gantt diyagramı biçiminde gösterilmesi ortalama 18 saniyede gerçekleşmektedir.

KAYNAKÇA

ACAR, Nesime; **Üretim Planlaması ve Uygulamaları.** MPM Yayını No. : 280, Ankara 1985.

BAKER, R.; **Introduction to Sequencing and Scheduling.** John Wiley and Sons, NewYork, 1974.

ÇINAR, Mehmet; **Yönetsel Kararlara İlişkin Sayısal Yöntemler.** Erciyes Üniversitesi Yayını No. : 8, Kayseri, 1990.

HOFEMAN, R.J.; **Production : Concepts, Analysis and Control.** 3.rd Ed., Columbus, Ohio : Merrill, 1976.

EK : EnKısaTeslim PROGRAMI

```
EnKısaTeslim [d1--, p1--] : =
Block [{d=d1, p=p1},
        i = Range [1, Length [d]];
        issıra = Table [0, {Length [i]}];
        set = {d, p, i}; tci = { }; set1 = set;
        yersıra =Range [1, Length [i]];
        kul = Sort [Transpose [set]]; kull = kul;
        f1 = TArastır [kul];
        li = f1 - Sort [d];

If [Max [li] <= 0,
Do [
        f1 = TArastır [kul];
        Ci = Max [f1]; tci = åppend [tci, Ci];
        poz = Length [yersıra];
        islem = Transpose [kul];
        sec = Select [islem [[1]], Apply [And, Positive [# - Ci]]&];

        top = { };
Do[bi = Take [islem [[2]],
        Position [islem [[1]], sec [[yy]] [[1]] [[1]];
        top = Append [top, bi], {yy, 1, Length [sec]}];
        yer1 = Position [islem [[2]], Max [top]] [[1, 1]];
        yer = Last [kul [[yer1]]];
        issıra [[poz]] = yer;
        yersıra = Dorp [yersıra, {poz}]; poz = poz-1;
        Ci = Ci - Max [top];
        kul = Drop [kul, {yerf}]
        , {Length [i]}
],
Nut [set1]; Break [ ]];
Print ["Enb F = ", Max [tci]," ***",
"   Ortalama  F = ",
Apply [Plus, Flatten [tci]]/Length [i]];
Print ["Optimum Sıralama = ", issıra]
mer = 0;
cınar = { }; zaman = set1 [[2]]; teslim = set1 [[1]];
For [k = 1, k! = Length [issıra]+1, k++,
    efk = issıra [[k]];
    basla = mer;
    bit = basla + zaman [[efk]];
```

```
    tes = teslim [[efk]];
    top = {efk, basla, bit, tes};
    mer = bit;
    cınar = Append [cınar, top]];
    kaz = ColumnTake [cınar, {2, 3}];
tabkaz = Prefend [cınar, {İs, basla, Bit, Teslim}];
tablo = Transpose [tabkaz]//TableForm; Print [tablo];
GANTCIZ [{kaz}]

]

ColumnTake [data : {---List},  spec--] : =
Transpose [Take [Transpose [data], spec]]

TArastır [data 75 --] : =
    Block [{ara = dat 75, c, b},
        f = Transpose [ara] [[2]];
        f1 = { };
    Do [k = sum [f [b]], {b, 1, c}];
        f1 = Append [f1, k], {c, 1, Length [f]}]; f1
    ]

NuArastır [dat 78 --] : =
    Block [{ara = dat78, c, b},
        f = ara [[2]]; f1 = { };
    Do [k = Sum [f [[b]], {b, 1, c}];
        f1 = Append [f1, k], {c, 1, Length [f]}]; f1
    ]

Nut [dat 58 --] : =
    Block [
        {set = set1},
        sota = { }; secsı = 0;
        kul = Sort [Transpose [set]];
    Do [  secsı = secsı + 1;
        islem = Transpose [kul]...
        f1 = NuArastır [islem];
        Ci = Max [f1]; li = f1 - islem [[1]];
    If [secsı == 1,
        sec = Select [li, Apply [And, Negative [# - 0]]&],
        sec = Select [li, Apply [And, Positive [# -0]]&]];
    If [sec == { }, islem [[3]] = Append [islem [[3]], sota];
        Print ["Geciken İş sayısı =" , Length [sota],
        "*** Eniyi Sıralama -> ", Flatten [islem [[3]]]];
```

```
    Break [ ]];
    top = { };
    Do [
        ai = Position [li, sec [[yy]]] [[1]] [[1]];
        bi = islem [[2]] [[ai]]; li [[ai]] = 0;
        top = Append [top, bi], {yy, 1, Length [sec]}];
        yer1 = Position [islem [[2]], Max [top]] [[1, 1]];
        yer2 = islem [[3]] [[yer1]];
        sota = Append [sota, yer2];
        kul = Drop [kul, {yer1}],
        {Length [p]}
    ]]

GANTCIZ [data12 --] : =
Block [{basla = data12, to},
    kod = Range [1, 30, 1];
    kod1 = {RGBColor [1, 0, 0], RGBColor [0, 1, 0],
    RGBColor [0, 0, 1], RGBColor [1, 0, 1], RGBColor [0, 1, 1],
    RGBColor [1, 1, 0], RGBColor [0, 1, 0]};

kutular = { }; eksen = { };
For [x = 1, x! = Length [basla]+1, x++,
    s1 = basla [[x]];
    kutu1 = { };
    eks1 = { };
    For [x1 = 1, x1! = Length [s1]+1, x1++,
    s2 = s1 [[x1]];
kutu = {kod1 [[x1]], Rectangle [{s2[[1]], kod [[x]]},
{s2 [[2]], kod [[x]] + 0.15}] };
kutu1 = Append [kutu1, kutu];
eks = Text [issıra [[x1]], {(s2 [[1]] + s2 [[2]]/2, 1.17];
eks1 = Append [eks1, eks];
    ]; kutular = Append [kutular, kutu1];
    eksen = Append [eksen, eks1]
    ];
    Show [Graphics [kutular, Axes -> {0, 1},
    AxesLabel -> {"  Zaman", "    "},
Ticks -> {Rangle [0, 32, 2], None}, Graphics [eksen]]
]
```