T.C. Balıkesir Üniversitesi
Endüstri Mühendisliği Bölümü

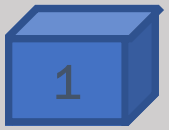
EMM4129
ÇİZELGELEME

Ders Notları

Doç. Dr. İbrahim KÜÇÜKKOÇ
Web: http://ikucukkoc.baun.edu.tr
Email: ikucukkoc@balikesir.edu.tr

Güncelleme: 19/07/2020

Ders Planı
Değerlendirme Kriterleri
Yararlanılacak Kaynaklar
Çizelgeleme Nedir?



Genel Bakış

Dersin Amaci:

Endustride karsilasilan farklı tipteki çizelgeleme problemleri ile ilgili:

- Temel terminolojiyi sunmak.
- Bunlarin matematiksel olarak nasil modellenebilecegini gostermek.
- Çözümlenmesi ve analiz edilmesinde kullanilan yontemleri ornekler ve paket program (LEKIN®) uygulamalariyla aciklamak.

Genel Bakış

Dersin Web Sayfasi:

http://ikucukkoc.baun.edu.tr/lectures/EMM4129

Değerlendirme:

- Vize (%40) + Final (%60)
- Final puanının %30'u dönem icinde yapılacak olan projeden alınacaktır.

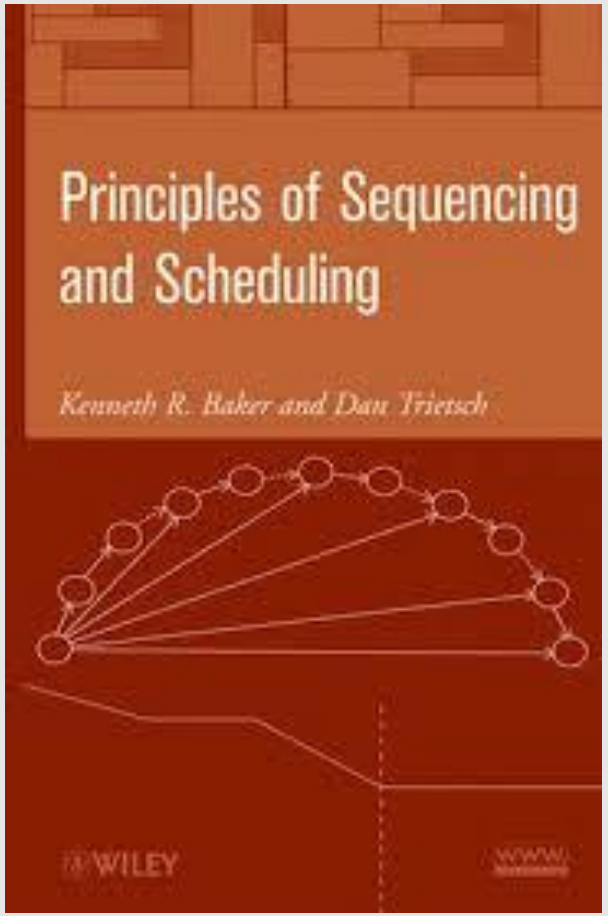
Derse Katılım:

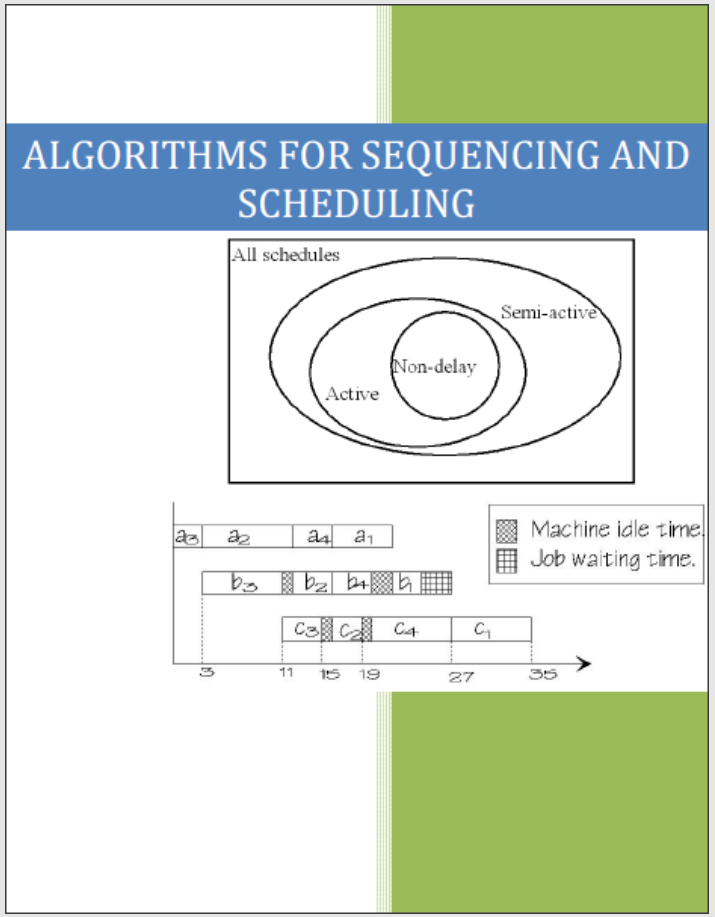
- Derslere zamaninda gelmeniz gerekmektedir.
- 5 hafta yada daha fazla devamsızlık yapan öğrenciler devamsızlıktan bırakılacak ve final sınavına alınmayacaktır.
- Derste kesinlikle cep telefonu vb. konuyla alakasız materyallerle ilgilenilmemesi beklenmektedir.

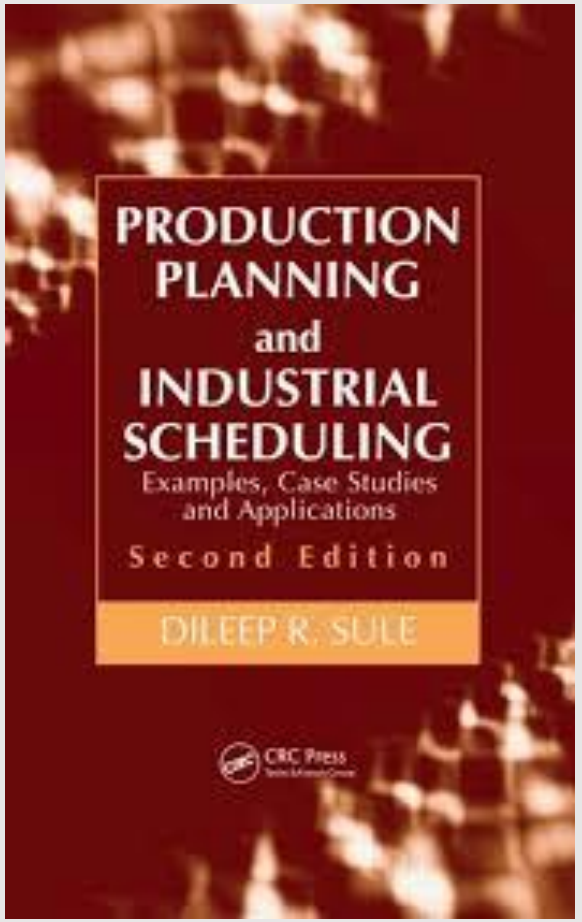
Yararlanılacak Kaynaklar

- İş Sıralama ve Çizelgeleme Ders Notları
  Prof. Dr. Huseyin Basligil
  http://www.yarbis.yildiz.edu.tr/basligil/course/viewCourse/id/461
- Üretimde Sıralama ve Çizelgeleme Ders Notları
  Yrd. Doc. Dr. A. Ayca Supciller
- Principles of Sequencing and Scheduling
  Kenneth R. Baker, Dan Trietsch
  http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0470391650.html
  Numerical Examples:
  http://ikucukkoc.baun.edu.tr/lectures/lecturenotes/EMM4129/Baker_Numerical_Examples_Excel_Files.zip
- Algorithms for Sequencing and Scheduling
  Ibrahim M. Alharkan
  http://ikucukkoc.baun.edu.tr/lectures/lecturenotes/EMM4129/Algorithms_for_Sequencing_and_Scheduling.pdf
- Production Planning and Industrial Scheduling: Examples, Case Studies and Applications, Second Edition
  Dileep R. Sule
  https://www.crcpress.com/Production-Planning-and-Industrial-Scheduling-Examples-Case-Studies-and/Sule/p/book/9781420044201

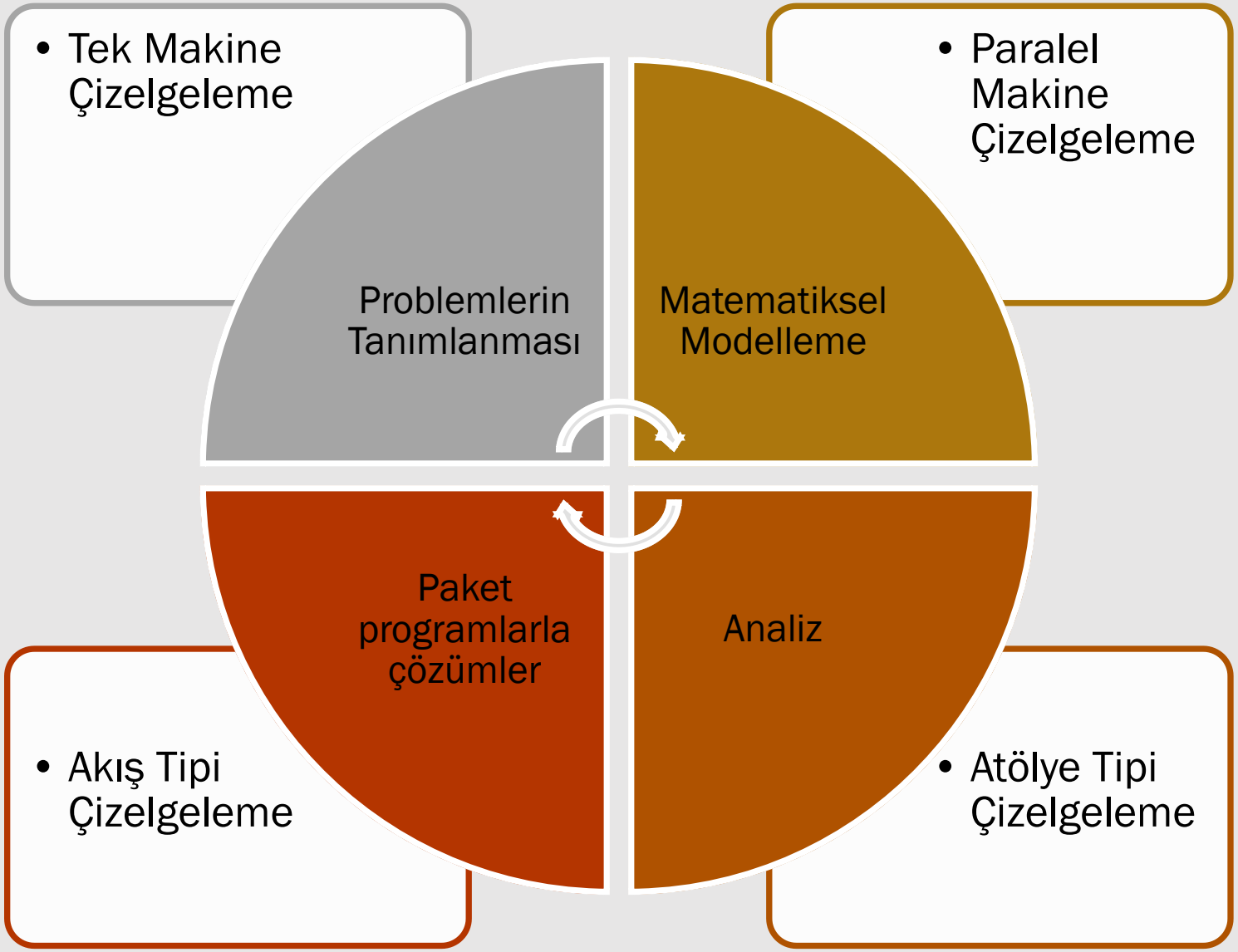
Yararlanılacak Kaynaklar



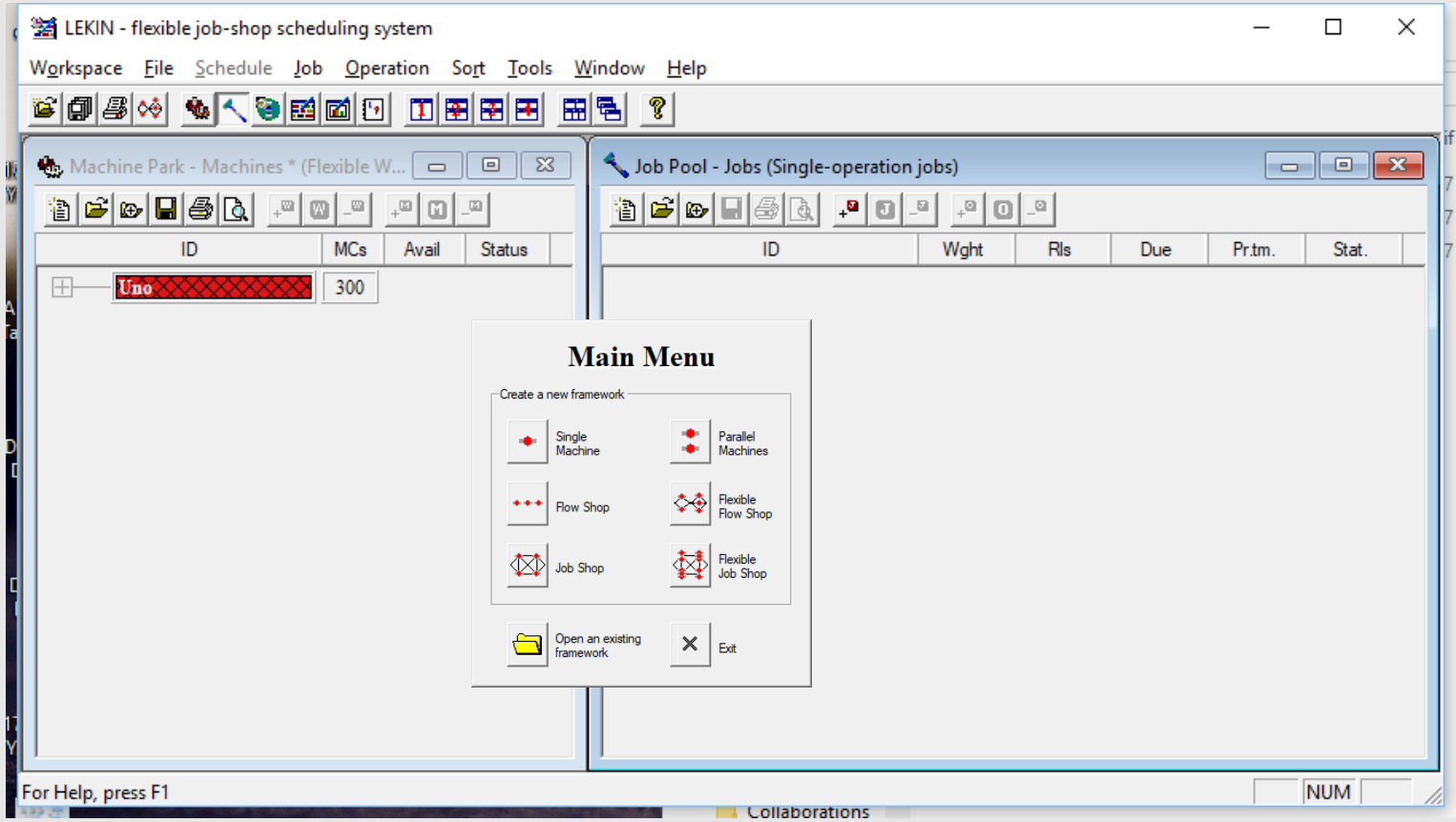




Ders İçeriği



Kullanılacak Paket Program



Lekin® - New York University

Download: http://web-static.stern.nyu.edu/om/software/lekin/download.html

Sıralama ve Çizelgeleme Nedir?

- Bir dizi işin belirli özelliklerine göre sıraya dizilmesi işlemine sıralama denir.

- Çizelgeleme ise sınırlı kaynakların (makine, işçi, donanım, alet v.b.) belirli bir amaç veya amaçlar doğrultusunda, belirli kısıtlar altında ve belirli bir zaman aralığında, işlere atanması ile ilgili karar verme sürecidir.

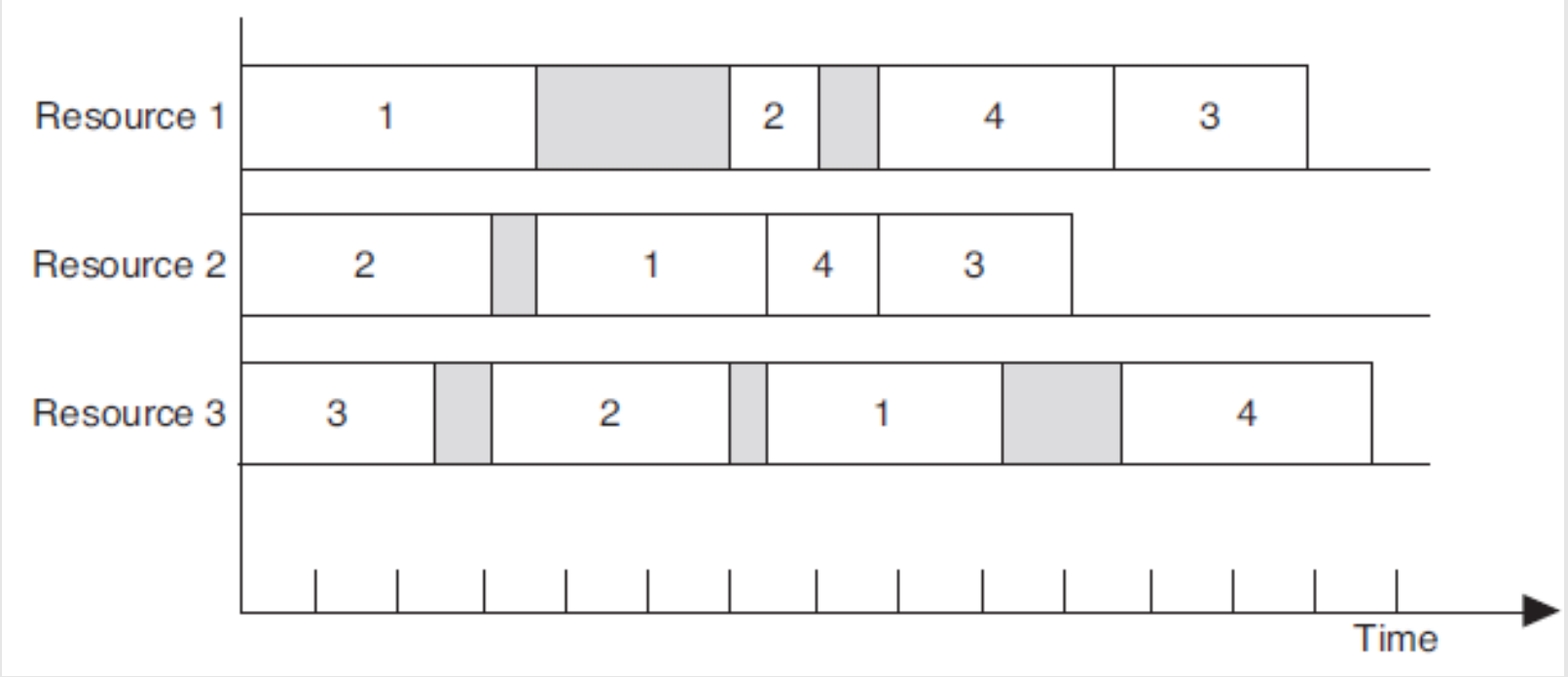
İş Çizelgeleme bir ürünü oluşturan iş parçalarının eldeki tek veya çok sayıda makinelerde hangi sırada, ne zaman işleneceğinin saptanmasıdır.

Çizelgelemenin Unsurları

1. Faaliyet (Uretim, servis,vb.)
2. Kaynak (Makine, Operator vb.)
3. Zaman (Baslama, Bitis, vb.)

Sıralama ve Çizelgeleme Nedir?

(Resource: Kaynak.
Örneğin Makine, Operator…)



https://goo.gl/images/Ee8y6e

Sıralama ve Çizelgeleme Nedir?

- Çizelgeleme faaliyeti sonucu, bir üretim veya hizmet merkezinde işlem görmesi gereken birden fazla sayıda işin “hangi sıra” ile yapılacağı belirlenir.
- Yapılan sıra bir performans kriteri veya kuralına göre değerlendirilir, örneğin ilk gelen ilk işlem görür veya en acil olan ilk işlem görür gibi kurallar olabilir.

https://goo.gl/images/h1L7VV

https://goo.gl/images/uckcSb

https://goo.gl/images/D9fPtK

Çizelgelemenin “en” temel amaçları nelerdir?

- Üretim olanaklarının en “etkin” biçimde kullanılması.
- Müşteri taleplerine olabildiğince “çabuk” cevap verilmesi.
- İşlerin, teslim tarihlerinde gecikmeye neden olunmadan tamamlanması.
- Yarı mamul envanterinin en küçüklenmesi.
- Fazla mesai çalışmalarının en küçüklenmesi

Çizelgeleme Uygulamaları

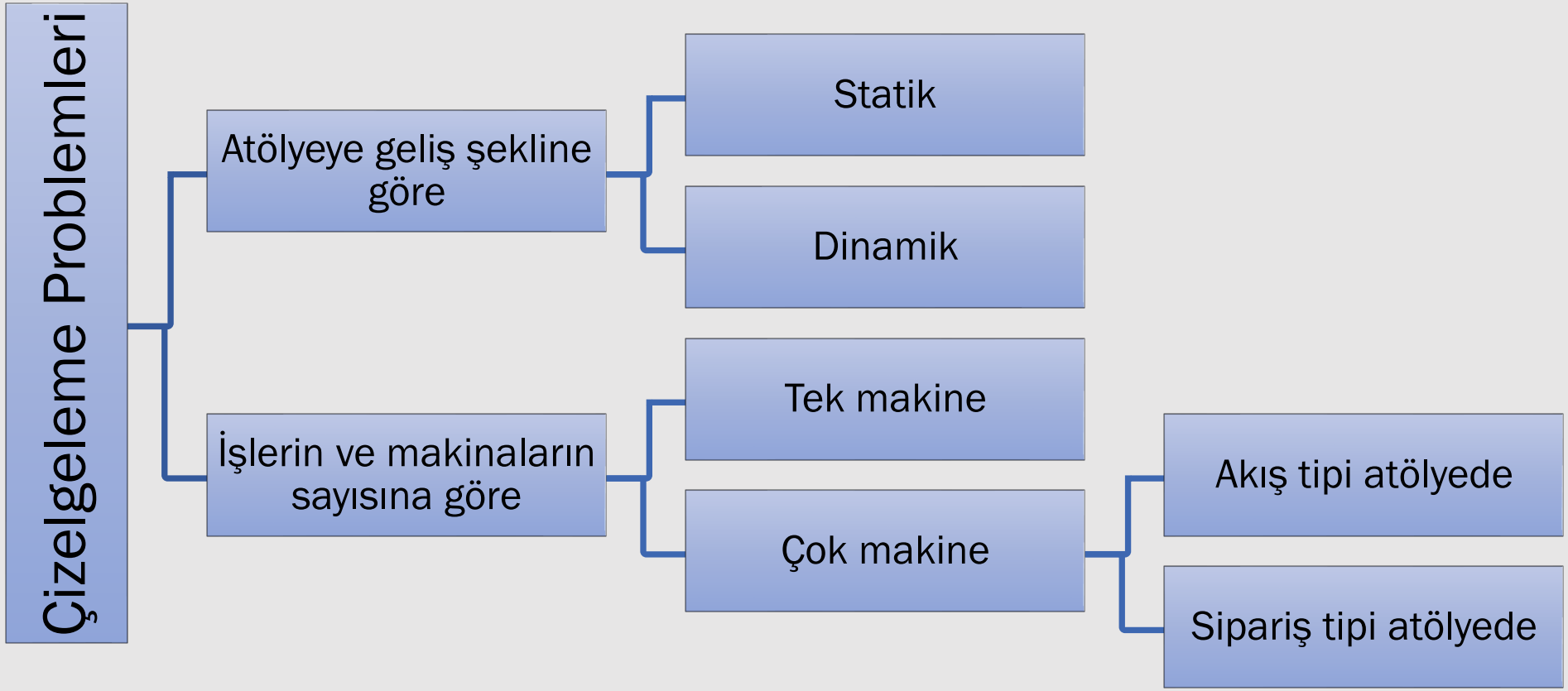
- Kaynak Kısıtlı Proje çizelgeleme
- Montaj hatlarının çizelgelenmesi
- Uçak/Otobüs/Tren Çizelgeleme
- Hastanelerde doktor/hemşire çizelgeleme
- Ders programı/sınav çizelgeleme
- Otel/havayolu/araba rezervasyon sistemleri
- Mürettebat çizelgeleme
- Proje çizelgeleme
- Atölye çizelgeleme
- Esnek montaj hatlarının çizelgelenmesi
- Ekonomik parti çizelgeleme
- Tedarik zincirinde planlama ve çizelgeleme
- Araç/Ekipman (AGV, vinç) çizelgeleme ve rotalama

Çizelgeleme Neleri Önler?

- Siparişlerin geç teslim edilmesi,
- Kurulu kapasitenin tam kullanılmaması,
- Üretim temin süresinin artması,
- Üretimde darboğazlar,
- WIP stoklarının artması,
- Müşterilerin memnuniyetsizliği,
- Müşteri kaybı.

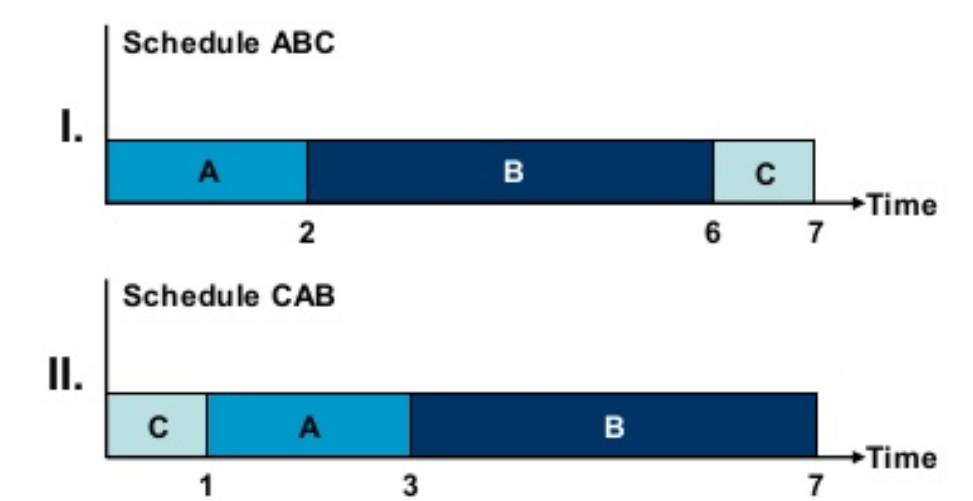
Planlama Aralığı	Girdiler	Hesaplamalar ve Kararlar	Planlar/Çizelgeler	Çıktılar
Uzun Vade	Uzun Vadeli Talep Tahminleri Sermaye Durumu ve İşyeri Analizi Kapasite Verileri ve Analizi		Uzun Vadeli Kapasite Planı	1- Üretim faaliyetleri planı 2- Majör Tedarikçi Planı 3- Majör işleme, proses geliştirme planları (uzun dönem planlama için stratejik teknoloji, dizayn vs geliştirme ve seçme stratejileri)
Orta Vade	Orta Vadeli Talep Tahminleri	Kapasite Kısıtları	Toplu Üretim Planı	1- İş gücü planları 2- Malzeme tedarikle ilgili kontratlar 3- Faaliyet Değişiklik planları 4- Stok Planları
Kısa Vade	Kısa Vadeli Talep Tahminleri Eldeki Siparişler Diğer Siparişler Tedarikçilerin Malzeme Yeterliliği	İş Merkezi Kapasite Kısıtları Stok Durumu Kısa Vadeli bitmiş madde talep tahmini İş Merkezi Yükleme Çizelgesi İş Merkezi Çizelgeleme Kararları Stok Durumu (parçalar) Malzeme Durumu	Ana Üretim Planı Kapasite İhtiyaç Planlaması Malzeme İhtiyaç Planlaması	1- Kısa Vadeli bitmiş parça çizelgesi 2- Her iş merkezinde üretilecek kısa vadeli parça, alt montaj, montaj parçaları 3- İş merkezindeki üretim çizelgesini destekleyecek malzeme planı 4- İş merkezleri için gerekli olan kısa vadeli atölye planları (partilerin hareketleri, ayar planları vs.)

Üretimde Çizelgeleme Türleri



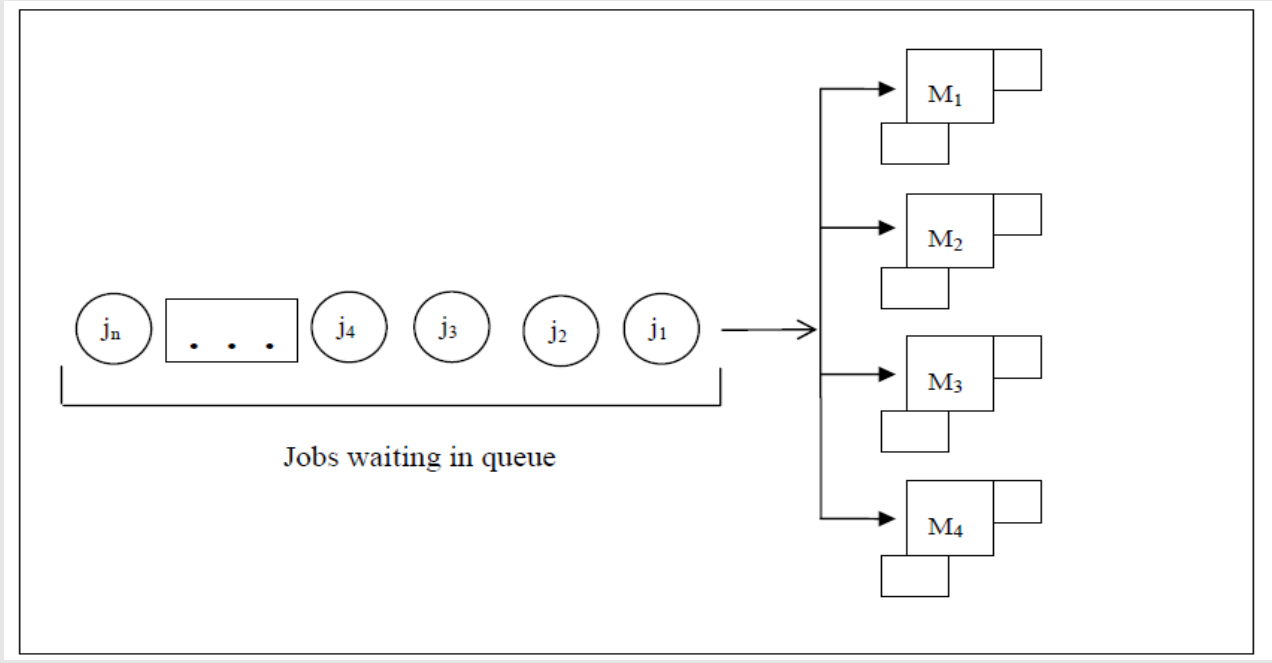
İşlerin ve Makinaların Sayısına Göre

- Tek makine çizelgeleme
- Paralel makine çizelgeleme
- Akış tipi çizelgeleme
- Atölye tipi çizelgeleme



Tek makineli çizelgeleme ortamı

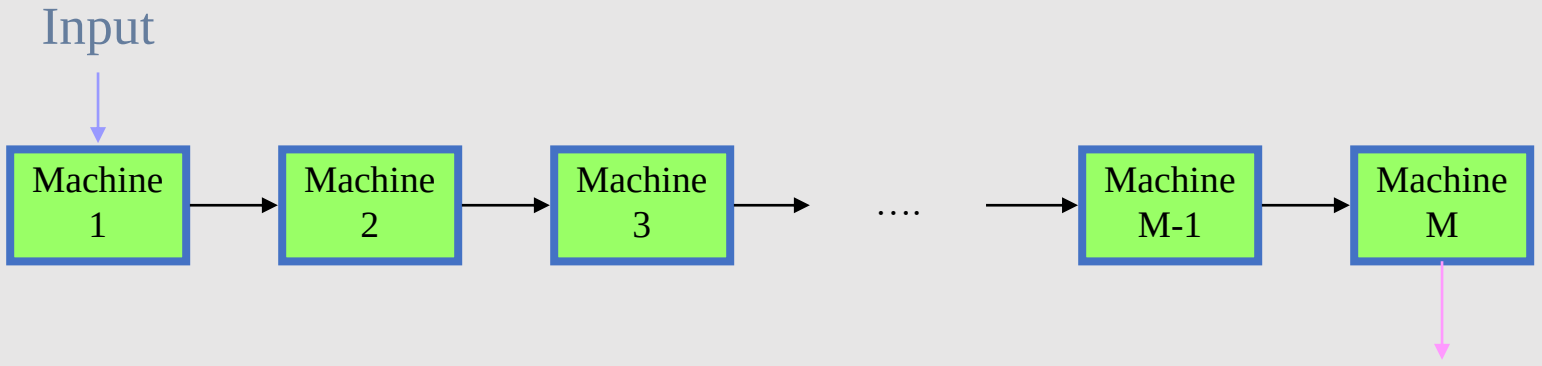
https://goo.gl/images/o8ME5m



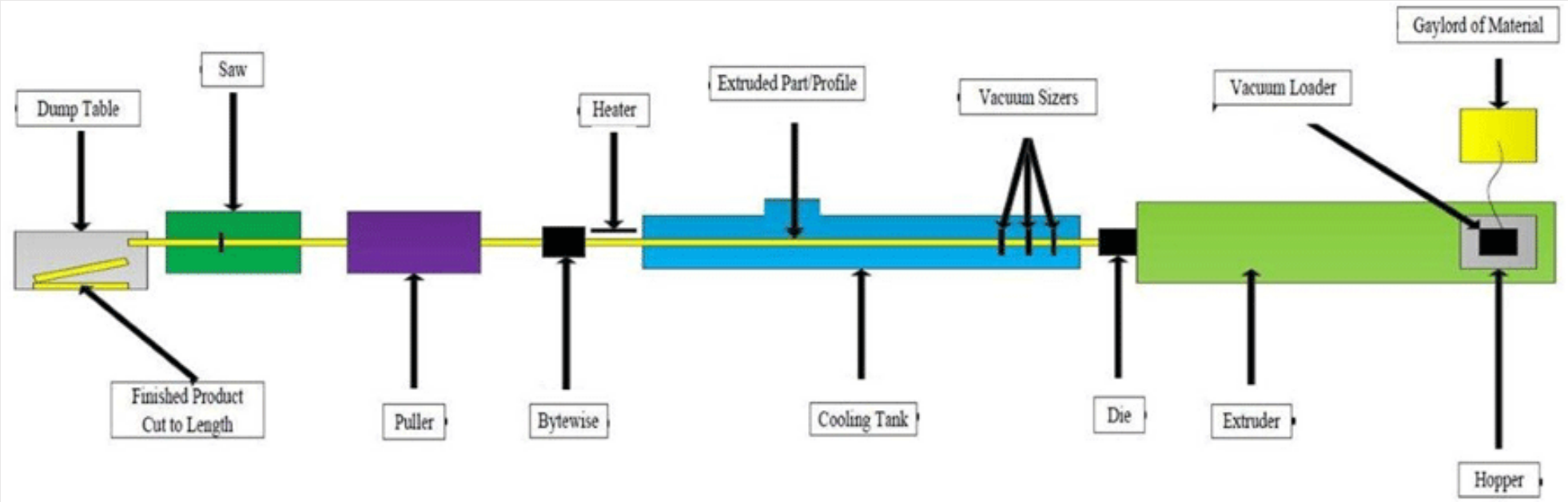
Paralel makineli çizelgeleme ortamı

Akış Tipi Üretim (Flow-shop)

İşlerin rotaları aynıdır (akış tek bir yöndedir).



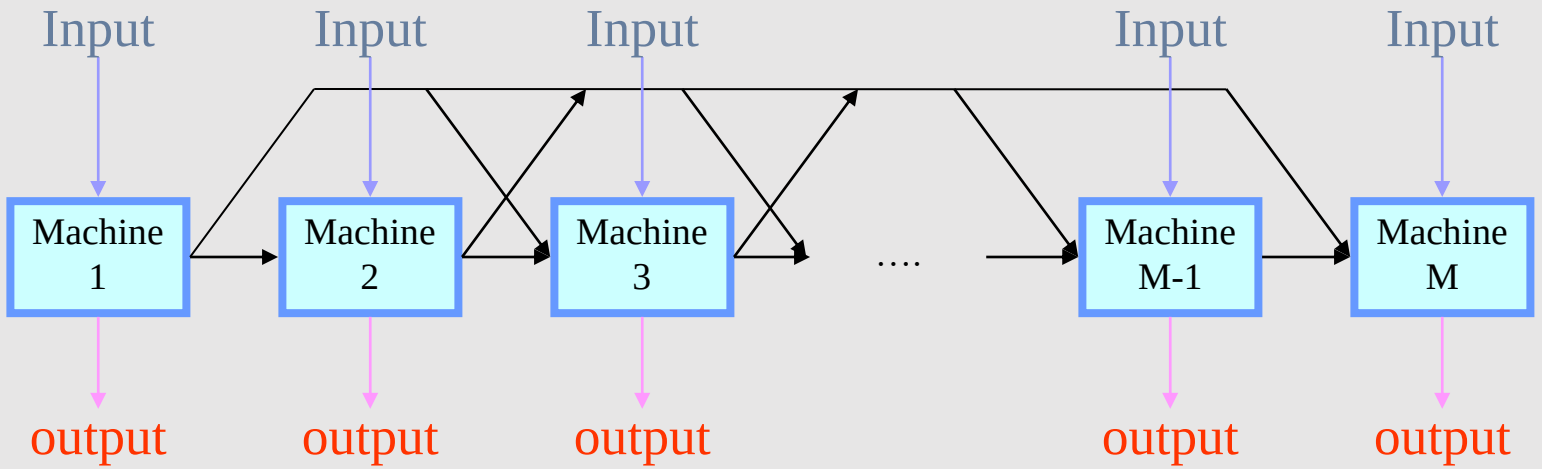
PVC Extrusion



https://goo.gl/images/nrspNG

Atölye Tipi Üretim (Job-shop)

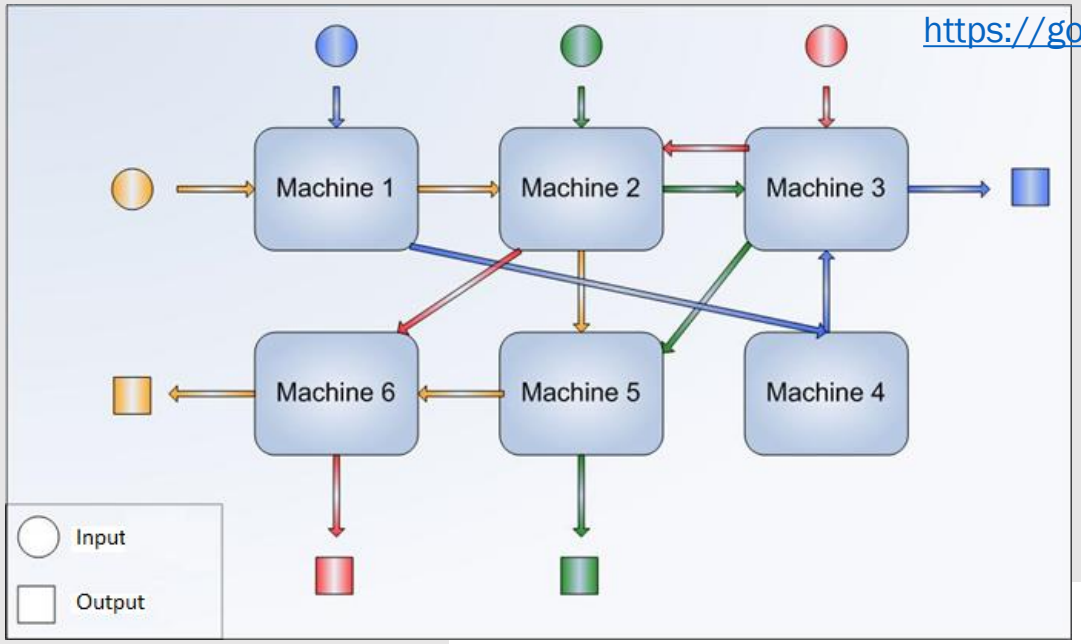
Her iş farklı bir rota izler.



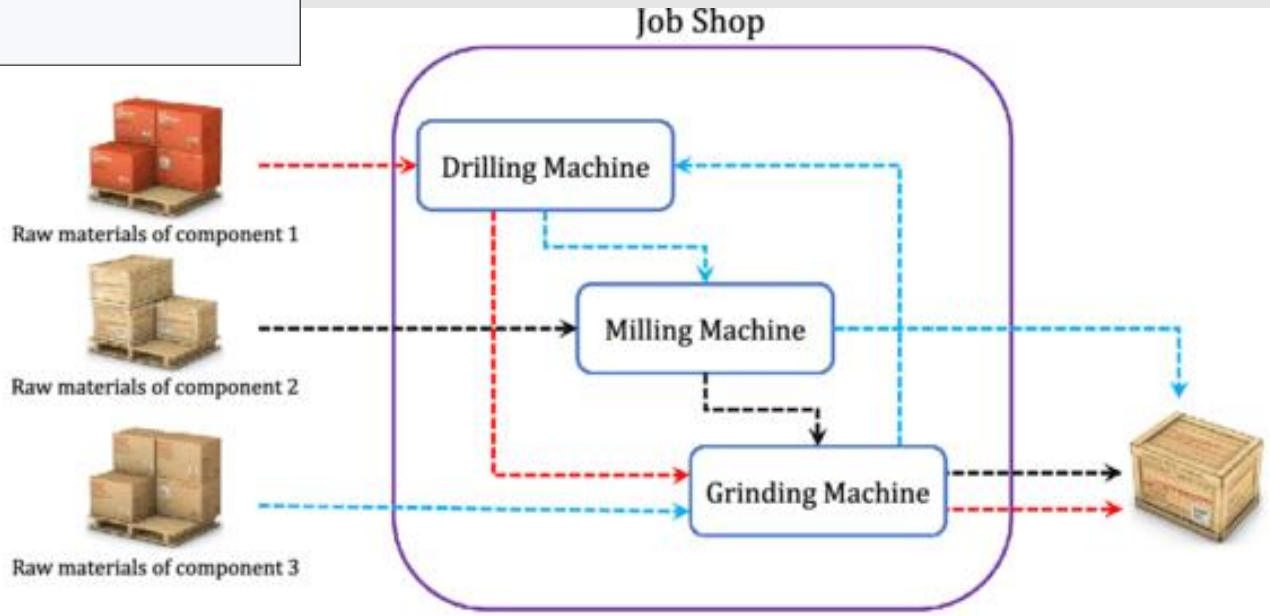
https://goo.gl/images/aVdqM8

Atölye Tipi Üretim (Job-shop)

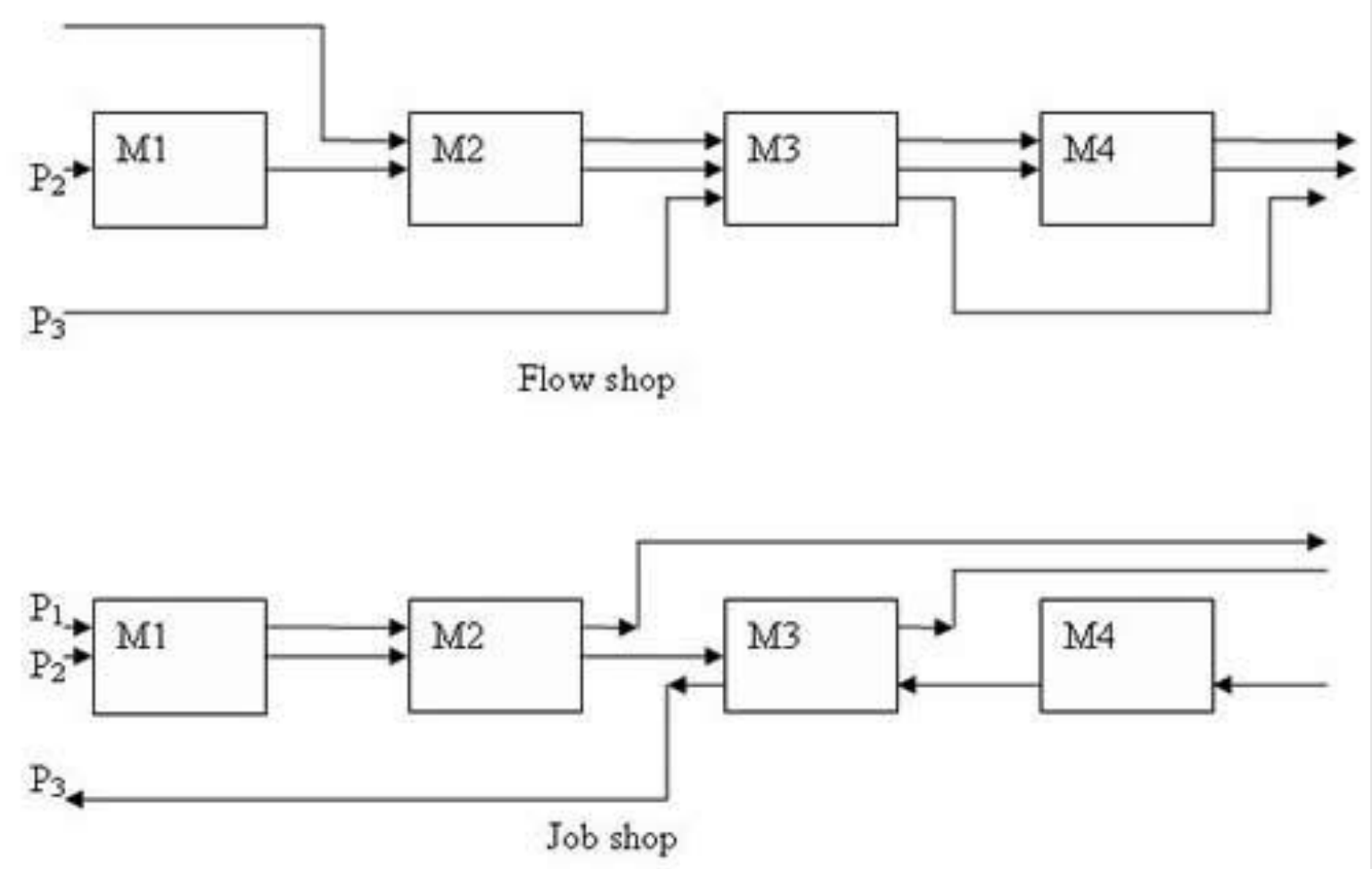
https://goo.gl/images/jW3e1J



https://goo.gl/images/5ur3p9



Flow-Shop vs. Job-Shop



Statik/Dinamik Çizelgeleme

Statik Çizelgeleme

- Sonradan iş gelişi yok
- Belirsizlik yok

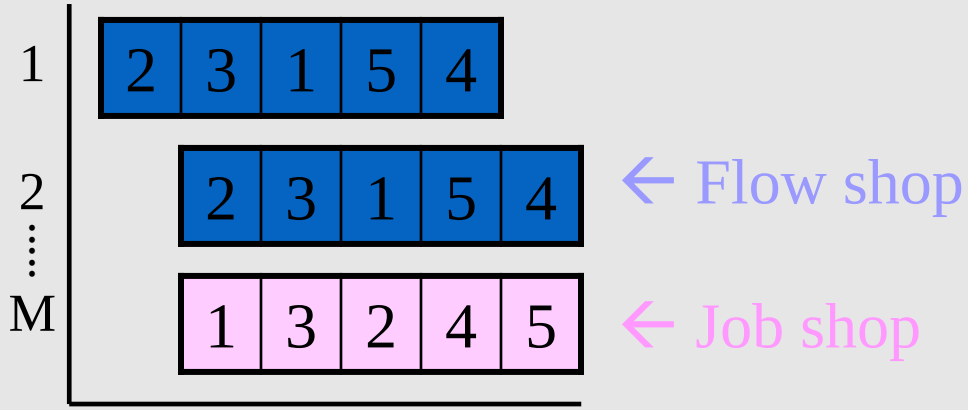
Dinamik Çizelgeleme

- Rastsal iş gelişleri
- İşlem sürelerinde değişkenlik
- Makine arızaları
- Teslim tarihinde değişiklik
- Sipariş iptali
  vb. tahmin edilemeyen olaylar

Çizelgeleme probleminin zorluğu

- M adet farklı makine mevcut
- Her iş tüm makinelerde işlem görmekte yani M adet operasyona sahip
- N adet iş sistemde operasyon ihtiyacı var
- İş *i'nin operasyonları:*
  (i,1) - (i,2) - (i,3) - ... - (i,m)
- Eğer iş *i* için makine *k* da işlem yoksa,

  P(i,j) = 0



n! - flow shop permutasyon çizelge

n!.n! …….n! - Job shop

$(n!)^m$ or $\frac{(n!)^m}{k}$ k : constraint (∵ routing problem)

Varsayımlar

- ✓ Sıralanacak işler başlangıçta tümüyle bilinmektedir.
- ✓ İşlem süreleri belirlidir ve uygulandıkları sıradan bağımsızdır.
- ✓ Bir işlem tamamlanmadıkça o tezgâhta başka bir işlem başlatılamaz.
- ✓ Bir tezgahta başlatılan bir işlem, bitinceye dek aralıksız sürdürülür.
- ✓ Tezgahlarda arıza vb. bir nedenle durma söz konusu değildir.

Varsayımlar

- ✓ Tezgâhlar arası taşıma süreleri, işlem süreleri içinde düşünülür ve bu süreler taşınma şeklinden bağımsızdır.
- ✓ Her işin rotası belirlidir.
- ✓ İşlerin bitirilmesi için belirlenmiş olan tarih, sonradan değiştirilemez.
- ✓ Her tip tezgahtan salt bir tane vardır.
- ✓ Her bir iş, bir tezgahta salt bir kez işlenebilir.
- ✓ Hiç bir rastgelelik yoktur.

Performans Ölçütleri

Tamamlanma zamanına dayalı performans ölçütleri

- En büyük akış süresi/zamanı
- Ortalama akış süresi/zamanı
- **Toplam akış zamanı (Flow Time)**
- **En büyük tamamlanma zamanı (Yayılma süresi - Makespan)**
- Ortalama tamamlanma zamanı
- Toplam ağırlıklı tamamlanma zamanı

Teslim zamanına dayalı performans ölçütleri

- En büyük gecikme
- Ortalama gecikme
- En büyük teslim gecikme süresi
- Ortalama teslim gecikme süresi
- Toplam ağırlıklı gecikme süresi
- Geciken iş sayısı

Stok ve makine kullanım maliyetine dayalı performans ölçütleri

- İşlenmek üzere makinelerde bekleyen ortalama iş sayısı
- Tamamlanmamış ortalama iş sayısı
- Tamamlanmış ortalama iş sayısı
- Süreçteki ortalama iş sayısı
- Ortalama makine boş zamanı
- En büyük makine boş zamanı

İş BEKLER

-

Makine BOŞTA KALIR

Performans Ölçütleri

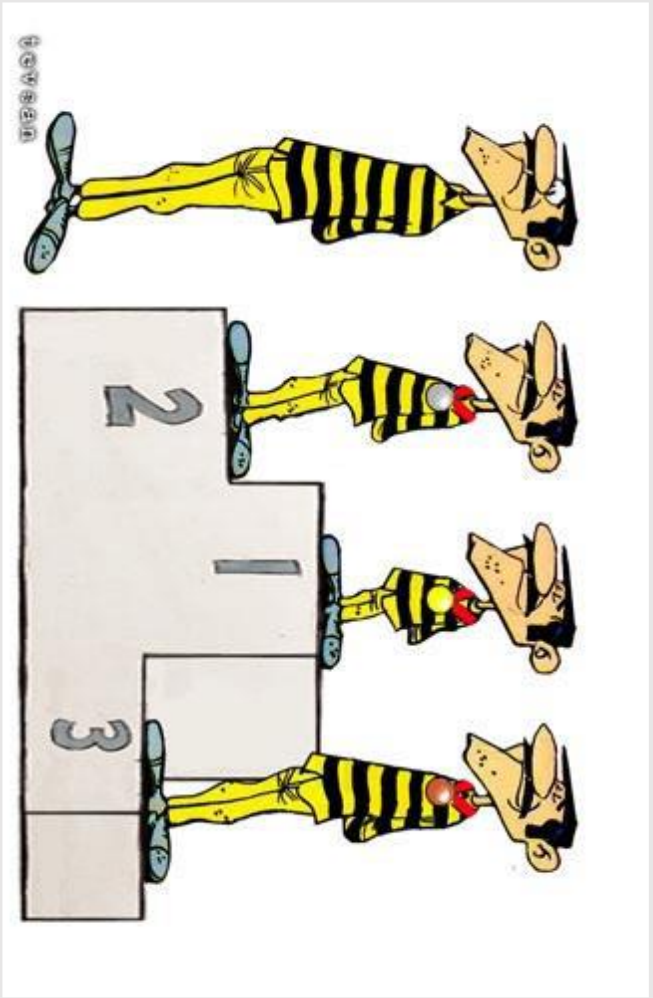
- **Akış süresi:** Bir iş ya da hizmetin iş istasyonunda harcadığı zamana akış süresi denir. Bu süre iş istasyonunda gerekli tüm hazırlıkların süreleri, işlem süresi, üretimler arasında geçiş için harcanan süre, makine bozulmalarında harcanan süre, girdi tükenmesinden kaynaklı gecikme süreleri gibi sürelerin toplamıdır.

$$\text{Akış süresi} = \text{işin tamamlanma süresi} + \text{işin istasyona gelmesinden beri geçen süre}$$

Performans Ölçütleri

- Literatüre bakıldığında yapılan çalışmaların genellikle;
  - *toplam veya ortalama tamamlanma zamanı,*
  - *toplam gecikme,*
  - *geciken iş sayısı,*
  - *toplam erken-geç tamamlanma maliyetinin*

  minimize edilmesini amaçladığı görülmektedir.



Notasyon

Processing time (p_j)	The amount of processing required by job j
Release date (r_j)	The time at which job j is available for processing
Due date (d_j)	The time at which the processing of job j is due to be completed
Completion time (C_j)	The time at which the processing of job j is finished

Quantitative measures for evaluating schedules are usually functions of job completion times. Two important quantities are:

Flowtime (F_j)	The time job j spends in the system: $F_j = C_j - r_j$
Lateness (L_j)	The amount of time by which the completion time of job j exceeds its due date: $L_j = C_j - d_j$
Tardiness (T_j)	The lateness of job j if it fails to meet its due date, or zero otherwise: $T_j = \max\{0, L_j\}$

Notasyon

Yukarıdaki terimler arasında geçen;

- L_j; sipârişin teslim edilmesi plânlanan d_j zamanından önce veya sonra bitmesi durumunu belirten bir ölçüdür.
  - *($L_j<0$) olması işin erken bittiğini gösterir, yâni bir gecikme yoktur ve iyi bir servis durumu sözkonusudur.*
  - *($L_j>0$) olması ise, gelişmenin kötü olduğunu, yâni bir gecikmenin bulunduğunu gösterir.*
- Diğer bir deyişle L_j, gecikmenin gelişme seyrini gösteren bir büyüklük olarak nitelendirilebilir.
- Performansın değerlendirilmesi açısından gecikmenin değerlendirilmesi ve mâliyete bağlanması daha uygundur.
- Bu bakımdan pozitif değere sahip olan gecikme önem taşır ve L_j yerine T_j değişkeni kullanılır.

Notasyon

- ✓ Terimler, tezgâh sayısı birden fazla olduğu zaman değişiklik gösterir ve kullanılan her tezgâh için ayrı ayrı tanımlanır.

- ✓ Örneğin P_{kj}; k tezgâhında yapılacak j işinin işlem süresini gösterir.
  C_{kj}; k tezgâhında yapılan j işinin tamamlanma zaman noktasini gösterir.

- ✓ Diğer değişkenler de benzer şekilde düşünülür ve hesaplamalar da bu yeni duruma göre, yâni tüm tezgâhlar düşünülerek yapılır.

- ✓ m; kullanılan tezgah sayısı olmak üzere, örneğin yayılma süresi şu şekilde hesaplanır:

  $M = Enb \{C_{kj} \mid (j=1,2,\ldots,n), (k=1,2,\ldots, m)\}$

Notasyon

Total flowtime: $F = \sum_{j=1}^{n} F_j$

Total tardiness: $T = \sum_{j=1}^{n} T_j$

Maximum flowtime: $F_{\max} = \max_{1 \le j \le n} \{F_j\}$

Maximum tardiness: $T_{\max} = \max_{1 \le j \le n} \{T_j\}$

Number of tardy jobs, or the total unit penalty: $U = \sum_{j=1}^{n} \delta(T_j)$,

where $\delta(x) = 1$ if $x > 0$ and $\delta(x) = 0$ otherwise

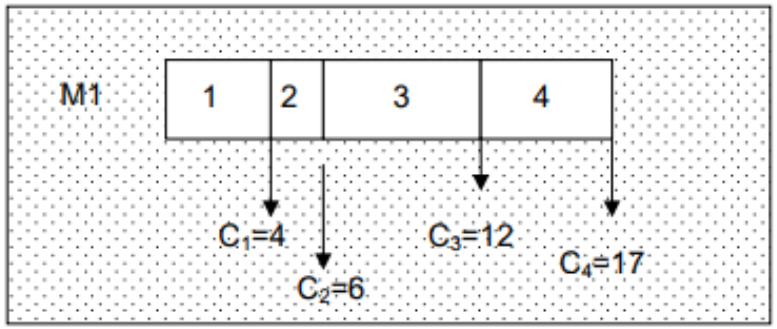
Maximum completion time: $C_{\max} = \max_{1 \le j \le n} \{C_j\}$

The following table contains data pertaining to $1||\bar{F}$ problem.

Job (j)	1	2	3	4
p_j	4	2	6	5
dj	6	6	10	18

i) Numerical Order Sequence (1-2-3-4)

Using the numerical or natural order sequence, the schedule is shown using Gantt chart below.



Gantt chart for numerical order sequence.

The waiting times for all the jobs is computed as shown in the following table

Job (j)	p_j	S_j	C_j	W_j	F_j
1	4	0	4	0	4
2	2	4	6	4	6
3	6	6	12	6	12
4	5	12	17	12	17

Problem gösterimi

Klasik Gösterim:

n / m / A / B

- n: İş sayısı
- m: Makine sayısı
- A: Atölye tipi vb (F/P...)
- B: Çalışma ölçütü (Performans kriteri)

Modern Gösterim:

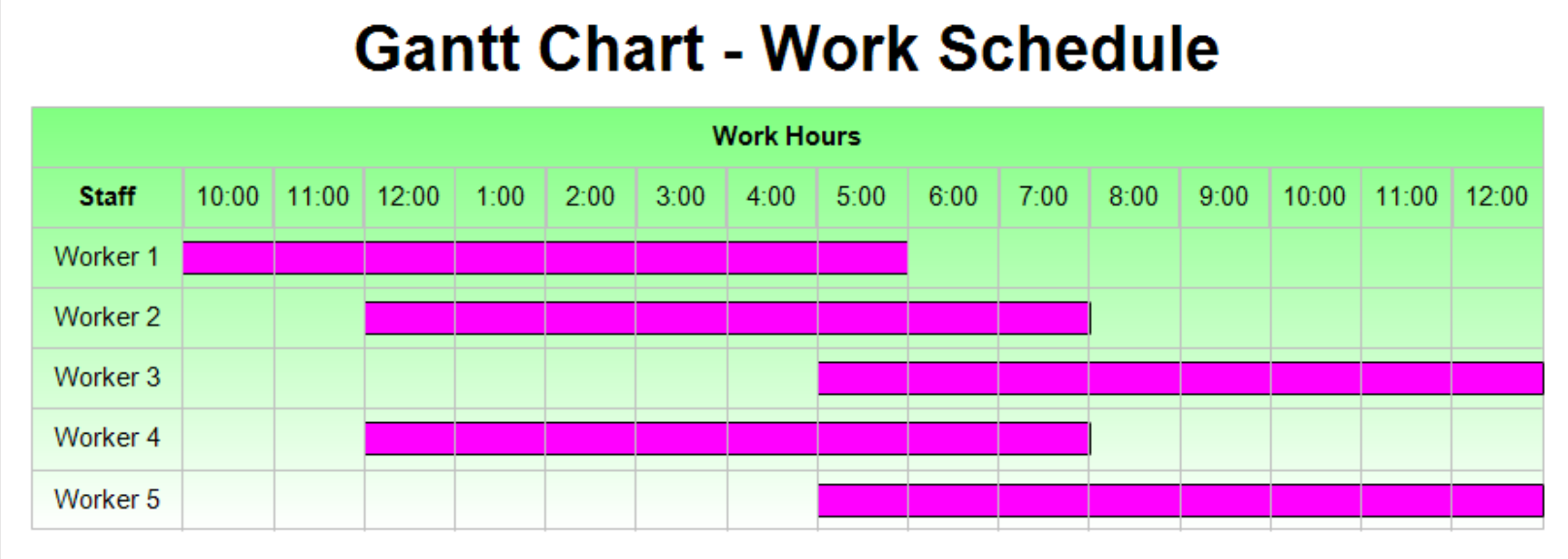
$$FFs \mid rj \mid \sum WjTj$$

Bu gösterimde iş ve makine sayıları yoktur. Hepsi çalışma esaslı da olabilmektedir.

GANTT Diyagramı
Tek Makine Çizelgeleme
Öncelik Kuralları
WSPT

Gantt Diyagramı ile Gösterim

- Bu diyagramda her tezgahta yapılacak işler, zamanın bir işlevi şeklinde gösterilir.
- Yatay çizgilerin tezgahları gösterdiği bu diyagram, varolan durumu açıkça gözönüne serdiği için yararlıdır.
- Ancak Gantt diyagramları, varolan kötü bir durumun iyileştirilebilmesi için herhangi bir çözüm üretme özelliğine sahip değillerdir.



https://www.rff.com/workschedule.php

Gantt Diyagramı ile Gösterim

- Beş iş (i), iki tezgah (T1 ve T2) üzerinde 3-2-4-5-1 sırasıyla yapılmakta, her iş önce T1'de, ardından T2'de gerçekleştirilmektedir.
- İş süreleri (zaman birimi) aşağıdaki tabloda verildiği gibidir:

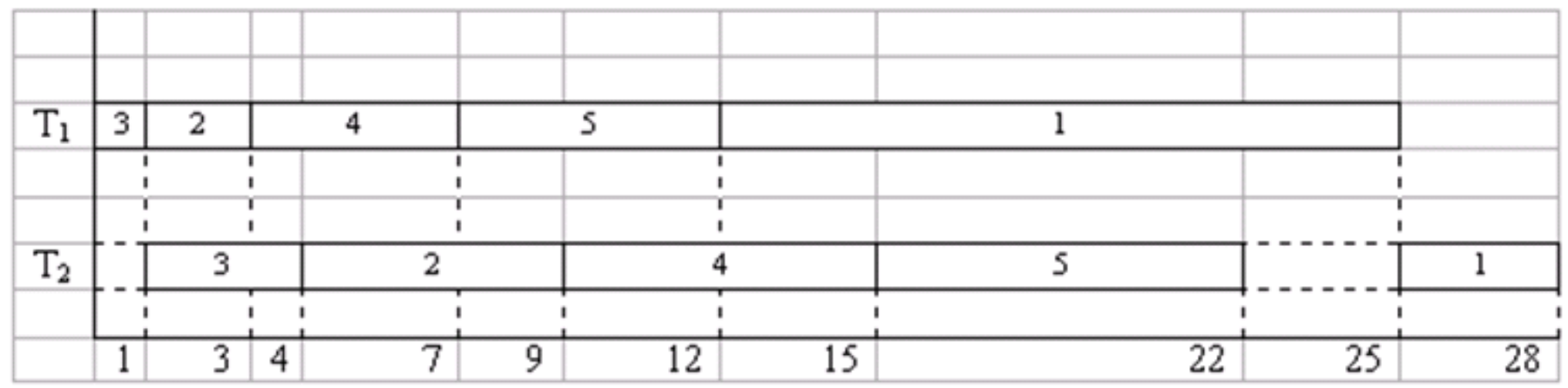
İş	T1	T2
1	13	3
2	2	5
3	1	3
4	4	6
5	5	7

Gantt Diyagramı ile Gösterim

- İş sırası **‘3-2-4-5-1’** olarak düşünüldüğünde Gantt diyagramı (iş çizelgesi) izleyen şekildeki gibi olacaktır
  (x ekseni zaman (zb), y ekseni tezgah adı, kutuların içine yazılan değerler ise iş numaralarıdır).

İş	T1	T2
1	13	3
2	2	5
3	1	3
4	4	6
5	5	7

Gantt diyagramı



Gantt Diyagramı ile Gösterim

Bu diyagram incelendiği zaman;

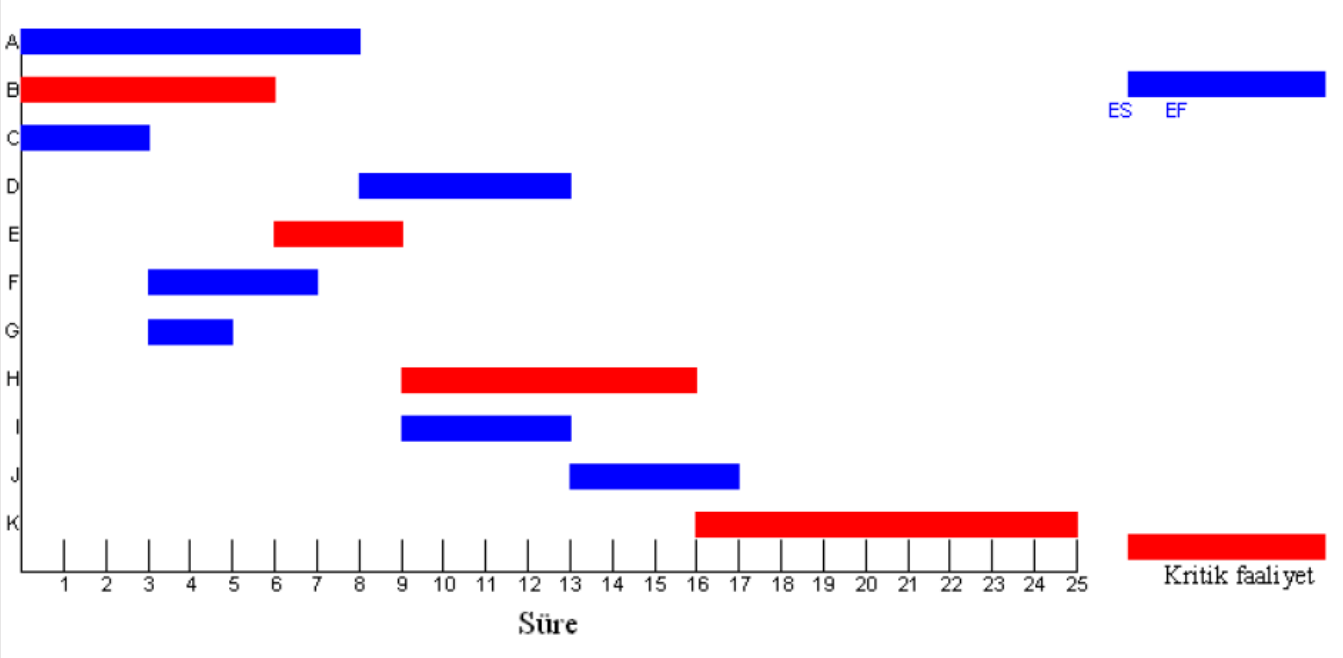
- Tamamlanma zamanının 28 zb olduğu,
- T1'in boş kalmadığı,
- T2'nin toplam 4 zb boş kaldığı,
- 2, 4 ve 5 no'lu işlerin T2'de toplam 6 zb beklediği görülmektedir.

- Burada birinci tezgahtaki en son işin tamamlanma zamanı 25, ikinci tezgahtaki en son işin tamamlanma zamanı ise 28 olduğuna göre yayılma süresi:

$$M = Max \{25, 28\} = 28 \text{ zb'dir.}$$

Gantt Diyagramı ile Gösterim

Farklı Gösterimler



	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
Torna	A101				A72				A105	
Freze	A72				A105					
Taşlama	X87				A101		x63			
Boyama	B21		X87				A101		A72	

Öncelik Kuralları (Priority Rules)

İşlerin iş merkezlerinde hangi sıra ile yapılması gerektiğini belirler.

- ✓ **İlk Gelene Ilk Hizmet Verme (*FCFS: First come, first served*)**
- ✓ **En Kısa İşlem Süresi (*SPT: Shortest processing time*)**
- ✓ **En Erken Teslim Süresi (*EDD: Earliest due date*)**
- ✓ **En Uzun İşlem Süresi (*LPT: Longest processing time*)**
- ✓ **Kritik Oran *(CR: Critical Rate)***
- ✓ Son Gelen İlk Yapılır (LIFO)
- ✓ Rastgele Seçim
- ✓ Akış Süresi En Uzun Olan İlk Yapılır
- ✓ Bekleme Süresi En Uzun Olan İlk Yapılır
- ✓ Kalan İşlem Süresi En Uzun Olan İlk Yapılır
- ✓ Kalan İşlem Sayısı En Fazla Olan İlk Yapılır
- ✓ En Yüksek Mâliyetli Olan İlk Yapılır

Öncelik Kuralları

- **İlk gelene ilk hizmet verme (FCFS)** benimsenmiş ise işler üretim merkezine geliş sırasına göre makinelere atanacaktır.
- **En Kısa İşlem Süresi (SPT)** yönteminde en kısa süreye sahip olan işin ilk olarak atanması söz konusu iken **En Uzun İşlem Süresi (LPT)** yönteminde ise uzun süreli işe öncelik tanınmaktadır.
- **En Erken Teslim Tarihi (EDD)'ne** göre yapılan sıralamada teslim tarihi erken olan işin ilk önce yapılması söz konusudur.
- **Kritik Oran (CR)** yönteminde, teslim için geri kalan zamanın işlem süresine oranı en küçük olan iş önceliklenir.

Örnek 1

Örnek 1: Yukarıda verilen FCFS, SPT, LPT ve EDD kurallarını aşağıdaki işler için uygulayalım (r_j tüm işler için "0"dır):

Job	***Job Work (Processing) Time (Days)***	***Job Due Date (Days)***
A	*6*	*8*
B	*2*	*6*
C	*8*	*18*
D	*3*	*15*
E	*9*	*23*

Örnek 1

FCFS: Sequence A-B-C-D-E

Job Sequence	***Job Work (Processing) Time***	***Flow Time***	***Job Due Date***	***Job Lateness***
A	*6*	*6*	*8*	*0*
B	*2*	*8*	*6*	*2*
C	*8*	*16*	*18*	*0*
D	*3*	*19*	*15*	*4*
E	*9*	*28*	*23*	*5*
Total	*28*	*77*	-	*11*

Örnek 1

FCFS: Sequence A-B-C-D-E

$$\text{Average completion time} = \frac{\text{Sum of total flow time}}{\text{Number of jobs}} = 77/5 = 15.4 \text{ days}$$ (Fort)

$$\text{Utilization} = \frac{\text{Total job work time}}{\text{Sum of total flow time}} = 28/77 = 36.4\%$$

$$\text{Average number of jobs in the system} = \frac{\text{Sum of total flow time}}{\text{Total job work time}} = 77/28 = 2.75 \text{ jobs}$$

$$\text{Average job lateness} = \frac{\text{Total late days}}{\text{Number of jobs}} = 11/5 = 2.2 \text{ days}$$

Örnek 1

SPT: Sequence B-D-A-C-E

Job Sequence	*Job Work (Processing) Time*	*Flow Time*	*Job Due Date*	*Job Lateness*
B	*2*	*2*	*6*	*0*
D	*3*	*5*	*15*	*0*
A	*6*	*11*	*8*	*3*
C	*8*	*19*	*18*	*1*
E	*9*	*28*	*23*	*5*
Total	*28*	*65*	-	*9*

Örnek 1

SPT: Sequence B-D-A-C-E

$$\text{Average completion time} = \frac{\text{Sum of total flow time}}{\text{Number of jobs}} = 65/5 = 13 \text{ days}$$ (Fort)

$$\text{Utilization} = \frac{\text{Total job work time}}{\text{Sum of total flow time}} = 28/65 = 43.1\%$$

$$\text{Average number of jobs in the system} = \frac{\text{Sum of total flow time}}{\text{Total job work time}} = 65/28 = 2.32 \text{ jobs}$$

$$\text{Average job lateness} = \frac{\text{Total late days}}{\text{Number of jobs}} = 9/5 = 1.8 \text{ days}$$

Örnek 1

EDD: Sequence B-A-D-C-E

Job Sequence	*Job Work (Processing) Time*	*Flow Time*	*Job Due Date*	*Job Lateness*
B	*2*	*2*	*6*	*0*
A	*6*	*8*	*8*	*0*
D	*3*	*11*	*15*	*0*
C	*8*	*19*	*18*	*1*
E	*9*	*28*	*23*	*5*
Total	*28*	*68*	-	*6*

Örnek 1

EDD: Sequence B-A-D-C-E

$$\text{Average completion time} = \frac{\text{Sum of total flow time}}{\text{Number of jobs}} = 68/5 = 13.6 \text{ days} \quad \text{(Fort)}$$

$$\text{Utilization} = \frac{\text{Total job work time}}{\text{Sum of total flow time}} = 28/68 = 41.2\%$$

$$\text{Average number of jobs in the system} = \frac{\text{Sum of total flow time}}{\text{Total job work time}} = 68/28 = 2.43 \text{ jobs}$$

$$\text{Average job lateness} = \frac{\text{Total late days}}{\text{Number of jobs}} = 6/5 = 1.2 \text{ days}$$

Örnek 1

LPT: Sequence E-C-A-D-B

Job Sequence	***Job Work (Processing) Time***	***Flow Time***	***Job Due Date***	***Job Lateness***
E	*9*	*9*	*23*	*0*
C	*8*	*17*	*18*	*0*
A	*6*	*23*	*8*	*15*
D	*3*	*26*	*15*	*11*
B	*2*	*28*	*6*	*22*
Total	*28*	*103*	-	*48*

Örnek 1

LPT: Sequence E-C-A-D-B

$$\text{Average completion time} = \frac{\text{Sum of total flow time}}{\text{Number of jobs}} = 103/5 = 20.6 \text{ days} \quad \text{(Fort)}$$

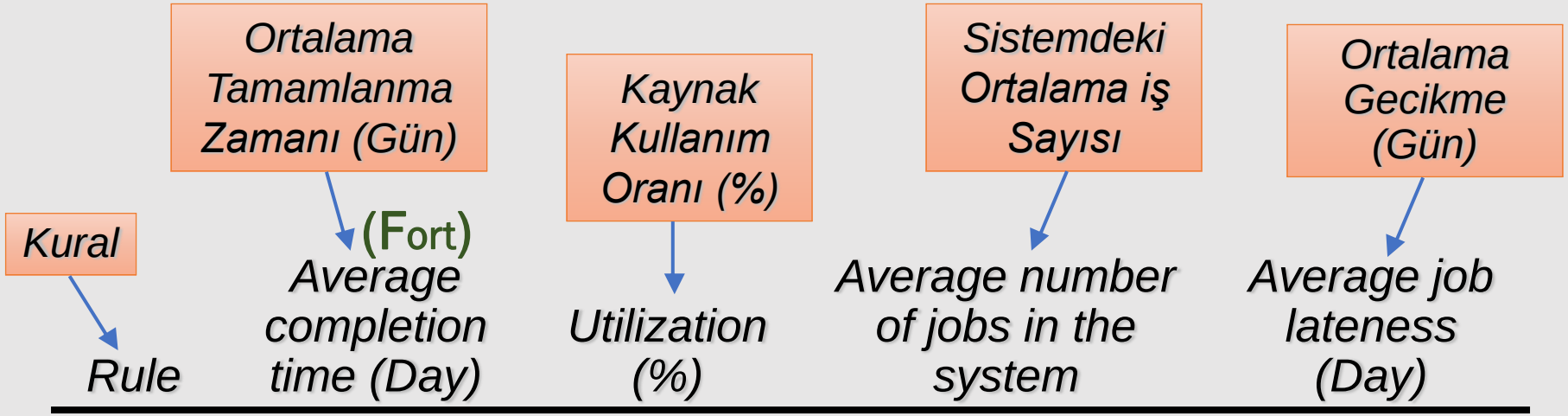
$$\text{Utilization} = \frac{\text{Total job work time}}{\text{Sum of total flow time}} = 28/103 = 27.2\%$$

$$\text{Average number of jobs in the system} = \frac{\text{Sum of total flow time}}{\text{Total job work time}} = 103/28 = 3.68 \text{ jobs}$$

$$\text{Average job lateness} = \frac{\text{Total late days}}{\text{Number of jobs}} = 48/5 = 9.6 \text{ days}$$

Örnek 1

Sonuçları özetlersek:



Rule	Average completion time (Day)	Utilization (%)	Average number of jobs in the system	Average job lateness (Day)
FCFS	15.4	36.4	2.75	2.2
SPT	**13.0***	**43.1***	**2.32***	1.8
EDD	13.6	41.2	2.43	**1.2***
LPT	20.6	27.2	3.68	9.6

Örnek 1

Sonuç:

- **Tüm kriterlerde en iyi sonucu verecek tek bir sıralama kuralı yoktur.**
- **FCFS tüm müşterilere sadece eşit davranır.**
- **SPT, akış zamanını azaltmada ve sistemdeki iş sayısını azaltmada iyidir.**
- **SPT, işlem süresi uzun olanları sona attığı için müşteri tatmini açısından dezavantaj sağlar.**
- **EDD, geciken iş sayısını azaltır.**

Öncelik Kuralları: Kritik Oran (CR)

- Kritik oran indeksi her bir iş için ayrı hesaplanır.
- Indeks değeri küçük olan işler ilk olacak şekilde sıralanır.
- Ortalama geciken iş sayısı performans kriterine göre iyi sonuç verir.
- Eğer CR < 1 ise, siparişin programın gerisinde olduğu anlaşılır

$$CR = \frac{\text{Time remaining}}{\text{Workdays remaining}} = \frac{\text{Due date - Today's date}}{\text{Work (lead) time remaining}}$$

$$CR = \frac{\text{Arta Kalan Zaman}}{\text{Arta Kalan İş}}$$

Arta Kalan Zaman

Örnek (Bugün: 25. Gün):		
İş	***Teslim Zamanı***	***Arta Kalan İş (Gün)***
A	*30. Gün*	*4*
B	*28. Gün*	*5*
C	*27. gün*	*2*

Öncelik Kuralları: Kritik Oran (CR)

Currently: Day 25

İş	Teslim Zamanı	Arta Kalan İş (Gün)	Kritik Oran (CR)	Öncelik Sırası
A	30. Gün	4	(30 - 25)/4 = 1.25	3
B	28. Gün	5	(28 - 25)/5 = 0.60	1
C	27. gün	2	(27 - 25)/2 = 1.00	2

With CR < 1, Job B is late. Job C is just on schedule and Job A has some slack time.

Ödev: Örnek-1'i Kritik Oran yöntemi ile çözüp elde edilen sonucu FCFS, SPT, LPT ve EDD kurallarıyla elde edilen sonuçlarla kıyaslayınız.

Örnek 2

Bir PVC doğrama üretim atölyesinin elindeki siparişlere ilişkin veriler aşağıdaki tablodadır. Buna göre her öncelik kuralı için siparişleri sıralayıp oluşturulan programın performansını değerlendiriniz.

Sipariş	Siparişin alınmasından beri geçen süre (gün)	İşlem süresi (gün)	Teslim tarihine kalan süre (gün)
A	15	25	29
B	12	16	27
C	5	14	68
D	10	10	48
E	0	12	80
F	7	16	47

Örnek 2

-İlk gelen ilk hizmet alır-

Sipariş	Siparişin alınmasından beri geçen süre (gün)	İşlem süresi (gün)	Teslim tarihine kalan süre (gün)	Başlangıç zamanı (gün)	Bitiş zamanı (gün)	Gecikme süresi (gün)	Akış süresi (gün)
A	15	25	29	0	25	0	40
B	12	16	27	25	41	14	53
D	10	10	48	41	51	3	61
F	7	16	47	51	67	20	74
C	5	14	68	67	81	13	86
E	0	12	80	81	93	13	93

$$Ortalama\ gecikme = \frac{0+14+3+20+13+13}{6} = 10{,}5\ \text{gün}$$

$$Ortalama\ akış\ süresi = \frac{40+53+61+74+86+93}{6} = 67{,}8\ \text{gün}$$

Örnek 2

- Teslim tarihi en erken olan ilk hizmet alır -

Sipariş	Siparişin alınmasından beri geçen süre (gün)	İşlem süresi (gün)	Teslim tarihine kalan süre (gün)	Başlangıç zamanı (gün)	Bitiş zamanı (gün)	Gecikme süresi (gün)	Akış süresi (gün)
B	12	16	27	0	16	0	28
A	15	25	29	16	41	12	56
F	7	16	47	41	57	10	64
D	10	10	48	57	67	19	77
C	5	14	68	67	81	13	86
E	0	12	80	81	93	12	93

$$Ortalama\ gecikme = \frac{0+12+10+19+13+12}{6} = 11{,}2\ g\ddot{u}n$$

$$Ortalama\ akış\ süresi = \frac{28+56+64+77+86+93}{6} = 67{,}3\ g\ddot{u}n$$

Örnek 2

- İşlem süresi en kısa olan ilk hizmet alır -

Sipariş	Siparişin alınmasından beri geçen süre (gün)	İşlem süresi (gün)	Teslim tarihine kalan süre (gün)	Başlangıç zamanı (gün)	Bitiş zamanı (gün)	Gecikme süresi (gün)	Akış süresi (gün)
D	10	10	48	0	10	0	20
E	0	12	80	10	22	0	22
C	5	14	68	22	36	0	41
B	12	16	27	36	52	25	64
F	7	16	47	52	68	21	75
A	15	25	29	68	93	64	108

$$Ortalama\ gecikme = \frac{0+0+0+25+21+64}{6} = 18{,}3\ gün$$

$$Ortalama\ akış\ süresi = \frac{20+22+41+64+75+108}{6} = 55\ gün$$

Örnek 2

- İşlem süresi en uzun olan ilk hizmet alır -

Sipariş	Siparişin alınmasından beri geçen süre (gün)	İşlem süresi (gün)	Teslim tarihine kalan süre (gün)	Başlangıç zamanı (gün)	Bitiş zamanı (gün)	Gecikme süresi (gün)	Akış süresi (gün)
A	15	25	29	0	25	0	40
B	12	16	27	25	41	14	53
F	7	16	47	41	57	10	64
C	5	14	68	57	71	3	76
E	0	12	80	71	83	3	83
D	10	10	48	83	93	45	103

$$Ortalama\ gecikme = \frac{0+14+10+3+3+45}{6} = 12{,}5\ gün$$

$$Ortalama\ akış\ süresi = \frac{40+53+64+76+83+103}{6} = 69{,}8\ gün$$

Örnek 2

- En son gelen ilk hizmet alır -

Sipariş	Siparişin alınmasından beri geçen süre (gün)	İşlem süresi (gün)	Teslim tarihine kalan süre (gün)	Başlangıç zamanı (gün)	Bitiş zamanı (gün)	Gecikme süresi (gün)	Akış süresi (gün)
E	0	12	80	0	12	0	12
C	5	14	68	12	26	0	31
F	7	16	47	26	42	0	49
D	10	10	48	42	52	4	62
B	12	16	27	52	68	41	80
A	15	25	29	68	93	64	108

$$Ortalama\ gecikme = \frac{0+0+0+4+41+64}{6} = 18{,}2\ gün$$

$$Ortalama\ akış\ süresi = \frac{12+31+49+62+80+108}{6} = 57\ gün$$

Örnek 2

- Kalan boşluk süresi en kısa olan ilk hizmet alır -

Sipariş	Siparişin alınmasından beri geçen süre (gün)	İşlem süresi (gün)	Teslim tarihine kalan süre (gün)	Kalan boşluk süresi (gün)	Başlangıç zamanı (gün)	Bitiş zamanı (gün)	Gecikme süresi (gün)	Akış süresi (gün)
A	15	25	29	4	0	25	0	40
B	12	16	27	11	25	41	14	53
F	7	16	47	31	41	57	10	64
D	10	10	48	38	57	67	19	77
C	5	14	68	54	67	81	13	86
E	0	12	80	68	81	93	13	93

$$Ortalama\ gecikme = \frac{0+14+10+19+13+13}{6} = 11{,}5\ gün$$

$$Ortalama\ akış\ süresi = \frac{40+53+64+77+86+93}{6} = 68{,}8\ gün$$

Örnek 2

- Karşılaştırma-

Kural	Ortalama gecikme süresi (gün)	Ortalama akış süresi (gün)	Geciken faaliyet sayısı
İlk gelen	10,5	67,8	5
Teslim tarihi en erken olan	11,2	67,3	5
İşlem süresi en kısa olan	18,3	55,0	3
İşlem süresi en uzun olan	12,5	69,8	5
En son gelen	18,2	57,0	3
Kalan boşluk süresi en kısa olan	11,5	68,8	5

Örnek 3

Bir makine istasyonunda, üretim programının 12. gününde eldeki işler aşağıdaki tabloda görüldüğü gibidir. Kritik orana göre işlerin sıralaması nasıl olmalıdır?

İş	Teslim Zamanı (gün)	Arta kalan iş (gün)
A	18	8
B	16	2
C	20	12
D	15	3
E	21	8

Örnek 3

İş	Tamamlanma süresi (gün)	Teslim İçin Kalan Zaman (gün)	Arta kalan iş (gün)	Kritik Oran
A	18	18-12 = 6	8	6/8 = 0,75
B	16	16-12 = 4	2	4/2 = 2,00
C	20	20-12 = 8	12	8/12 = 0,67
D	15	15-12 = 3	3	3/3 = 1,00
E	21	21-12 = 9	8	9/8 = 1,12

İş sırası C, A, D, E, B olmalıdır. C ve A işleri hızlandırılmadığı sürece zamanında teslim edilemeyecektir.

Örnek 4

- İşlem görmesi gereken 5 işe (J1, J2, J3, J4, J5) ait bilgiler aşağıdaki gibidir:

İşler #	İşlem zamanları	Teslim Tarihi
1	11	61
2	29	45
3	31	31
4	1	33
5	2	32

http://www.d.umn.edu/~rlindek1/POM/Lecture_Slides/Scheduling%20of%20Jobs_Sset11.ppt

Örnek 4

FCFS kuralına göre yapılan çizelgeleme

Sıralama	Tamamlanma Zamanı	Teslim Tarihi	Gecikmeler
J1	11	61	0
J2	40	45	0
J3	71	31	40
J4	72	33	39
J5	74	32	42
Totals	268		121

- Ortalama Akış Zamanı: (268)/5 = 53.4
- Ortalama Gecikme: (121)/5 = 24.2
- # Geciken İşler : 3

Örnek 4

SPT kuralına göre yapılan çizelgeleme

Sıralama	Tamamlanma Zamanı	Teslim Tarihi	Gecikmeler
J4	1	61	0
J5	3	45	0
J1	14	31	0
J2	43	33	10
J3	74	32	42
Totals	135		52

- Ortalama Akış Zamanı: (135)/5 = 27.
- Ortalama Gecikme : (52)/5 = 10.4
- # Geciken İşler : 2

Örnek 4

EDD kuralına göre yapılan çizelgeleme

Sıralama	Tamamlanma Zamanı	Teslim Tarihi	Gecikmeler
J3	31	31	0
J5	33	32	1
J4	34	33	1
J2	63	45	18
J1	74	61	13
Totals	235		33

- Ortalama Akış Zamanı : (235)/5 = 47.
- Ortalama Gecikme : (33)/5 = 6.6
- # Geciken İşler : 4

Ağırlıklı Ortalama Tamamlanma Zamanı (WSPT)

- Bazı durumlarda, tüm işler aynı öneme sahip değildir ve bazı işlere öncelik tanınması gerekmektedir.
- Bu durumdaki işler için önemleri derecesinde ağırlık değerleri tanımlanır. İşin önemi arttıkça bu ağırlık değeri de artar.
- Teslim zamanının bilinmemesi durumunda da bu yaklaşım kullanılabilir.
- Ağırlıklandırma etmeni olarak üretim içi stok düzeyi ile orantılı olan elde bulundurma maliyetlerini almak uygun bir yoldur.
- Bu tip problemlerde n adet işin ağırlıklı ortalama tamamlanma zamanını enküçükleyecek sıralamanın saptanması amaçlanır.

Ağırlıklı Ortalama Tamamlanma Zamanı (WSPT)

Ağırlıklı ortalama tamamlanma zamanı (Cw*), AKİSÖ (Ağırlıklı Kısa İşlem Süresi Önce) kuralı ile enküçüklenir [WSPT (Weighted Shortest Processing Time) first]. Yani işleri artan ağırlıklı işlem sürelerine göre sıralarsak, elde edilen iş sırasına göre işlerin yapılması durumunda, Cw* en küçük değerini alır.

Bu durumda;

$$(P[1]/w[1]) \leq (P[2]/w[2]) \leq \ldots \leq (P[n]/w[n])$$

koşulunu sağlayan iş sırası, Cw* değerini enküçükleyen iş sırasıdır.

- Kısaca, daha önce anlattığımız SPT kuralı, WSPT kuralına dönüşmüş oluyor.*

Ağırlıklı Ortalama Tamamlanma Zamanı – Örnek 1

- Tek bir tezgahta 6 işin sıralaması yapılacaktır. İşlerin işlem süresi (Pi), önem derecesi (wi) ve işlem sürelerinin ağırlıklara oranı (Pi/wi) değerleri, Tabloda verilmiştir.

İş (i)	P_i	w_i	P_i/w_i
1	10	5	2,0
2	6	10	0,6
3	5	5	1,0
4	4	1	4,0
5	2	3	0,7
6	8	5	1,6

Ağırlıklı ortalama tamamlanma zamanını enküçükleyen iş sırası, WSPT kuralının genel ifadesi olan

$$(P[1]/w[1]) \leq (P[2]/w[2]) \leq \ldots \leq (P[n]/w[n])$$

ilişkisi kullanılarak (2-5-3-6-1-4) şeklinde saptanır.

Eğer tüm işler eşit ağırlıklı olsaydı [wi=1, ∀i için], en iyi iş sırası (5-4-3-2-6-1) olacaktı.

Ağırlıklı Ortalama Tamamlanma Zamanı – Örnek 2

Assume the weight assigned ω_j for each job j (importance or priority) and then solve the following 4-jobs problem using WSPT sequence. Also, compute the total weighted completion times.

Job(j)	p_j	ω_j	p_j/ω_j	d_j
1	4	8	0.5	8
2	2	7	0.285	12
3	6	3	2	11
4	5	15	0.333	10

Ağırlıklı Ortalama Tamamlanma Zamanı – Örnek 2

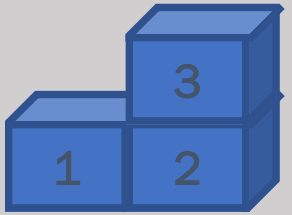
Solution: Jobs Sequence based on WSPT → (2-4-1-3)

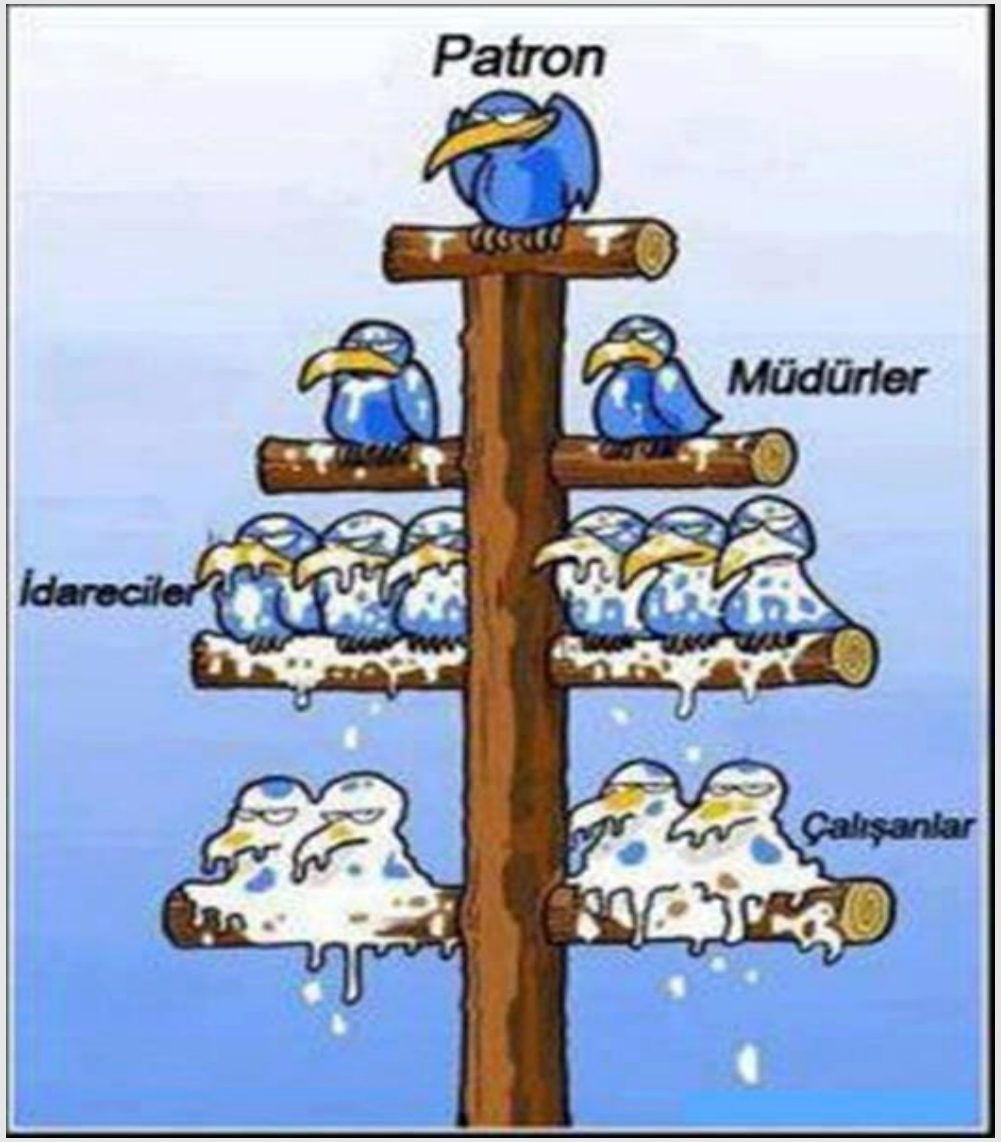
Job(j)	p_j	ω_j	S_j	C_j	d_j	W_j	L_j	T_j	$\omega_j C_j$
2	2	7	0	2	12	0	-10	0	14
4	5	15	2	7	10	2	-3	0	105
1	4	8	7	11	8	7	3	3	88
3	6	3	11	17	11	11	6	6	51

???

Hence, from the above table, the total weighted completion $\Sigma\omega_j C_j$ can be found which is ~~388~~. 258

Moore Algoritması
Öncelik Kısıtları
Lawler Algoritması
Öne Geçmeli Durum




Patron
Müdürler
İdareciler
Çalışanlar

Moore Algoritması

- **Amaç: Geciken iş sayısını minimize etmek.**
- 6/1//n probleminde, Moore Algoritmasına göre çözümü yapınız.

İŞ	1	2	3	4	5	6
Teslim Tarihi	15	6	9	23	20	30
İşlem zamanı	10	3	4	8	10	6

Moore Algoritması

- İlk adım olarak EDD ‘ ye göre sıralama yapılır.

İŞ	2	3	1	5	4	6
Teslim Tarihi	6	9	15	20	23	30
İşlem Zamanı	3	4	10	10	8	6
Bitiş Zamanı	3	7	17	…		

Moore Algoritması

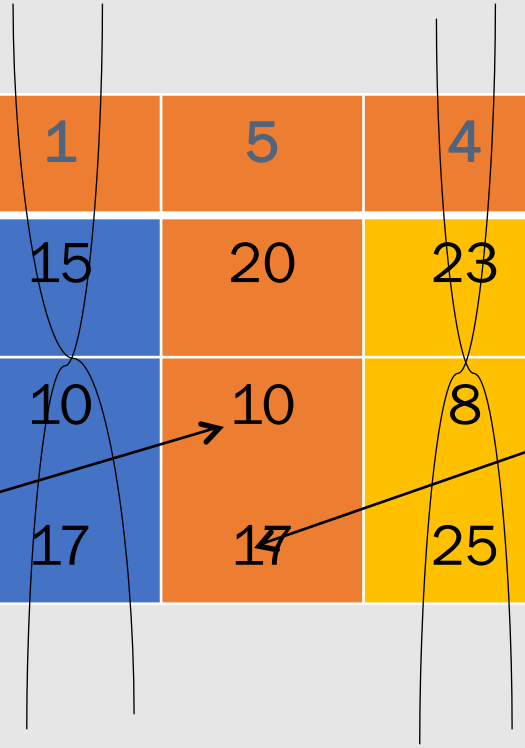
- Daha sonra 1 işi çıkartılır. 17, 15’ ten büyüktür. Yani teslim tarihini geçmiştir.

İŞ	2	3	1	5	4	6
Teslim Tarihi	6	9	15	20	23	30
İşlem Zamanı	3	4	10	10	8	6
Bitiş Zamanı	3	7	17	17	25	

Moore Algoritması

- 1 işi çıktı. İlerledik. 4 işine geldik. Bitim zamanı 25, teslim tarihi ise 23. 4 işi de sıradan çıkartılacaktır.

İŞ	2	3	1	5	4	6
Teslim Tarihi	6	9	15	20	23	30
İşlem Zamanı	3	4	10	10	8	6
Bitiş Zamanı	3	7	17	17	25	23



Nihai sıralama şöyle olacaktır: {2, 3, 5, 6, 1, 4} ya da {2, 3, 5, 6, 4, 1}

Moore Algoritması (Örnek 2)

- 10/1//nt problemini çözelim.

İş	1	2	3	4	5	6	7	8	9	10
D(i)	19	16	25	3	8	14	31	23	2	15
P(i)	5	3	1	2	4	4	2	1	1	4
EDD' ye göre sıralama > (9, 4,5,6,10,2,1,8,3,7)										
D(i)	2	3	8	14	15	16	19	23	25	31
P(i)	1	2	4	4	4	3	5	1	1	2

Moore Algoritması (Örnek 2)

İş	9	4	5	6	10	2	1	8	3	7
D(i)	2	3	8	14	15	16	19	23	25	31
P(i)	1	2	4	4	4	3	5	1	1	2
Bitiş	1	3	7	11	15	18				
Bitiş	1	3	7	11	15	***	20			
Bitiş	**1**	**3**	**7**	**11**	**15**	***	***	**16**	**17**	**19**

Sonuçta, 2 ve 1 işi çıkartılır. Sıralamalar şöyledir:

- {9-4-5-6-10-8-3-7-2-1} veya
- {9-4-5-6-10-8-3-7-1-2}

Öncelik Kısıtları (Lawler algoritması)

- Çok az firma tüm müşterilerinin eşit önemde olduğunu düşünür.
- Örneğin; işlenmek üzere sıra bekleyen bir dizi iş olsun. Varsayalım bu işlerden biri, özel önemdeki bir müşteri içindir.
- Bu durumda firmanın kararı, diğer işlerin programlanma sıraları her ne olursa olsun, bu müşteriye ait işe öncelikle başlanması ve ilk önce bitirilmesi olabilir.
- **Lawler algoritması**, böyle bir ortamda çizelgeleme yapılması için kullanılabilecek bir algoritmadır.
- Öncelik kısıtları, diğer işler bitirilmeden önce tamamlanması gereken işler olduğunda ortaya çıkmaktadır.

Lawler Algoritması

- Burada, öncelik kısıtlarının genelleştirilmesi önemlidir. Bu algoritma, bir işin maksimum maliyetini minimize eder.

$$\underset{i=1}{\overset{n}{Max}}\{\gamma_i(C_i)\} \qquad (4.5)$$

- Burada ki γ_i değeri zamanın parasal değerini veren bir katsayıdır.
- Böylece, zamanın parasal değerini veren bir katsayı ile bizler Cmax' i yani en büyük tamamlanma zamanını (yayılım zamanı) minimize etmeye çalışmaktayızdır.
- **1 | prec | fmax (Lawler's Algorithm)**

Lawler Algoritması

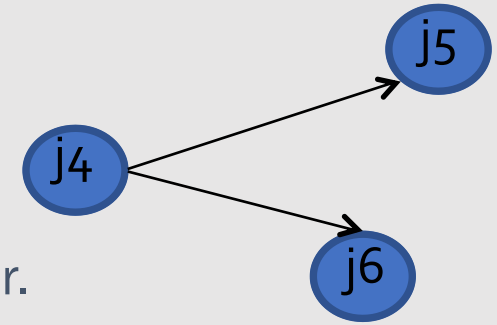
- Her bir $\gamma_i(C_i)$'de azalma olmadığı için, (4.5) numaralı performans ölçütü düzenlidir.
- Yine de $\gamma_i(C_i)$'nin spesifik tercihleri için daha çok bilinen formlar kullanılmaktadır.
- Eğer $\gamma_i(C_i)$=Ci–di=Li ise (4.5), Lmax ölçütünü verir.
- Eğer $\gamma_i(C_i)$ =max{Ci–di, 0} ise bu, Tmax ölçütünü verir.
- Lawler algoritması, değişik kısıtlı çizelgeleme problemlerinin çözümünde güçlü bir tekniktir.
- Algoritma öncelik kısıtlarına dayanır.
- Eş zamanlı olarak, bir kaynağa ulaşan bir dizi işleri (öncelik kısıtları ile) sıralar.
- Amaç, maksimum tardiness veya lateness' ı minimize etmektir.

Örnek: Lawler Algoritması

6/1/Lmax problemine ait öncelik baskıları (kısıtları) aşağıdaki gibidir:

- J1, J2' den önce gerçekleştirilmelidir.
- J2, J3' ten önce gerçekleştirilmelidir.
- J4; J5 ve J6'dan önce gerçekleştirilmelidir.

Bu durum yandaki gibi gösterilebilir:



- İşlere ait süreler ve teslim tarihleri ise yandaki tabloda verilmektedir.

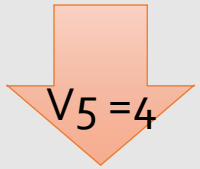
İŞ	j1	j2	j3	j4	j5	j6
İşlem Zamanı (pj)	2	3	4	3	2	1
Teslim Tarihi (dj)	3	6	9	7	11	7

Örnek: Lawler Algoritması

- V = J3, J5, J6 (Bunlar son yapılması gereken işlerdir)
- Bir küme şeklinde toplarız.
- Çizelgelenmesi gereken işlerin işlem zamanları toplamı = 2+3+4+3+2+1 = 15 olur.
- Böylece V şu formda yazılır;
- V = min { (15-9), (15-11), (15-7) }

- Buna göre minimum gecikme J5' te olur.
- Böylece J5 işi 6. sıraya yerleştirilir.

Tablo Adım #1



(– – – – – J5)

Örnek: Lawler Algoritması

Adım	İş Süreleri Toplamı	j1	j2	j3	j4	j5	j6	Programlanan iş
1	15	*	*	6	*	4	8	J5
2	13	*	*	4	*	S	6	J3
3	9	*	3	S	*	S	2	J6
4	8	*	2	S	1	S	S	J4
5	5	*	-1	S	S	S	S	J2
6	2	-1	S	S	S	S	S	J1

Örnek: Lawler Algoritması

- Listeden J5’ i sileriz.
- Böylece geri kalan işlerin tamamlanma zamanı = 15 – 2 = 13 olur.
- Şimdi J3 veya J6 işleri son işler olabilir. Yani, V = {J3, J6} dır.
- Böylece V=min {(13-9), (13-7)}. İçinde minimum gecikme J3’ de olmaktadır. Bu nedenle J3 işi 5. sıraya yerleştirilir. J3 listeden silinir.

(- - - - J3 J5)

Tablo Adım #2

Örnek: Lawler Algoritması

- Geri kalan işlerin süreleri toplamı = 13 – 4 = 9 ve V={J2, J6} dır.
- V=min {(9-6),(9-7)}.
- Minimum gecikme J6’ da olmaktadır. J6, 4. sıraya yerleştirilir. Listeden silinir.

  (- - - J6 J3 J5)

Tablo Adım #3

- Böylece V ={J2, J4} olur.
- T = 9 -1 = 8 ve V=min {(8-6), (8-7)}.
- Minimum gecikme J4’ de olmaktadır. J4, 3. sıraya yerleştirilir. Listeden silinir.

  (- - J4 J6 J3 J5) yeni sıramız olur.

Tablo Adım #4

Örnek: Lawler Algoritması

- Kalan işlerin süreleri toplamı = 8-3 = 5 ve V = {J2} olur.
- V=min {(5-3)}.
- J2, 2. sıraya yerleştirilir. Listeden silinir.

  (- J2 J4 J6 J3 J5) .

Tablo Adım #5

- Son olarak J1 de sıralamaya yerleştirilir ve böylece nihai sıra

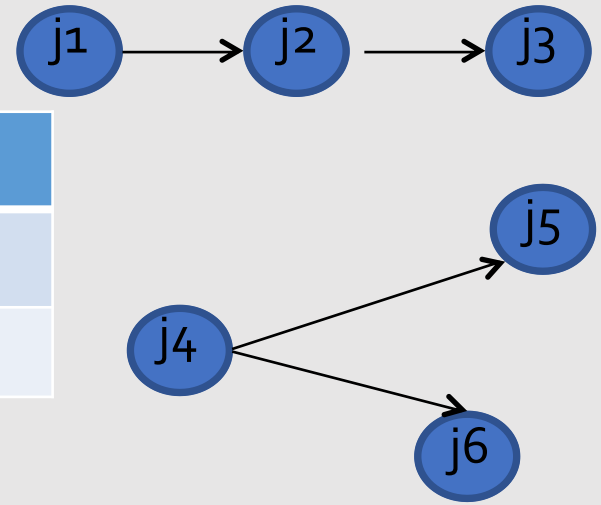
  (J1 J2 J4 J6 J3 J5) olmaktadır.
- Ardından işler Gantt şeması veya bir tablo üzerinde gösterilerek maksimum gecikmeleri hesaplanabilir.

Tablo Adım #6

Lawler Algoritması – 2. Yol

- Lawler algoritması, daha basite indirgenmiş şekilde aşağıdaki prosedür uygulanarak da çalıştırılabilir:
  - *Adım 1: Tüm atanmamış (çizelgelenmemiş) işleri ekleyerek Liste A'yı oluştur.*
  - *Adım 2: Liste A'da bulunan işlerden, öncelik ilişkilerine göre sıralamanın sonuna atanabilir işleri belirle (ardılı olmayan veya tüm ardılları atanmış olan) ve Liste B'ye ekle.*
  - *Adım 3: Liste B'den en geç teslim tarihine (due date) sahip olan işi, son atanan işin bir öncesine ta (son atanan iş yoksa en sona ata).*
  - *Adım 4: Liste A'yı güncelle.*
  - *Adım 5: Liste A ≠ Ø ise Adım 2 ye dön.*

Lawler Algoritması – 2. Yol



İŞ	j1	j2	j3	j4	j5	j6
İşlem Zamanı (pj)	2	3	4	3	2	1
Teslim Tarihi (dj)	3	6	9	7	11	7

Liste A

Atanmamış İşler
1
2
3
4
5
6

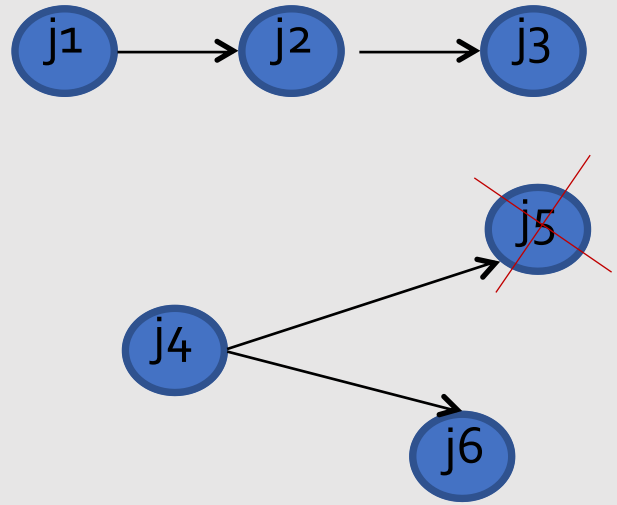
Liste B

İş	Teslim Tarihi (dj)
3	9
5	11*
6	7

Liste B'den Teslim Tarihi en geç olan iş (J5), sıralamanın sonuna eklendi.

Sıralama: (- - - - - 5)

Lawler Algoritması – 2. Yol



- 5 numaralı iş atandığı için Liste A dan çıkartıldı ve yeni durumda listelerin son durumu aşağıdaki gibi oldu.

Liste A

Atanmamış İşler
1
2
3
4
6

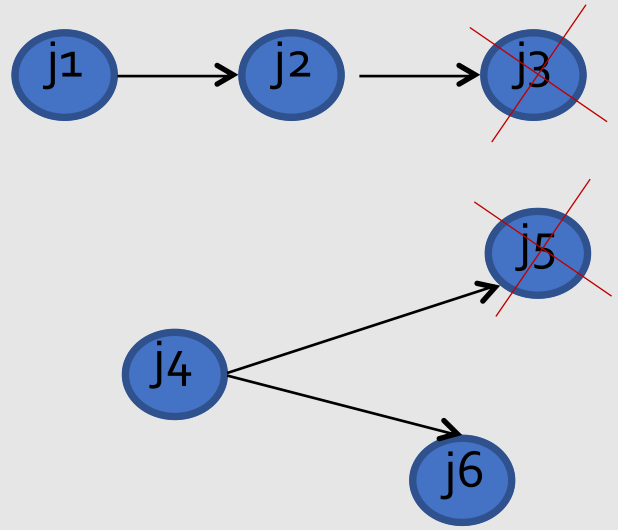
Liste B

İş	Teslim Tarihi (dj)
3	9*
6	7

Liste B'den Teslim Tarihi en geç olan iş (J3), son atanan işten bir önceki sıraya eklendi.

Sıralama: (- - - - 3 5)

Lawler Algoritması – 2. Yol



- 3 numaralı iş atandığı için Liste A dan çıkartıldı ve yeni durumda listelerin son durumu aşağıdaki gibi oldu.

Liste A

Atanmamış İşler
1
2
4
6

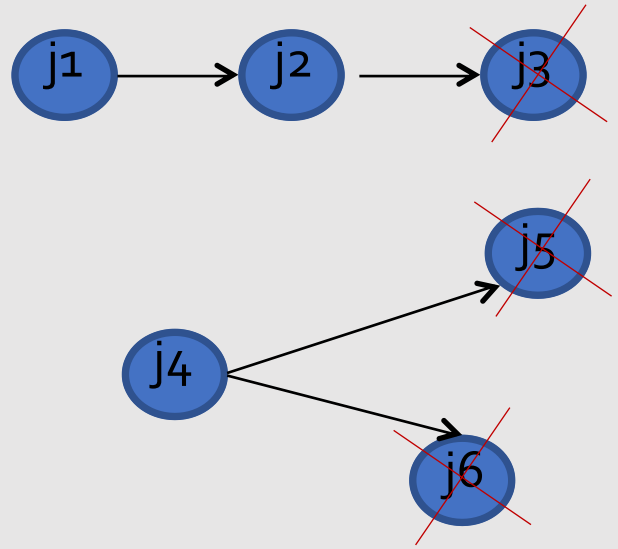
Liste B

İş	Teslim Tarihi (dj)
2	6
6	7

Liste B'den Teslim Tarihi en geç olan iş (J6), son atanan işten bir önceki sıraya eklendi.

Sıralama: (- - - 6 3 5)

Lawler Algoritması – 2. Yol



- 6 numaralı iş atandığı için Liste A dan çıkartıldı ve yeni durumda listelerin son durumu aşağıdaki gibi oldu.

Liste A

Atanmamış İşler
1
2
4

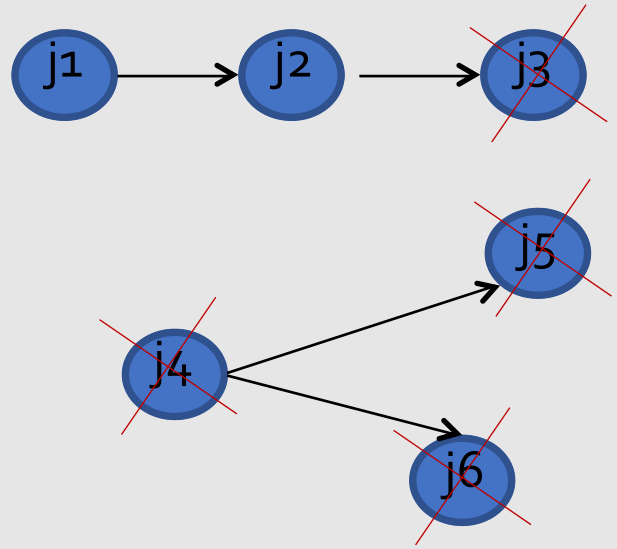
Liste B

İş	Teslim Tarihi (dj)
2	6
4	7

Liste B'den Teslim Tarihi en geç olan iş (J4), son atanan işten bir önceki sıraya eklendi.

Sıralama: (- - 4 6 3 5)

Lawler Algoritması – 2. Yol



- 4 numaralı iş atandığı için Liste A dan çıkartıldı ve yeni durumda listelerin son durumu aşağıdaki gibi oldu.

Liste A

Atanmamış İşler
1
2

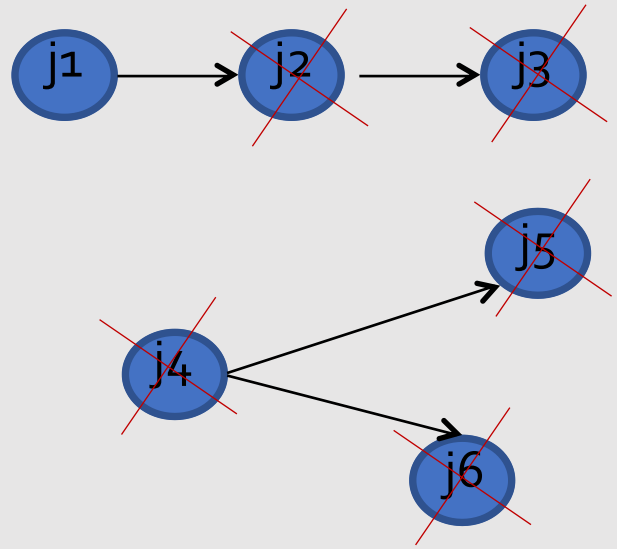
Liste B

İş	Teslim Tarihi (dj)
2	6

Öncelik ilişkisi açısından tek atanabilir iş olan (J2), son atanan işten bir önceki sıraya eklendi.

Sıralama: (- 2 4 6 3 5)

Lawler Algoritması – 2. Yol



- 2 numaralı iş atandığı için Liste A dan çıkartıldı ve yeni durumda listelerin son durumu aşağıdaki gibi oldu.

Liste A

Atanmamış İşler
1

Liste B

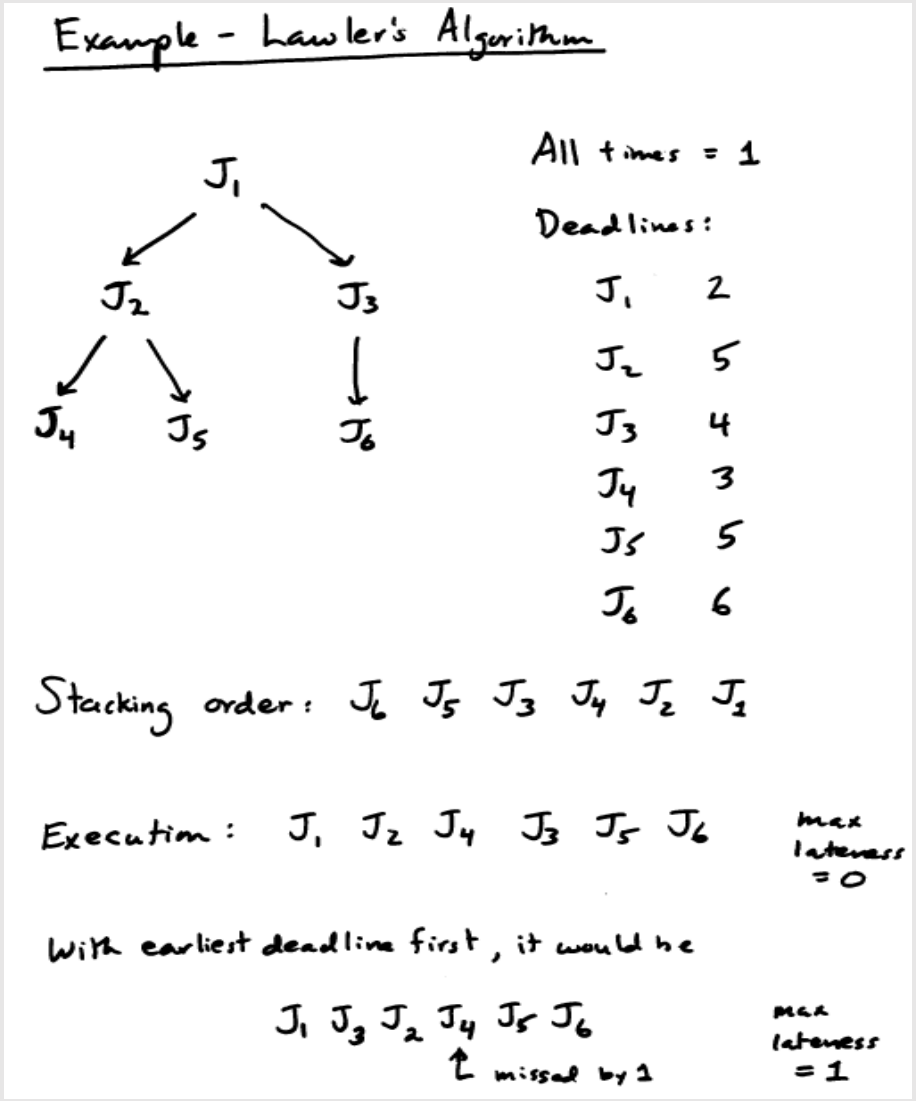
İş	Teslim Tarihi (dj)
1	3

Geriye kalan iş (J1), son atanan işten bir önceki sıraya eklendi ve nihai sıralama aşağıdaki şekilde oluştu:

Sıralama: (1 2 4 6 3 5)

*Bu sıralama, Lmax kriterini minimize etme açısından optimum sonucu verir.

Ödev


Example - Lawler's Algorithm
J1
J2
J3
J4
J5
J6
All times = 1
Deadlines:
J1 2
J2 5
J3 4
J4 3
J5 5
J6 6
Stacking order: J6 J5 J3 J4 J2 J1
Execution: J1 J2 J4 J3 J5 J6
max lateness = 0
With earliest deadline first, it would be
J1 J3 J2 J4 J5 J6
missed by 1
max lateness = 1

Öne geçmeli durum (preempitive)

- Eğer işler atölyeye farklı zamanlarda geliyorsa (r_j değerleri "Sıfır"dan farklıysa), veya bazı önemli işlerin sıralamada öne geçmesi gerekiyorsa, sıralamada bazı esnekliklere gidilebilir.
- Bu, şu anlama gelmektedir: Öncelikli olarak tamamlanması gereken (yani önemli) bir iş sisteme geldiğinde, mevcut iş durdurularak öncelikli iş yapılır. Ardından (öncelikli-önemli iş tamamlandıktan sonra), yarım kalan iş ve diğer işler yapılmaya devam edilir.
- **DİKKAT! Burada bahsedilen öncelik durumu, bir önceki derste anlatılan öncelik kısıtıyla karıştırılmamalıdır!**
- **Örnek:**

Data pertaining to 4-job single machine problem is given in the following table.

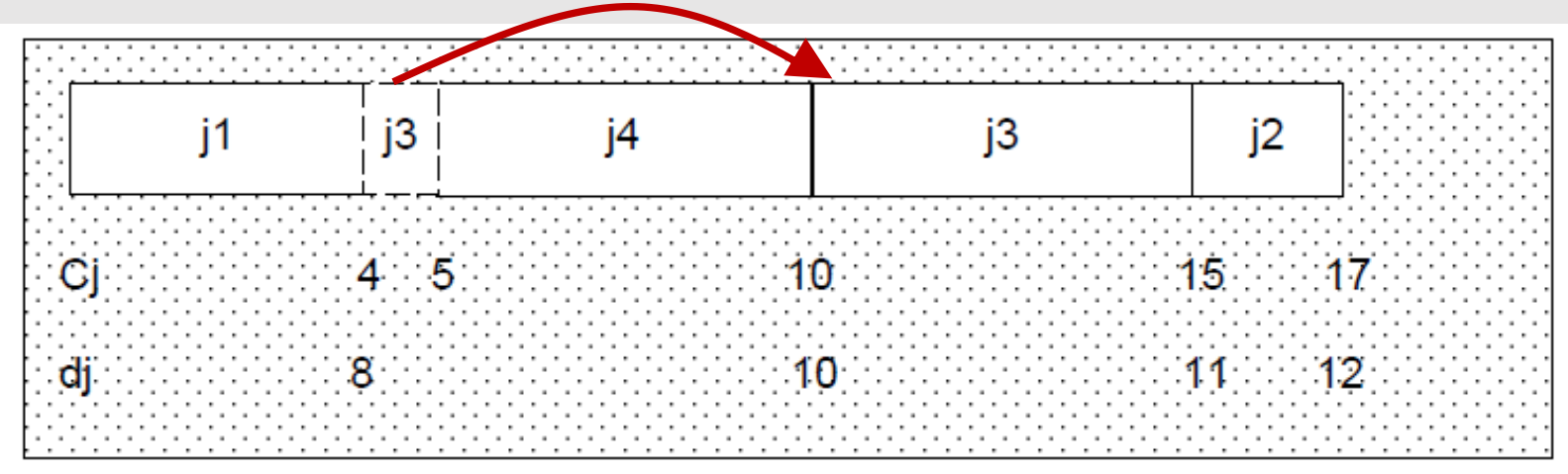
Job(j)	p_j	r_j	d_j
1	4	0	8
2	2	3	12
3	6	3	11
4	5	5	10

Öne geçmeli durum (preempitive)

- İşlerin önem düzeylerini (öncelik durumlarını) EDD kuralının belirlediğini varsayalım. Yani en erken teslim tarihine sahip olan işe öncelik (daha fazla önem) vererek çizelgeyi oluşturalım:
- EDD sıralaması: J1 – J4 – J3 - J2
- **t=0;** Sisteme gelen iş: J1. Dolayısıyla J1, t=0 da başlamak üzere çizelgelenir.
- **t=3;** J2 ve J3 işleri sisteme gelir. Makine meşgul, herhangi bir iş yapamaz.
- **t=4;** J1 işi tamamlanır. Makine boştadır ve herhangi bir işi yapabilir. EDD sıralamasına göre sıradaki iş J4'tür. Ama J4 henüz sisteme gelmediği için makinenin boş kalmaması adına EDD sıralamasındaki bir sonraki iş olan J3 işleme alınır.
- **t=5;** J4 sisteme gelir. Şu anda makine J3'ü işlemektedir. EDD sıralamasına göre J4, J3'e göre daha fazla öneme sahip olduğu için J3 işlemi durdurulur ve J4 makinede işlemeye alınır. J4, t=10'da tamamlanacaktır.
- **t=10;** J4 biter. EDD sıralamasındaki bir sonraki iş olan J3 işleme alınır. J3 işi t=4'ten t=5'e kadar işlendiği için geri kalan işlem süresi t=15'te tamamlanacaktır.

Öne geçmeli durum (preempitive)

- **t=15;** J3 biter. EDD sıralamasındaki bir sonraki iş olan J2 işleme alınır. J2 işi t=17'de tamamlanacaktır.
- **t=17;** J2 biter. Geri kalan iş olmadığı için prosedür sonlandırılır.



Makespan (yayılma zamanı): ?
En büyük gecikme (Lmax): ?

Bölüm Sonu Soruları

1. A single machine facility faces problem of sequencing the production work for six customer orders described in table below.

Order	1	2	3	4	5	6
Processing time	18	26	14	8	17	22

   a. What sequence will minimize the mean flow time of these orders? What is the mean flow time in this schedule?
   b. Suppose that customer orders 1 and 5 are considered twice important as the rest what sequence would you propose.

2. Consider the following single machine scheduling problem

Jobs (j)	1	2	3	4	5
P_j	8	3	3	6	3
d_j	4	8	11	10	11

a. What sequence will minimize the average waiting time? Then compute Average completion time, Maximum flow time, and Average waiting time.

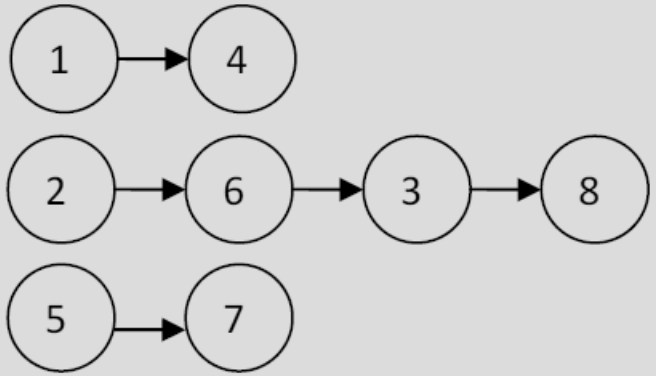
b. What sequence will minimize both maximum lateness and the maximum tardiness? Then, compute maximum lateness, maximum tardiness, maximum earliness, total tardiness, and total earliness.

3. Consider the following single machine scheduling problem

Job	1	2	3	4	5	6
P_j	8	12	7	16	9	4
ω_j	4	10	4	3	8	10

Generate a sequence to minimize weighted completion time. What is the flow time of each job in the shop?

4. Consider the following problem as an instance of the $1 \mid prec \mid \sum w_j C_j$. An 8-job single machine data with job precedence constraints graph is given below.



The weights and process times of the jobs are given in the following table.

Job	1	2	3	4	5	6	7	8
P_j	3	6	6	5	4	8	10	4
w_j	6	18	12	8	8	17	18	15

Solve the problem to minimize total weighted completion times

- 5. Consider the following single machine scheduling problem

Job	1	2	3	4	5	6	7
p_j	8	3	12	1	7	5	3
D_j	12	7	23	4	21	17	8

a) Use SPT sequence and, find average waiting time ($\overline{W}$).

b) Use EDD sequence and, find maximum lateness (L_{max}), average Lateness ($\overline{L}$).

c) Generate a sequence to minimize number of tardy jobs (n_t).

- 6. Consider $1 \parallel n_t$ problem with the following data:

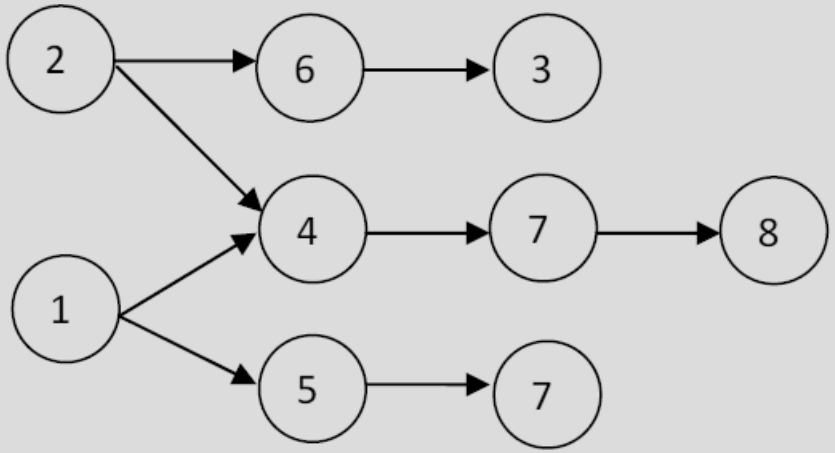
Job	1	2	3	4	5	6	7	8	9	10
P_j	15	11	10	5	25	4	8	3	20	11
d_j	71	76	73	88	47	59	24	55	23	47

Find the sequence that minimizes the number of jobs tardy and compute n_t

7. Consider the following data as an instance of the 1| prec | L_{max} problem.

Job	1	2	3	4	5	6	7	8
p_j	2	3	2	2	4	3	2	2
d_j	5	4	13	6	12	10	15	19

The precedence network for the problem is shown below:



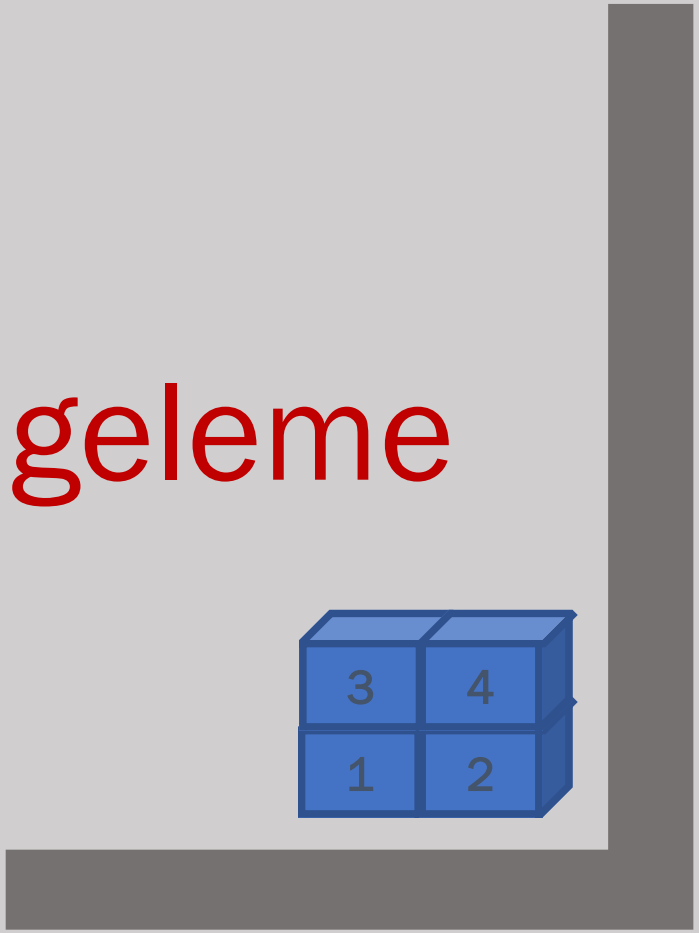
Find the best efficient sequence this problem and compute L_{max} ?

8. Consider an instance of the 1 | rj | L_{max} problem with data as follows.

Job	1	2	3	4	5
p_j	3	4	6	10	2
r_j	0	1	7	6	9
d_j	3	10	17	18	15

Find optimal sequence and show complete schedule.

Paralel Makine Çizelgeleme



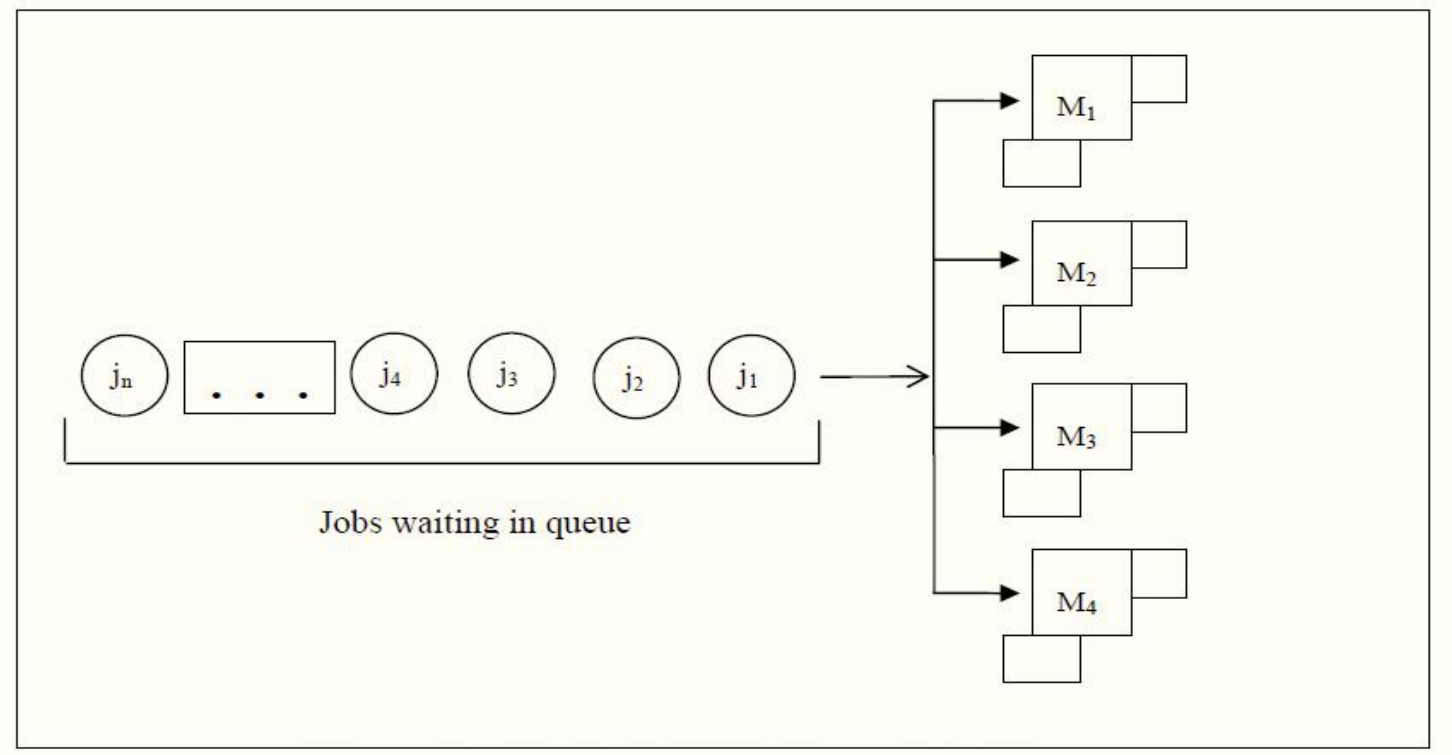
Paralel Makine Çizelgeleme

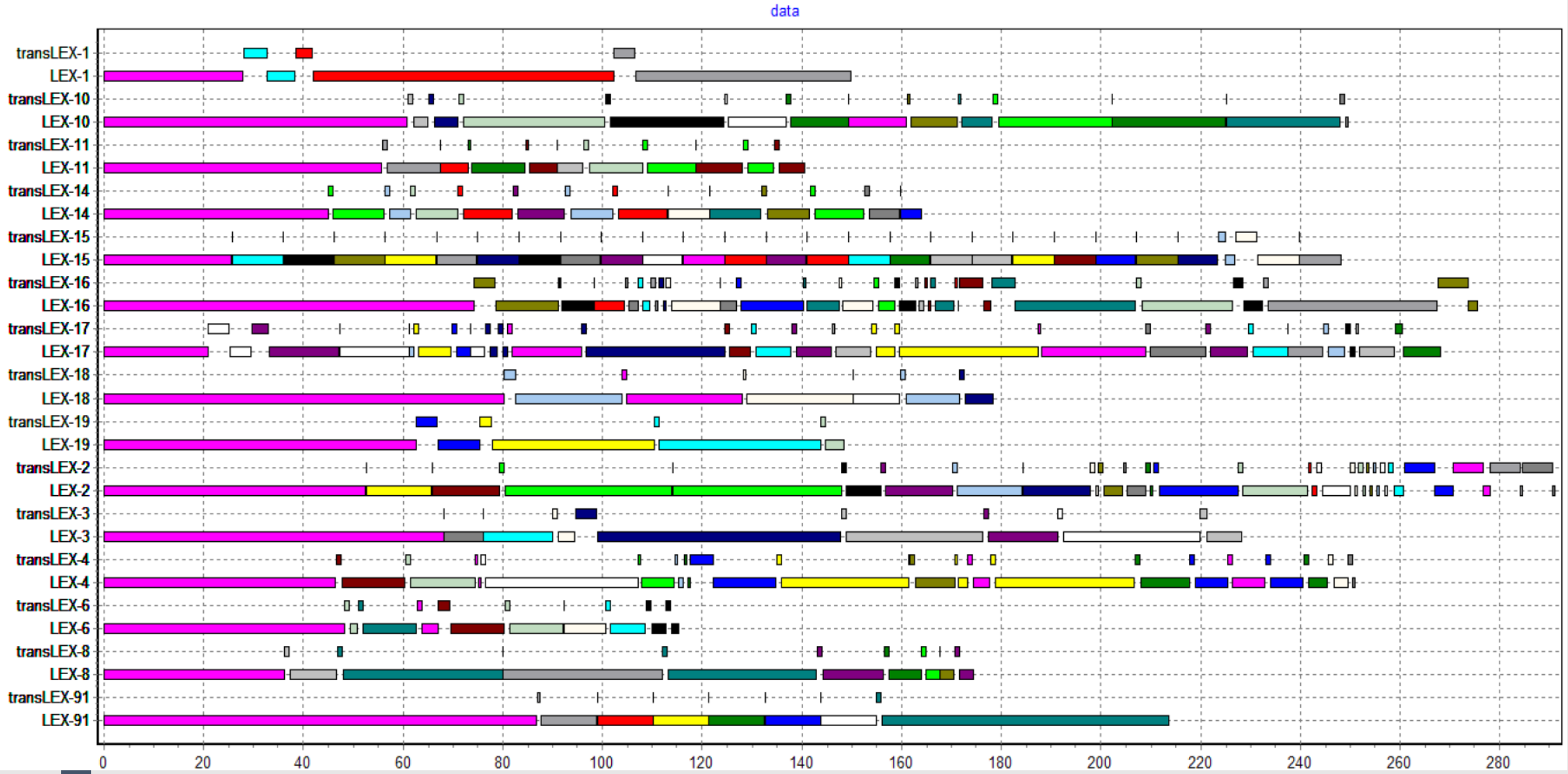
- Teorik bakış açısında paralel makine ortamı, tek makine ortamının genelleştirilmiş durumudur.
- Paralel makinelerde sıralama iki aşamalı bir işlem olarak kabul edilmelidir.
  - *Birinci aşamada, hangi işin hangi makinede işleneceği saptanmalıdır.*
  - *ikinci aşamada ise her makinedeki iş yükü için sıralama tespit edilmelidir.*
- İşlem zamanı performans kriterleri açısından sadece birinci aşama önemlidir.
- Paralel makine problemleri, m tane eş makinenin paralel olarak yerleştiği sistemler olmaktadır.
- Her iş, yalnız bir operasyona sahiptir ve bu, m makinenin herhangi birinde işlenebilir.
- Genellikle makine sayısı, iş sayısından az olmaktadır.
- Örnek bir paralel makine sistemi aşağıdaki Şekil ’de gösterilmiştir.

Literatür

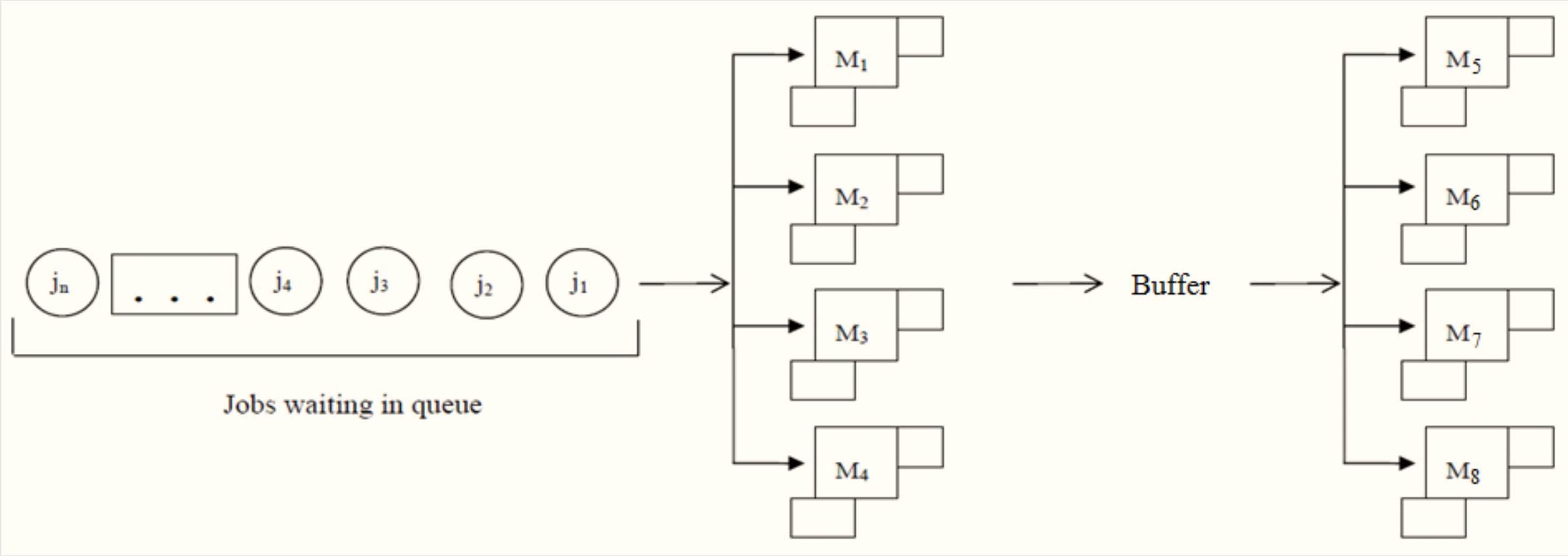
- Rajakumar ve arkadaşları (2004) yaptıkları çalışmada, paralel makine çizelgelemede iş sırasının belirlenmesi için üç farklı öncelik stratejisini kıyaslamışlardır. Rastgele, En kısa işlem süresine öncelik tanıma ve En uzun işlem süresine öncelik tanıma yöntemlerini kullanarak, n=50 iş ve m=2,3,4,5,6 makine sayıları için ayrı ayrı çizelgeleme yapmışlardır.
- C++ ile ele aldıkları problemlerde En uzun işlem süresine öncelik tanıma yöntemine göre yapılan çizelgelemenin diğer yöntemlere göre daha iyi sonuç verdiğini göstermişlerdir.
- Shim ve Kim (2007) Dal Sınır algoritması'nı kullanarak özdeş paralel makineleri çizelgelemiştir. Toplam gecikmeyi minimize etmeyi amaçladıkları çalışmada rastgele yaratılan test problemlerini kullanmışlardır. Yapılan hesaplamalar sonucunda, önerilen algoritmanın 30 iş ve 5 makineye kadar olan problemlerde optimuma yakın sonuçlar yaratacağı saptanmıştır.
- Min ve Cheng (1999) yılında yaptıkları çalışmada paralel makinelerde toplam tamamlanma zamanını genetik algoritma ile ele almışlardır.
- Tabu arama, benzetilmiş tavlama ve komşuluk arama yöntemlerinin pek çok özelliklerini bir araya getirerek yeni bir melez metasezgisel yöntem geliştiren Anghinolfi ve Paolucci (2007) paralel makinelerde toplam gecikmeyi minimize etmeyi amaçlamışlardır .

Paralel Makine Sistem Modeli




data
transLEX-1
LEX-1
transLEX-10
LEX-10
transLEX-11
LEX-11
transLEX-14
LEX-14
transLEX-15
LEX-15
transLEX-16
LEX-16
transLEX-17
LEX-17
transLEX-18
LEX-18
transLEX-19
LEX-19
transLEX-2
LEX-2
transLEX-3
LEX-3
transLEX-4
LEX-4
transLEX-6
LEX-6
transLEX-8
LEX-8
transLEX-91
LEX-91
0
20
40
60
80
100
120
140
160
180
200
220
240
260
280

Hibrit Paralel Makine Sistem Modeli



Paralel Makine Çeşitleri

- Literatürde yapılan araştırmalara göre paralel makine çeşitlerini dört grup altında toplayabiliriz.
- Bunlar şöyle sıralanmaktadır:
  - ***Aynı tip** olup, hızları da aynı olan paralel makineler.*
  - *Aynı tip olup, hızları **farklı** olan paralel makineler.*
  - ***Farklı tip** olup, aynı hızda olan paralel makineler.*
  - *Farklı tipte olup, **farklı** hızda olan paralel makineler.*

Paralel Makinelerden Oluşan Atölyede Sıralama Problemi

- Atölyede birden çok tezgahın paralel olarak çalışması, sıralama problemi için değişik modeller kurmayı gerektirmektedir.
- Bu tür bir atölyede bulunan tüm tezgahlar, mevcut *n* adet işi yapabilecek kapasite ve yetenektedirler. Bu kabulden başka, tezgah ve işler için şu kısıtlar konulmaktadır:
  1. *m adet tezgah sürekli olarak çalışabilir ve bir tezgah aynı anda birden çok işi işleyemez.*
  2. *Başlangıç anında mevcut, birbirinden bağımsız n adet iş, yalnız bir tezgah tarafından işlenerek atölyeyi terk eder.*
  3. *İş tarifleri önceden bilinmektedir, işlem süreleri belirli ve sabittir.*

Paralel Makinelerden Oluşan Atölyede Sıralama Problemi

- Paralel tezgahlar durumu diğer problemlerden farklı olarak 2 boyutlu bir kararı geliştirmek için incelenmektedir.
- Kararın birinci boyutu; işlerin tezgahlara dağıtılması, ikinci boyutu ise; bu tezgahlardaki işlenme sırası ile ilgilidir.
- Karar prosesi bu iki boyutu kapsayacak şekilde ele alınarak bazı etkinlik ölçütlerine göre problem incelenmektedir.

Paralel Makine Çizelgeleme – Örnek 1

LPT (En Uzun İşlem Süresi) Kuralı

- Paralel makinelerde yaygın olarak kullanılan en basit kurallardan birisi LPT (en uzun işlem süresi) kuralıdır.
- Bu kuralda işler işlem sürelerine göre büyükten küçüğe doğru sıralanır.
- Büyük işlem süresine ait işler paralel makinelere atanırken diğerlerine göre önceliğe sahip olur.
- Aşağıdaki, **aynı hızdaki dört makineden** oluşan paralel makine çizelgeleme örneğini LPT kuralını dikkate alarak çözelim:

Job (j)	1	2	3	4	5	6	7	8	9
p_j	7	7	6	6	5	5	4	4	4

Amaç: Yayılma Zamanını (Makespan) minimize etmek.

Paralel Makine Çizelgeleme – Örnek 1

- LPT kuralına göre sıralama: 1-2-3-4-5-6-7-8-9 şeklindedir.
- Sıralamaya göre en fazla önceliğe sahip iş, J1, alınarak birinci makineye (M1) atanır. Ardından J2, ikinci makineye (M2), J3 işi üçüncü makineye ve J4 işi dördüncü makineye atanır. Bu işlemler sonrası oluşan **kısmi sıralama** aşağıdaki gibi olur:

Machine (M)	Job assigned	Start Time S_j	Process Time p_{ij}	Completion Time (C_j)
M_1	1	0	7	7
M_2	2	0	7	7
M_3	3	0	6	6
M_4	4	0	6	6

Paralel Makine Çizelgeleme – Örnek 1

- Çizelgelenmemiş (atanmamış) işler: 5-6-7-8-9
- t=6'da M3 ve M4 boş oldukları için J5 ve J6 bu makinelere atanabilir:

Job (j)	Machine (M)	Start Time S_j	Process Time p_{ij}	Completion Time C_j
j_1	M_1	0	7	7
j_2	M_2	0	7	7
j_3	M_3	0	6	6
j_4	M_4	0	6	6
j_5	M_3	6	5	11
j_6	M_4	6	5	11

Paralel Makine Çizelgeleme – Örnek 1

- Çizelgelenmemiş (atanmamış) işler: 7-8-9
- t=7'de M1 ve M2 boş oldukları için J7 ve J8 bu makinelere atanabilir:

Job (j)	Machine (M)	Start Time S_j	Process Time p_{ij}	Completion Time C_j
j_1	M_1	0	7	7
j_2	M_2	0	7	7
j_3	M_3	0	6	6
j_4	M_4	0	6	6
j_5	M_3	6	5	11
j_6	M_4	6	5	11
j_7	M_1	7	4	11
j_8	M_2	7	4	11

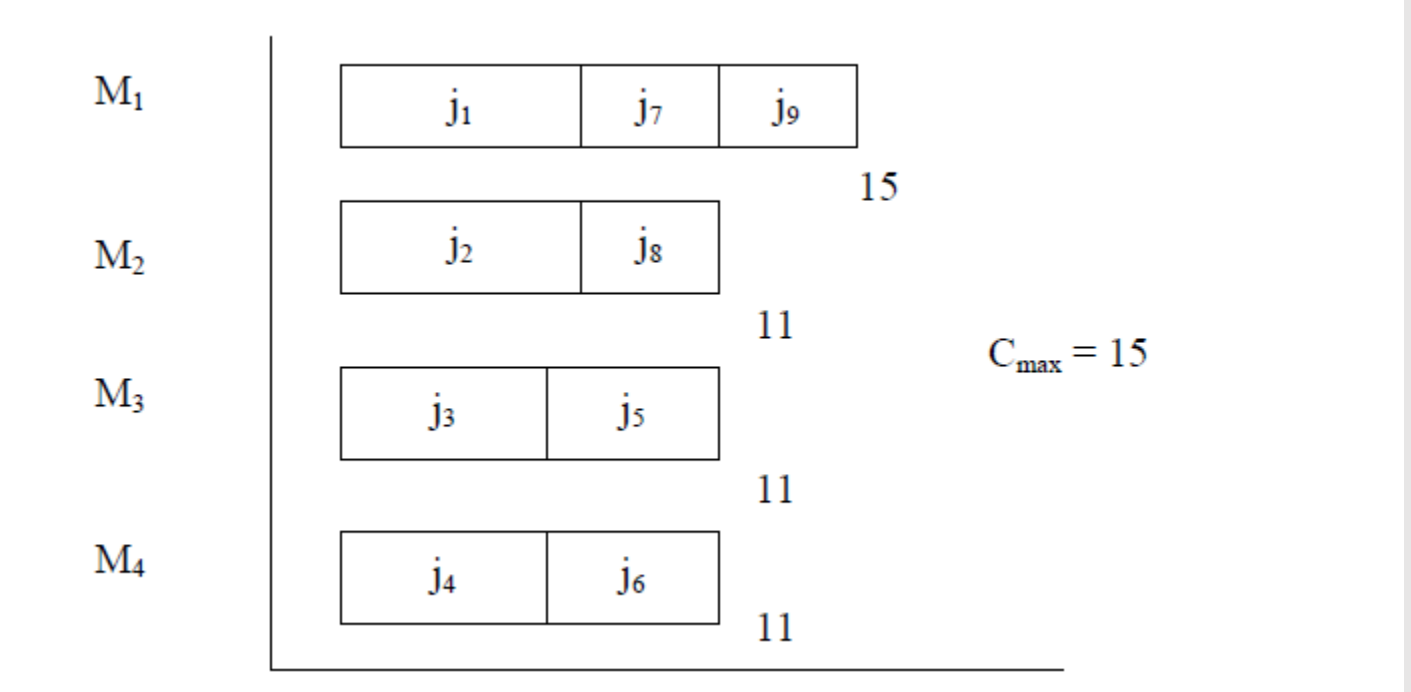
Paralel Makine Çizelgeleme – Örnek 1

- Çizelgelenmemiş (atanmamış) işler: 9
- t=11'de M1, M2, M3 ve M4 makinelerinin hepsi boş oldukları için J9 herhangi birisine atanabilir.

Job (j)	Machine (M)	Start Time S_j	Process Time p_{ij}	Completion Time C_j
j_1	M_1	0	7	7
j_2	M_2	0	7	7
j_3	M_3	0	6	6
j_4	M_4	0	6	6
j_5	M_3	6	5	11
j_6	M_4	6	5	11
j_7	M_1	7	4	11
j_8	M_2	7	4	11
j_9	M_1	11	4	15

Paralel Makine Çizelgeleme – Örnek 1

- Elde edilen çözümün Gantt diyagramı aşağıdaki gibi olur:



Çözüm optimum mu?

Paralel Makine Çizelgeleme – Örnek 1

- Bu problemde optimum çözüm tüm makinelerin boşluksuz şekilde yerleştirilmesiyle mümkün olacağı için, *teorik minimum yayılma zamanı* veya *teorik minimum en büyük tamamlanma zamanı* (Cmax) aşağıdaki formülle hesaplanabilir:

$$Cmax(OPT) = max\left\{ p_j, \frac{\sum_{j=1}^{J} p_j}{m} \right\} = 12$$

İşler bölünemez

- Yani, tüm makineler tam olarak yüklenebilseydi t=12 de tüm işler tamamlanmış olacaktı.
- Dikkat edilmesi gereken konu, bu formülasyonun özdeş paralel makinelerde geçerli oduğu hususudur.
- Elde ettiğimiz çözümün kalitesi ise aşağıdaki formülasyonu kullanarak (teorik optimum çözüme oranlamayla) bulunabilir:

$$\frac{Cmax(OPT)}{Cmax(LPT)} = \frac{12}{15} = 0.8$$

Bu hesaplama sonrası makinelerin ve doğal olarak bunlara ayrılan zamanın %80 oranında etkin kullanıldığını söyleyebiliriz.

Paralel Makine Çizelgeleme – Örnek 1

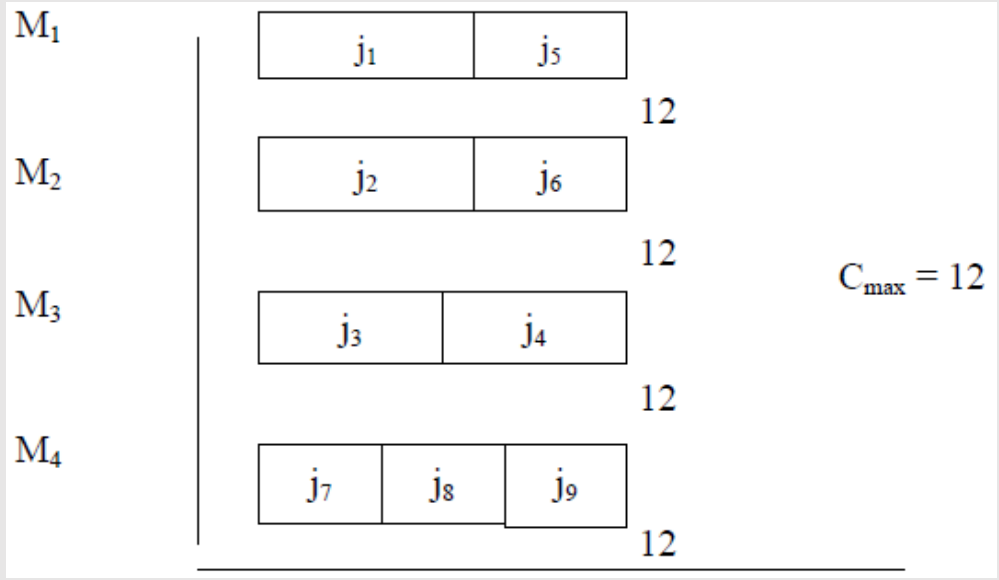
Yük Dengeleme (Load Balancing) Sezgiseli

- Aynı örnek için **Yük Dengeleme (Load Balancing)** sezgiseli kullanılarak optimum çözüm şu şekilde elde edilebilir:

- Teorik optimum Cmax hesaplanır: $Cmax(OPT) = \frac{\sum_{j=1}^{J} p_j}{m} = \frac{48}{4} = 12$

- Dolayısıyla, her makine için yüklenmesi gereken iş süresi miktarı teorik olarak 12 olmalıdır.

Machine	Jobs	Total Time
M_1	j_1, j_5	$7 + 5 = 12$
M_2	j_2, j_6	$7 + 5 = 12$
M_3	j_3, j_4	$6 + 6 = 12$
M_4	j_7, j_8, j_9	$4 + 4 + 4 = 12$



Optimum çözüme ait Gantt diyagramı

Öncelik Kısıtlı Paralel Makine Çizelgeleme

- Paralel makine ortamında, işler arasında öncelik ilişkileri bulunabilir. Bu durumda öncelik ilişkileri diyagramı dikkate alınarak işlerin en erken ve en geç tamamlanma zamanları hesaplanır:

$$C_j' : j\ işinin\ en\ erken\ tamamlanma\ zamanı$$

$$C_j' = \max_{i \in \psi}\{C_i'\} + p_j$$

- Burada ψ, j işinin öncüllerinin kümesidir. C_i' ise ψ kümesindeki i işinin en erken tamamlanma zamanıdır.

$$C_j'' : j\ işinin\ en\ geç\ tamamlanma\ zamanı$$

$$C_j'' = \min_{k \in \Omega}\{C_k'' - p_k\}$$

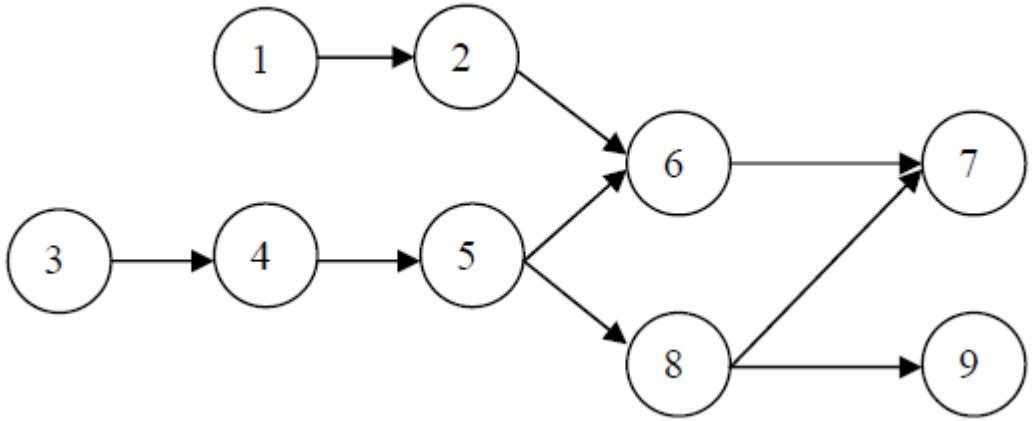
- Burada C_k'' , Ω kümesindeki k işinin en geç tamamlanma zamanıdır.
- Tüm ardılı olmayan işlerin en geç tamamlanma zamanı Cmax'a eşitlenir ve hesaplamalar geriye doğru yapılır.

Öncelik Kısıtlı Paralel Makine Çizelgeleme

- Dokuz işten oluşan bir paralel makine ortamını düşünelim:

Job (j)	1	2	3	4	5	6	7	8	9
p_j	4	9	3	3	6	8	8	12	6

- İşlere ait öncelik diyagramı aşağıdaki gibi olsun:

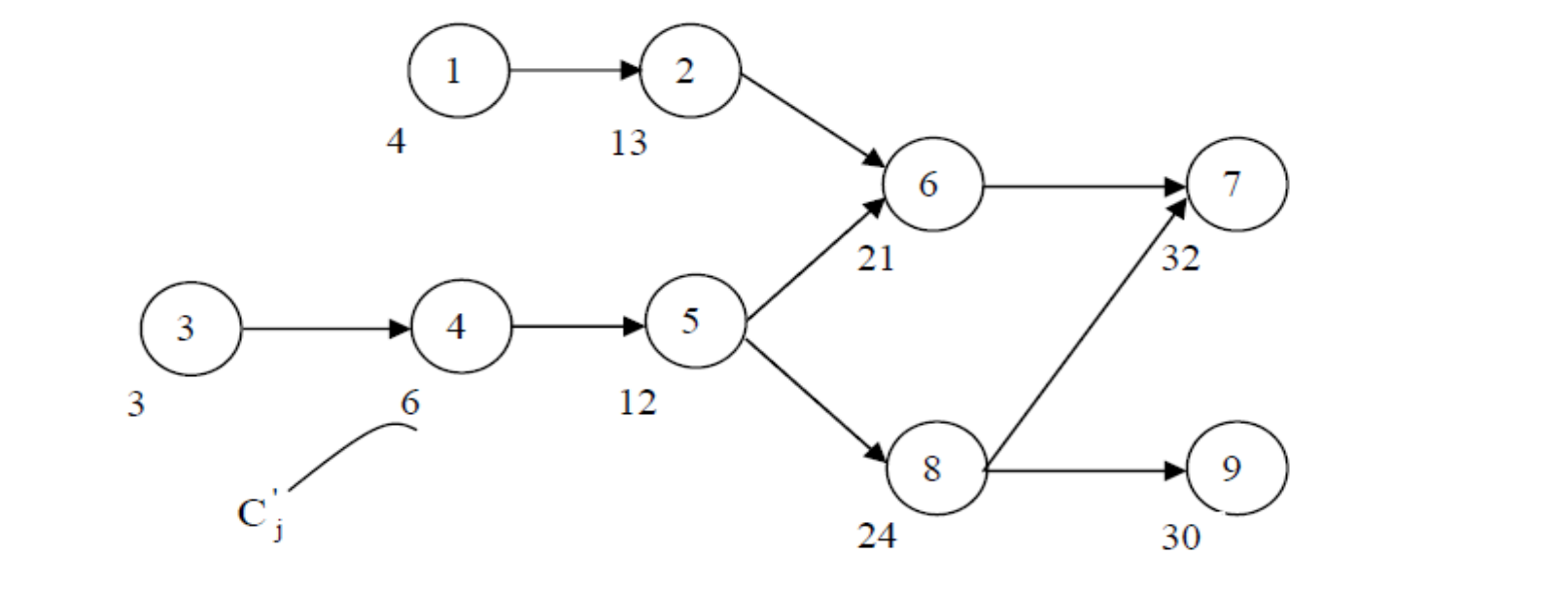


Yayılma Zamanını (M) ve Kritik Yolu (Critical Path – CP) bulunuz.

Öncelik Kısıtlı Paralel Makine Çizelgeleme

- İşlerin en erken tamamlanma zamanları aşağıdaki gibidir.

Job (j)	1	2	3	4	5	6	7	8	9
C'_j	4	13	3	6	12	21	32	24	30



Öncelik Kısıtlı Paralel Makine Çizelgeleme

- J7 en büyük tamamlanma zamanına (32) sahiptir. Bu yüzden Cmax=32'dir.
- Kritik yolu bulmak için işlerin en geç tamamlanma zamanları aşağıdaki gibi hesaplanır.
- Ardılı olmayan tüm işlerin (J7 ve J9) en geç tamamlanma zamanları Cmax'a eşitlenir.
- J8'in en geç tamamlanma zamanı aşağıdaki gibi hesaplanır:

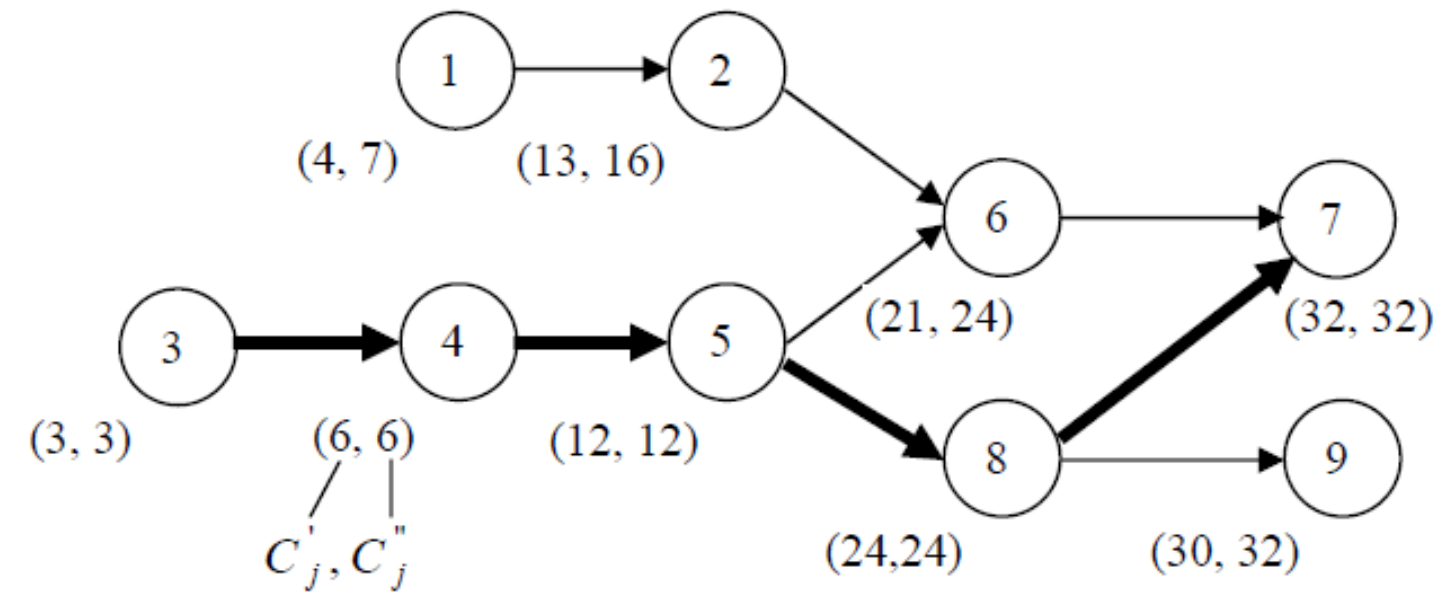
$$C_8'' = \min\{C_7'' - p_7, C_9'' - p_9\} = \min\{32-8, 32-6\} = \min\{24, 26\} = 24$$

- Tüm işlerin en geç tamamlanma zamanları aşağıdaki gibidir:

Job (j)	1	2	3	4	5	6	7	8	9
C_j''	5	16	3	6	12	24	32	24	32

Öncelik Kısıtlı Paralel Makine Çizelgeleme

- Aşağıdaki grafik en erken ve en geç tamamlanma zamanlarını göstermektedir. Burada erken ve en geç tamamlanma zamanları birbirine eşit olan işler kritik yolu oluşturmaktadır.



- Kritik yolun bulunmasındaki amaç, hangi işlerin gecikme olmadan (veya olabildiğince minimum gecikme ile) tamamlanması gerektiğini tespit etmektir.

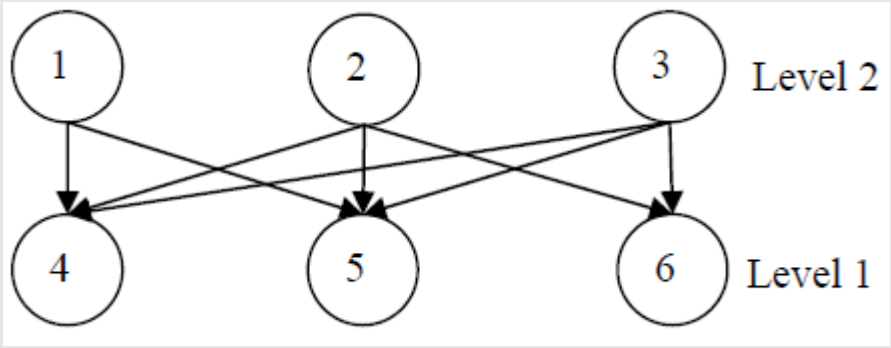
Öncelik Kısıtlı Paralel Makine Çizelgeleme (pj=1)

Kritik Yol (Critical Path – CP)

- CP kuralı, öncelik ilişkilerinin dikkate alındığı paralel makine çizelgeleme problemlerinde Cmax'ı minimize etmek için kullanılabilir.
- Bu kuralın işletilmesi için kritik yolun bulunmasına gerek yoktur. İşler, öncül ardıl ilişkileri bakımından seviyelere ayrılır, en fazla ardılı olan işe daha fazla önem verilmektedir ve bir bakıma *En Yüksek Seviye İlk İşlem Görür* kuralıdır.

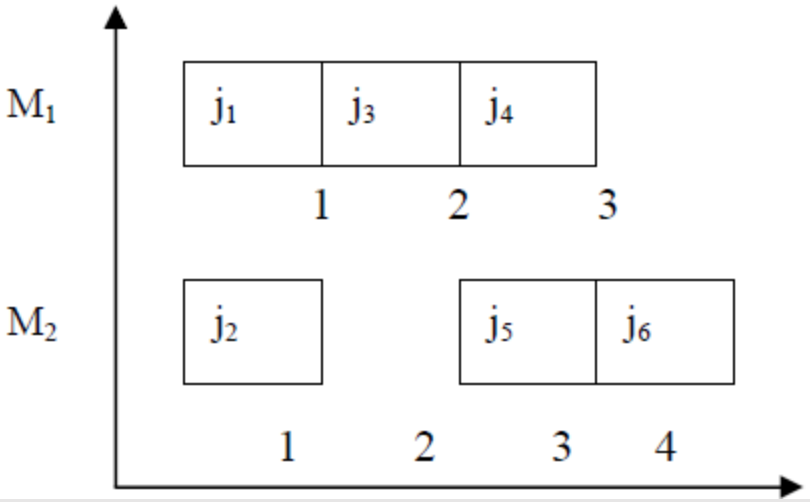
Örnek:

- Aşağıdaki Pm|pj=1,tree|Cmax problemi için CP kuralını uygulayıp Cmax'ı bulalım. Atölyede paralel konumda iki makine bulunmaktadır ve tüm işlerin süreleri 1 zb olarak dikkate alınmıştır.



Öncelik Kısıtlı Paralel Makine Çizelgeleme (pj=1)

- CP kuralını işletirken ilk olarak en yüksek seviyedeki (level 2) işler seçilir.
- t=0'da J1, J2 ve J3 bu işlerdir ve öncelikleri yoktur. İlk olarak J1 işi M1'e, J2 işi M2'ye atanır.
- t=1'de sadece J atanabilir. Bu yüzden J3 işi M1'e atanır.
- t=2'de, level 1'de bulunan işler (J4, J5, J6) atanabilir durumdadır. Fakat sadece ikisi atanabilir.J4 ve J5 işleri M1 ve M2 ye atanır.
- Son olarak ise t=3'te J6 işi M2'ye atanır.



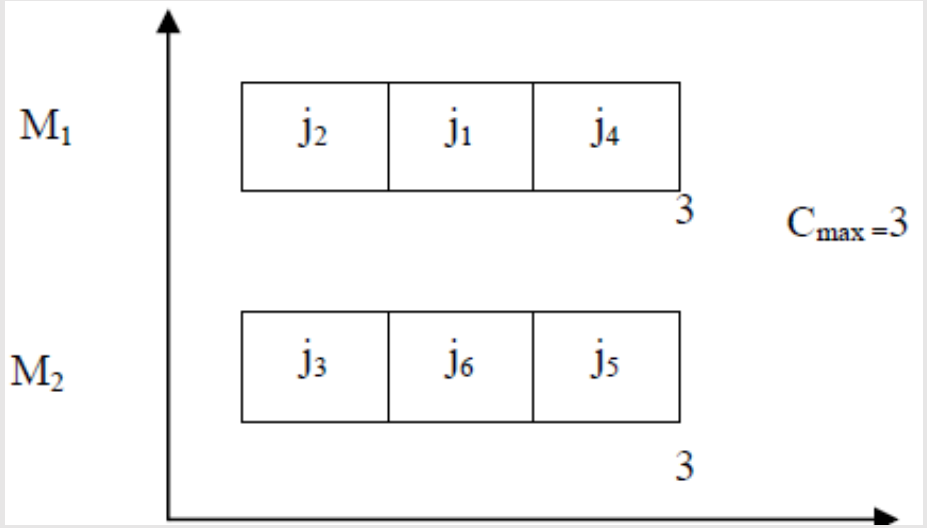
Cmax=4

Öncelik Kısıtlı Paralel Makine Çizelgeleme (pj=1)

En Fazla Ardıl Sayısı (Largest Number of Successors – LNS)

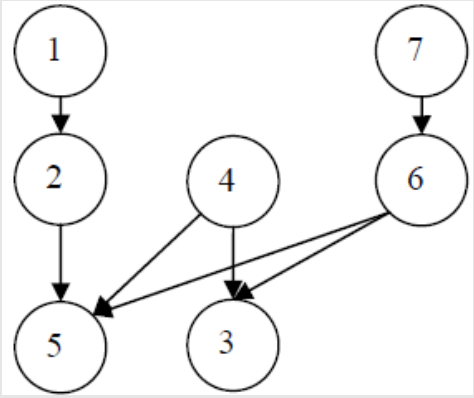
- Bu kuralda paralel makinelerde çizelgeleme yaparken en fazla ardıla sahip olan işe öncelik verilir.
- Bir önceki problem LNS kuralı kullanılarak aşağıdaki şekilde çözülebilir:

Job (j)	1	2	3	4	5	6
p_j	1	1	1	1	1	1
NS	2	3	3	0	0	0



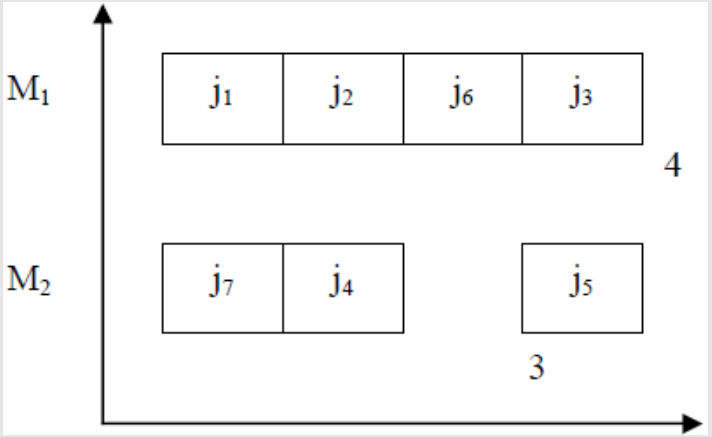
Öncelik Kısıtlı Paralel Makine Çizelgeleme (pj=1)

- Örnek: Yandaki P2|pj=1,tree|Cmax problemini CP ve LNS kurallarını kullanarak çözelim:



- Çözüm 1: CP kuralı kullanılarak yapılan çözüm:

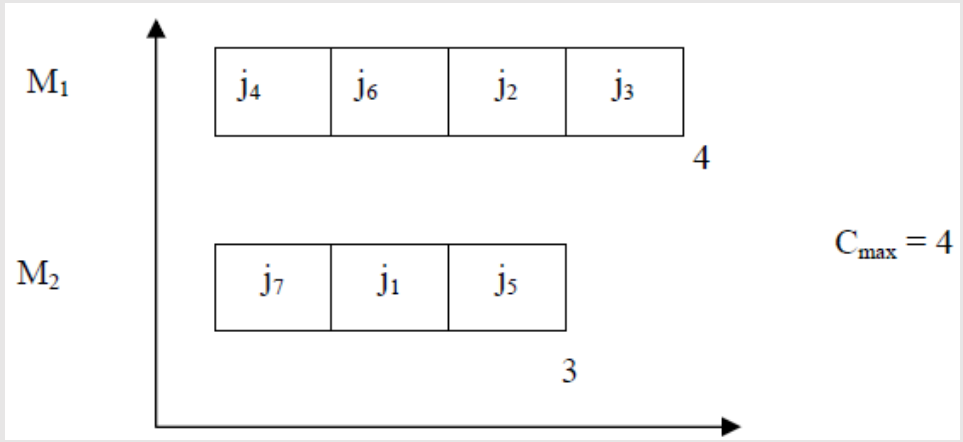
Jobs	Level No
j5, j3	1
j2, j4, j6	2
j1, j7	3



Öncelik Kısıtlı Paralel Makine Çizelgeleme (pj=1)

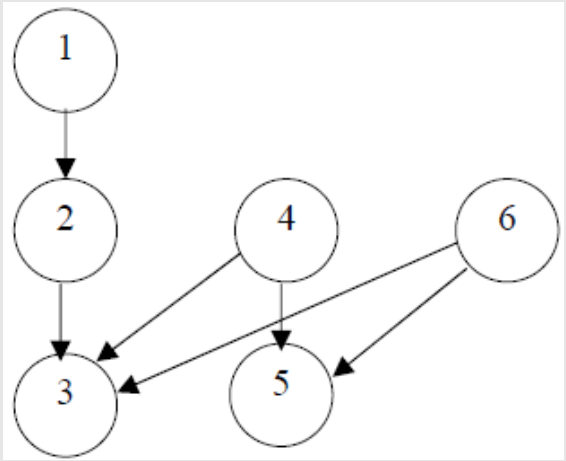
- Çözüm 2: LNS kuralı kullanılarak yapılan çözüm:

Job (j)	1	2	3	4	5	6	7
p_j	1	1	1	1	1	1	1
NS(j)	1	1	0	2	0	2	1



Öncelik Kısıtlı Paralel Makine Çizelgeleme (pj=1)

- Örnek: Yandaki P2|pj=1,tree|Cmax problemini LNS ve CP kurallarını kullanarak çözelim:

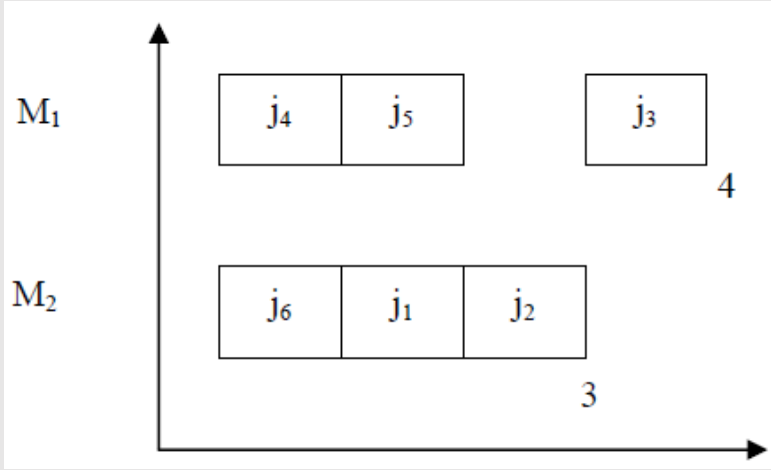


- Çözüm 1: LNS kuralı kullanılarak yapılan çözüm:

Job (j)	1	2	3	4	5	6
NS(j)	1	1	0	2	0	2

Öncelik Kısıtlı Paralel Makine Çizelgeleme (pj=1)

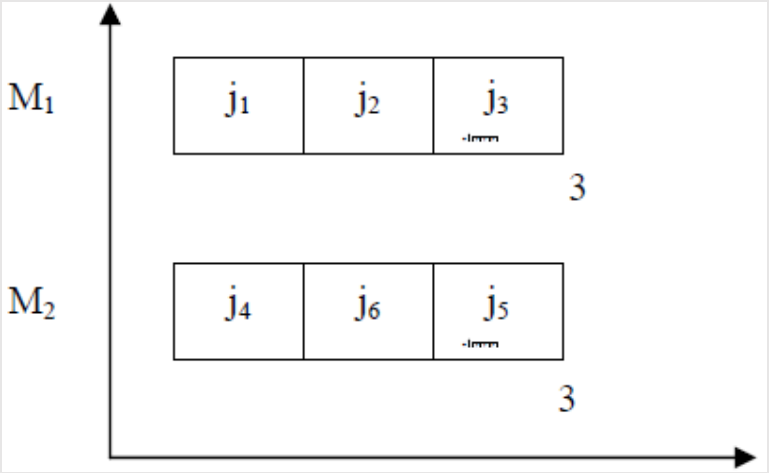
Time t	Set of Scheduled Jobs	Set of Unscheduled Jobs	Schedule on M_1	Schedule on M_2
0	{ }	All jobs	j_4 (NS = 2)	j_6 (NS = 2)
1	$\{j_4, j_2\}$	$\{j_1, j_3, j_5, j_6\}$ Schedulable jobs are j_5 and j_1	j_5 (NS = 0) Although j_2 has NS value of 1, but its predecessor is not yet scheduled	j_1 (NS = 1)
2	$\{j_4, j_2, j_1, j_5\}$	$\{j_2, j_3\}$ Schedulable job is j_2 only	None Job j_3's predecessor condition is not OK	j_2 (NS = 1}
3	All except job j_3	$\{j_3\}$ Job j_3 is schedulable	j_3 (NS = 0}	None



Öncelik Kısıtlı Paralel Makine Çizelgeleme (pj=1)

- Çözüm 2: CP kuralı kullanılarak yapılan çözüm:

Jobs	Levels
j_1	3
j_2, j_4, j_6	2
j_3 , j_5	1



Time t	Set of Scheduled Jobs	Set of Unscheduled Jobs	Schedule on M_1	Schedule on M_2
t=0	{ }	All jobs	Job j_1 has highest level of 3. So schedule job j_1	j_4 and j_6 at Level 2 can be scheduled. Pick job j_4
t=1	$\{j_4,j_1\}$	$\{j_2,j_3,j_6,j_5\}$ Job's Level Level 2 -> $j_2,j_6\}$ Level 1 -> $j_3,j_5\}$	Schedule j_2	Schedule j_6
t=2	All except job j_3,j_5	$\{j_3,j_5\}$	Schedule j_3	Schedule j_5
t=3	All jobs	{ }	***	***

Paralel Makinelerde Öne Geçmeli Hal (preempitive)

- Bir iş, bir tezgahta bitmeden başka bir tezgaha aktarılabiliyorsa, problem oldukça basit bir şekilde çözülebilir. Bu halde yayılım süresinin (makespan veya Cmax) minimum değeri;

$$Cmax^* = max\left\{ p_j, \frac{\sum_{j=1}^{J} p_j}{m} \right\}$$

şeklinde ifade edilebilir. $Cmax^*$ değeri hesaplandıktan sonra işler şu şekilde sıralanır:

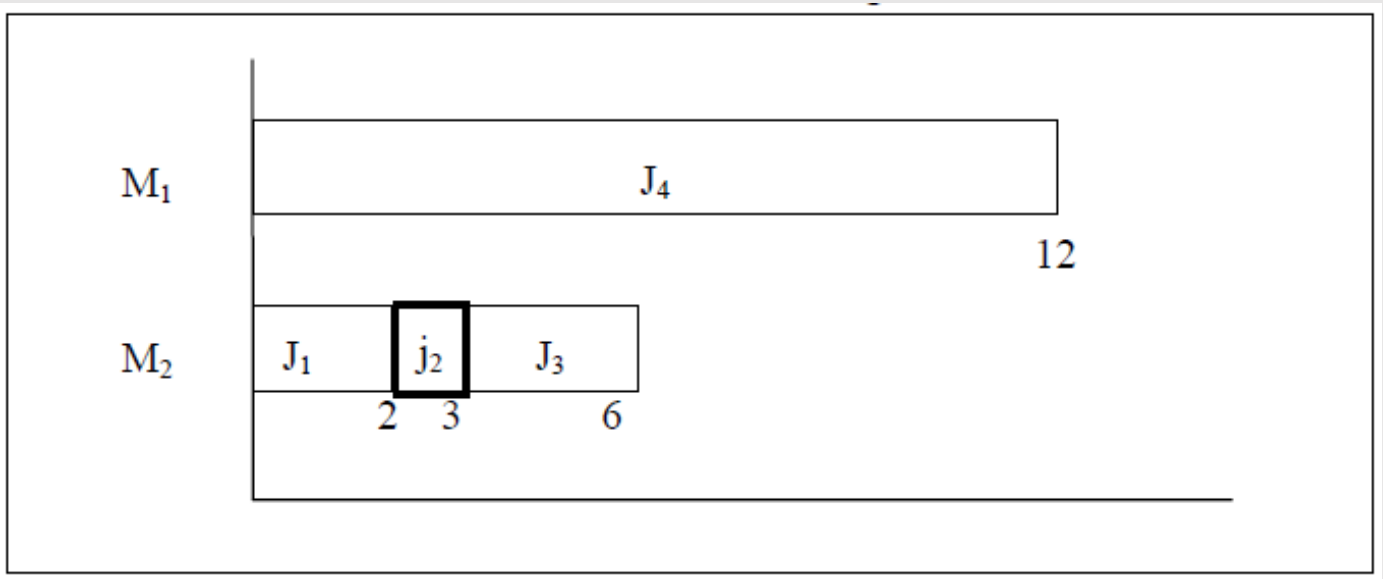
- Adım 1: Herhangi bir iş, "Sıfır" anında başlamak üzere 1. tezgahta programlanır.
- Adım 2: Programlanmamış işlerden biri bu işten sonra gelecek şekilde aynı tezgahta programlanır. Bu tezgahtaki işlem süreleri $Cmax^*$ oluncaya kadar aynı işlem tekrarlanır.
- Adım 3: Tezgahta en son programlanan iş $Cmax^*$ süresi içinde bitirilemiyorsa, bu işe ait geri kalan işlem süresi bir sonraki tezgahta programlanır ve adım 2 ye dönülür. Tezgahta en son programlanan iş süresi $Cmax^*$ sonunda bitiriliyorsa, bir sonraki tezgaha programlanmamış işlerden biri tahsis edilerek adım 2 ye dönülür.
- Adım 2 ve adım 3 tüm işler programlanıncaya kadar sürdürülür.

Paralel Makinelerde Öne Geçmeli Hal (preempitive)

- **Örnek:**

Jobs	1	2	3	4
Processing Time	2	1	3	12

- **Çözüm:**



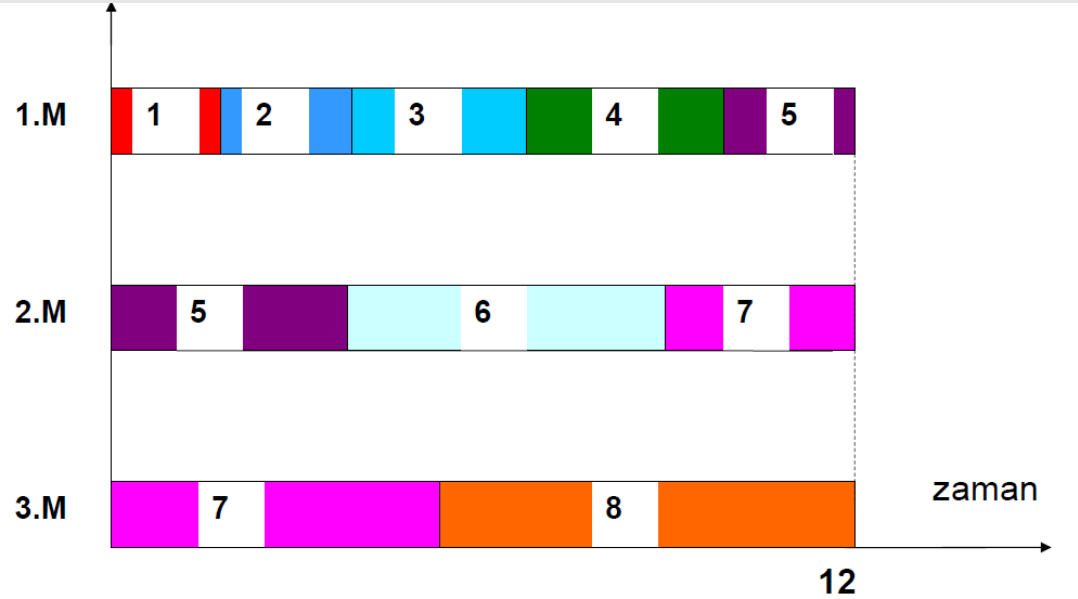
Görüldüğü üzere burada Cmax, en büyük işlem süresine sahip iş tarafından belirlenmektedir.

Paralel Makinelerde Öne Geçmeli Hal (preempitive)

- **Örnek:** Aşağıda verilen örnek problemi bu prosedür kapsamında SPT kuralı ile çözelim.

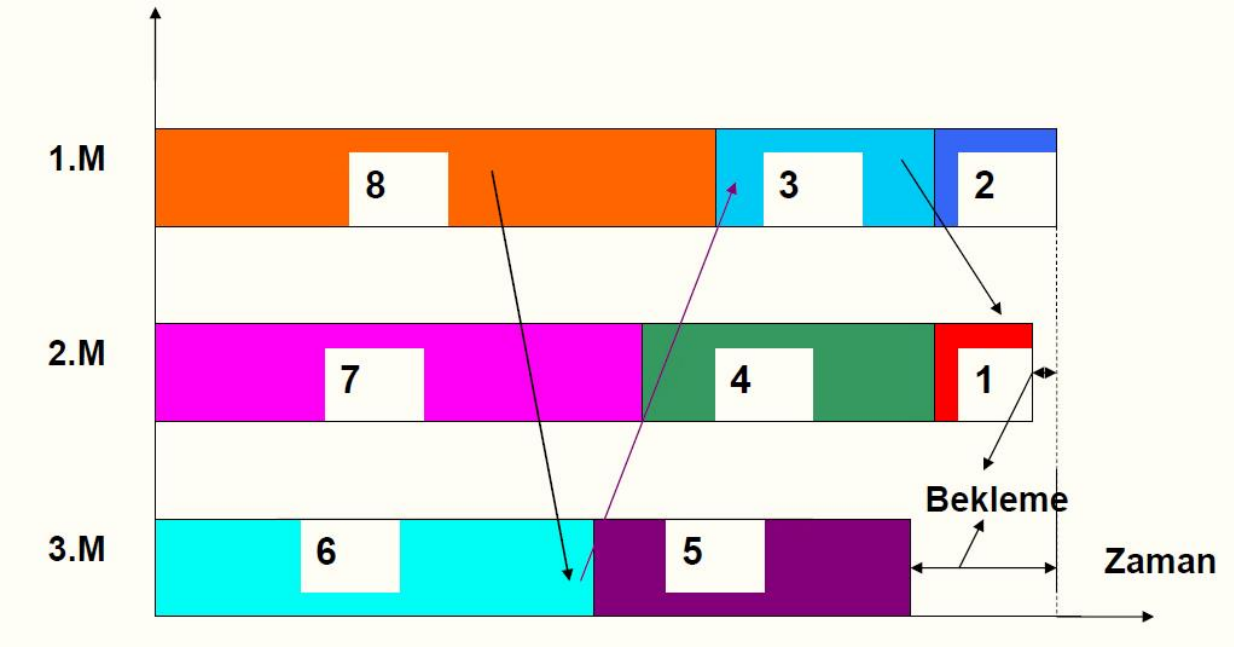
j	1	2	3	4	5	6	7	8
pj	1	2	3	4	5	6	7	8

- **Çözüm:** SPT sırası: 1,2,3,4,5,6,7,8



Paralel Makinelerde Öne Geçmeli Hal (preempitive)

- Aynı örnek **öne geçmesiz durumda LPT kuralı ile** çözülseydi elde edilen çözüm aşağıdaki gibi olacaktı:



Görüleceği üzere öne geçmeli hal ile elde edilen çözüm, öne geçmesiz duruma kıyasla belirgin şekilde avantaj ortaya koymaktadır.

Paralel Makinelerde Öne Geçmeli Hal (preempitive)

Longest Remaining Time First (LRPT)

- Bu kuralda geri kalan işlem süresi en büyük olan işe öncelik tanınır.
- **Örnek:**

job (j)	j_1	j_2	j_3
p_j	8	7	6

Using tabular method we will generate the schedule. One time unit in each row increases time. The jobs are given priority for assigning to machines M_1 and M_2 according to LRPT rule. Jobs having largest values of remaining process time are assigned to machines. If jobs with small value of remaining process time are currently assigned to machines, these jobs will be removed if jobs with higher values of LRPT are waiting in queue for their turn (Remember, preemptions are allowed). The complete procedure is presented in Table 3.10.

Table 3.10 Generation of schedule For $P_2 \mid prmp \mid C_{max}$ Problem.

start time	Remaining Process Time j1	j2	j3	Schedule Time	Job on M1	Job on M2
0	8	7	6	[0 , 1]	$j_1 \leftarrow$	$j_2 \leftarrow$
1	7	6	6	[1 , 2]	j_1	$j_2 \rightarrow$
2	6	5	6	[2 , 3]	j_1	$j_3 \leftarrow$
3	5	5	5	[3 , 4]	j_1	$j_3 \rightarrow$
4	4	5	4	[4 , 5]	$j_1 \rightarrow$	$j_2 \leftarrow$
5	3	4	4	[5 , 6]	$j_3 \leftarrow$	j_2
6	3	3	3	[6 , 7]	$j_3 \rightarrow$	j_2
7	3	2	2	[7, 8]	$j_1 \leftarrow$	$j_2 \rightarrow$
8	2	1	2	[8 , 9]	j_1	$J_3 \leftarrow$
9	1	1	1	[9 , 10]	$j_1 \rightarrow$	$j_3 \rightarrow$

Paralel Makinelerde Öne Geçmeli Hal (preempitive)

start time	Remaining Process Time j1	j2	j3	Schedule Time	Job on M1	Job on M2
10	0	1	0	[10 , 11]	$j_2 \leftarrow$	

($\rightarrow$) indicates job's removal from the machine and ($\leftarrow$) indicates job's loading on the machine

Time, t =0 and time interval, [0, 1]

As shown in Table, jobs j_1 and j_2 have large values of remaining process time (RPT) than job j_3. So, assign j_1 to M_1 and, j_2 to M_2 from time interval [0, 1].

Time, t =1 and time interval, [1, 2]

The values of remaining process time for jobs j_1 and j_2 are updated. Remaining process time (RPT) for job j_1 is still largest for job j_1. So, job j_1 remains assigned to machine M_1. However, jobs j_2 and j_3 have equal amount of remaining process time. Since, job j_2 is currently assigned to machine M_2, it is kept assigned to M_2 during interval [1, 2].

Time, t=2 and time interval, [2, 3]

At the start of time 2, remaining process times for jobs j_1, j_2 and j_3 are 6, 5 and 6, respectively. Keep job j_1 assigned to M_1. Since job 3 has larger value of RPT than job 2, *remove job j_2 from M_2 and, assign j_3 to M_2 during time interval [2, 3].* Preemption of job j_2 by j_3 is shown by arrows $\rightarrow$ and $\leftarrow$.

Paralel Makinelerde Öne Geçmeli Hal (preempitive)

Increase time by one unit. Reduce the remaining process time accordingly for the jobs currently assigned to machines M_1 and M_2.

Complete the procedure till all jobs have RPT equal to zero as shown in last row of the Table 3.10.

Final schedule is shown by Gantt chart in Figure 3.14.

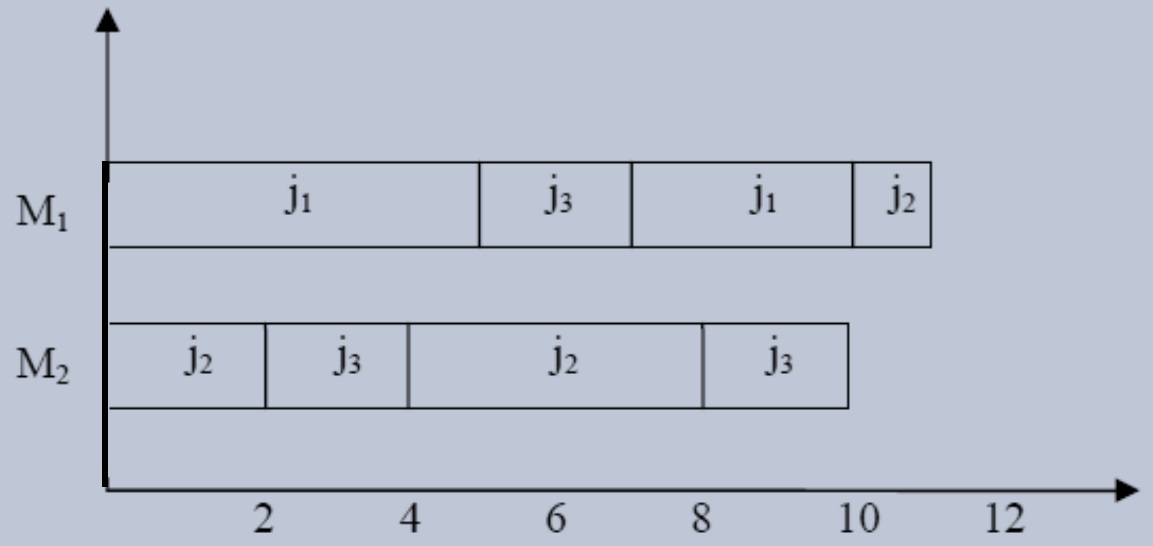


Figure 3.14 Gantt chart for LRPT schedule.

Farklı Hızdaki Paralel Makinelerde Öne Geçmeli Hal (preempitive) – Yayılma Zamanını Minimize Etmek

- Farklı hızdaki paralel makinelerde yayılma zamanını minimize etmek için en fazla kalan işlem süresi (Longest Remaining Processing Time – LRPT) değerine sahip olan iş en hızlı makineye (fastest machine-FM) atanır. Bu prosedür **LRPT-FM kuralı** olarak adlandırılır.
- En hızlı makinede her bir iş tamamlandıktan sonra, ikinci en hızlı makinedeki iş birinci makineye aktarılır, varsa üçüncü en hızlı makinedeki iş ikinci makineye aktarılır ve bu şekilde devam eder.

Örnek: Farklı hızlarda ve birbirine paralel üç adet makine ve bu makinelerde yapılması gereken işlere ait bilgiler aşağıdaki tablolarda verilmiştir. Öne geçmeli hali dikkate alarak Cmax'ı minimize etmek amacıyla LRPT-FM kuralından yararlanıp işleri çizelgeleyiniz ($Q_m|prmp|C_{max}$).

Machine	M_1	M_2	M_3
Speed (v_i)	3	2	1

job	j_1	j_2	j_3
p_j	36	34	12

Farklı Hızdaki Paralel Makinelerde Öne Geçmeli Hal (preempitive) – Yayılma Zamanını Minimize Etmek

- Kuralın işletilmesine yönelik adımlar aşağıdaki tabloda verilmiştir.

Kalan İşlem Zamanı

start time	Remaining P. T. j_1	j_2	j_3	Schedule Time	Job on M_1 (3)	Job on M_2 (2)	Job on M_3 (1)
0	36	34	12	[0 , 1]	j_1	j_2	j_3
1	33	32	11	[1 , 2]	j_1	j_2	j_3
2	30	30	10	[2 , 3]	j_1	j_2	j_3
3	27	28	9	[3 , 4]	j_2	j_1	j_3
4	25	25	8	[4 , 5]	j_2	j_1	j_3
5	23	22	7	[5 , 6]	j_1	j_2	j_3
6	20	20	6	[6 , 7]	j_1	j_2	j_3

Farklı Hızdaki Paralel Makinelerde Öne Geçmeli Hal (preempitive) – Yayılma Zamanını Minimize Etmek

start time	Remaining P. T. j_1	j_2	j_3	Schedule Time	Job on M_1 (3)	Job on M_2 (2)	Job on M_3 (1)
7	17	18	5	[7, 8]	j_2	j_1	j_3
8	15	15	4	[8 , 9]	j_2	j_1	j_3
9	13	12	3	[9 , 10]	j_1	j_2	j_3
10	10	10	2	[10 , 11]	j_1	j_2	j_3
11	7	8	1	[11 , 12]	j_2	j_1	j_3
12	5	5	-	[12 , 13]	j_2	j_1	-
13	3	2	-	[13 , 14]	j_1	j_2	-
14	-	-	-				

- Tablo incelendiğinde, öne geçme durumlarının J1 ve J2 arasında t=3, 5, 7, 9, 12 ve 14 zamanlarında M1 ve M2 makinalarında gerçekleştiği görülmektedir.

Farklı Hızdaki Paralel Makinelerde Öne Geçmeli Hal (preempitive) – Toplam Tamamlanma Zamanını Minimize Etmek

- Farklı hızdaki paralel makinelerde işlerin toplam tamamlanma zamanını minimize etmek için, kalan en küçük (kısa) SRPT (Shortest Remaining Processing Time) değerine sahip olan iş en hızlı makineye (fastest machine-FM) atanır. Öne geçmeli durum geçerlidir. Bu prosedür **SRPT-FM kuralı** olarak adlandırılır.
- En hızlı makinede her bir iş tamamlandıktan sonra, ikinci en hızlı makinedeki iş birinci makineye aktarılır.

Farklı Hızdaki Paralel Makinelerde Öne Geçmeli Hal (preempitive) – Toplam Tamamlanma Zamanını Minimize Etmek

- Örnek: Aşağıdaki 4 makine 7 iş problemini dikkate alalım:

Machine	M_1	M_2	M_3	M_4
speed (vi)	4	2	2	1

job (j)	j_1	j_2	j_3	j_4	j_5	j_6	j_7
pj	8	16	34	40	45	46	61

- İşler süreleri bakımından artan şekilde sıralanır. Benzer şekilde makineler de hızları bakımından en hızlıdan en yavaşa doğru sıralanır.
- t=0 iken; J1 işi M1'e, J2 işi M2'ye, J3 işi M3'e ve J4 işi M4'e atanır.
- t=2 iken; J1 işi tamamlanmıştır. J2 işi M1'e aktarılır ve J3 işi M2'ye aktarılır. J5 işi de M4'e atanır.
- t=5 iken; J2 işi M1'de tamamlanır. J3 işi M1'e, J4 işi M2'ye, J5 işi de M3'e aktarılır. J6 işi M4'e atanır.
- Bu şekilde tüm işler tamamlanana kadar devam edilir. Tüm işlerin atanmasına yönelik hazırlanmış tablo aşağıda verilmiştir.

Farklı Hızdaki Paralel Makinelerde Öne Geçmeli Hal (preempitive) – Toplam Tamamlanma Zamanını Minimize Etmek

start time	Remaining P. T. j_1	j_2	j_3	j_4	Schedule Time	Job on M_1 (4)	Job on M_2 (2)	Job on M_3 (2)	Job on M_4 (1)
0	8	16	34	40	[0 , 1]	j_1	j_2	j_3	j_4
1	4	14	32	39	[1 , 2]	j_1	j_2	j_3	j_4
2 ←	j_2 12	j_3 30	j_4 38	j_5 45	[2 , 3]	j_2	j_3	j_4	j_5
3	8	28	36	44	[3 , 4]	j_2	j_3	j_4	j_5
4	4	26	34	43	[4 , 5]	j_2	j_3	j_4	j_5
5 ←	j_3 24	j_4 32	j_5 42	j_6 46	[5 , 6]	j_3	j_4	j_5	j_6
6	20	30	40	45	[6 , 7]	j_3	j_4	j_5	j_6
7	16	28	38	44	[7, 8]	j_3	j_4	j_5	j_6
8	12	26	36	43	[8 , 9]	j_3	j_4	j_5	j_6
9	8	24	34	42	[9 , 10]	j_3	j_4	j_5	j_6

Farklı Hızdaki Paralel Makinelerde Öne Geçmeli Hal (preempitive) – Toplam Tamamlanma Zamanını Minimize Etmek

start time	Remaining P. T.				Schedule Time	Job on M_1 (4)	Job on M_2 (2)	Job on M_3 (2)	Job on M_4 (1)
10	4	22	32	41	[10 , 11]	j_3	j_4	j_5	j_6
11 ←	**j_4** 20	**j_5** 30	**j_6** 40	**j_7** 61	[11 , 12]	j_4	j_5	j_6	j_7
12	16	28	38	60	[12 , 13]	j_4	j_5	j_6	j_7
13	12	26	36	59	[13 , 14]	j_4	j_5	j_6	j_7
14	8	24	34	58	[14 , 15]	j_4	j_5	j_6	j_7
15	4	22	32	57	[15 , 16]	j_4	j_5	j_6	j_7
16 ←	**j_5 20**	**j_6 30**	**j_7 56**		[16 , 17]	j_5	j_6	j_7	
17	16	28	54		[17 , 18]	j_5	j_6	j_7	
18	12	26	52		[18 , 19]	j_5	j_6	j_7	
19	8	24	50		[19 , 20]	j_5	j_6	j_7	
20	4	22	48		[20 , 21]	j_5	j_6	j_7	

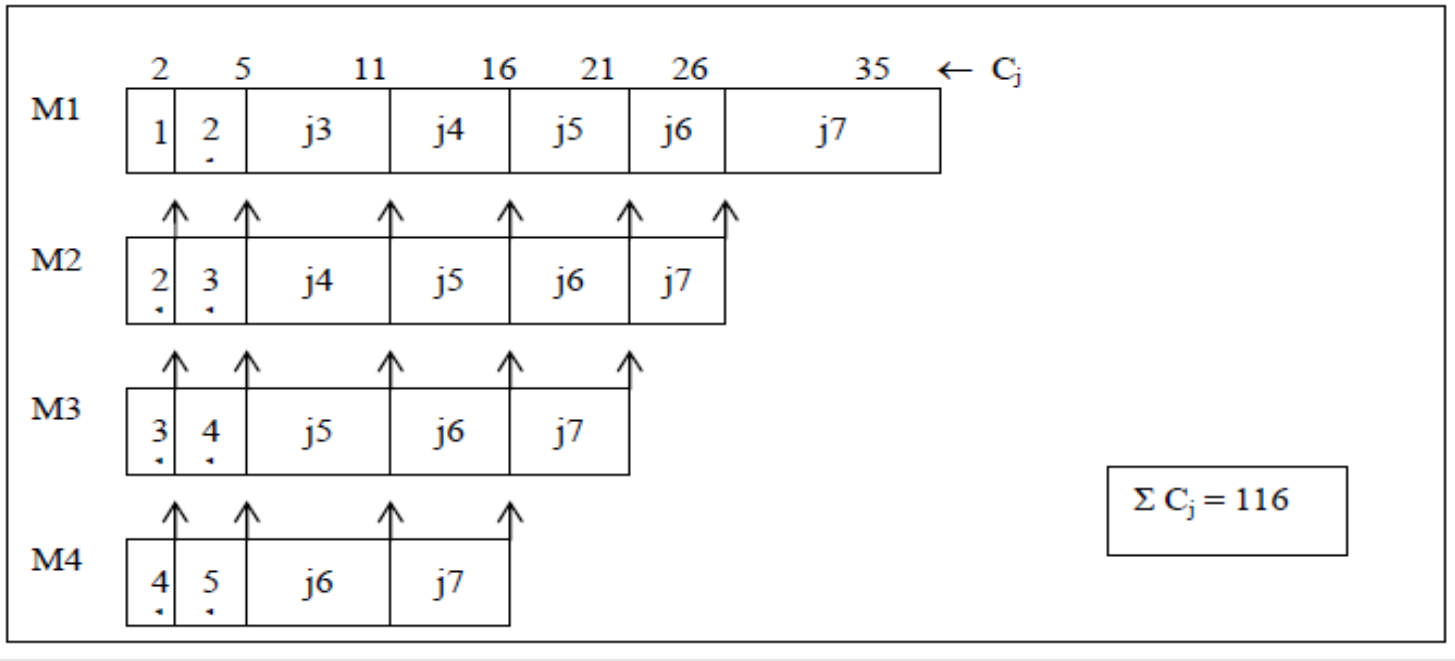
Farklı Hızdaki Paralel Makinelerde Öne Geçmeli Hal (preempitive) – Toplam Tamamlanma Zamanını Minimize Etmek

21 ←	**j_6** 20	**j_7** 46	[21 , 22]	j_6	j_7	
22	16	44	[22 , 23]	j_6	j_7	
23	12	42	[23, 24]	j_6	j_7	
24	8	40	[24 , 25]	j_6	j_7	
25	4	38	[25 , 26]	j_6	j_7	
26 ←		**j_7** 36	[26 , 27]	j_7		
27		32	[27 , 28]	j_7		
28		28	[28 , 29]	j_7		
29		24	[29 , 30]	j_7		
30		20	[30 , 31]	j_7		
31		16	[31 , 32]	j_7		

Farklı Hızdaki Paralel Makinelerde Öne Geçmeli Hal (preempitive) – Toplam Tamamlanma Zamanını Minimize Etmek

start time	Remaining P. T.	Schedule Time	Job on M_1 (4)	Job on M_2 (2)	Job on M_3 (2)	Job on M_4 (1)
32	12	[32 , 33]	j_7			
33	8	[33 , 34]	j_7			
34	4	[34 , 35]	j_7			
35	-	[35 , 36]				

- Elde edilen çözümün Gantt şeması yanda sunulmuştur.



Paralel Makinelerde Öne Geçmeli Hal (preempitive) – En Büyük Gecikmeyi Minimize Etmek

- Bu problem öne geçmeli halin geçerli olduğu durumda maksimum gecikmeyi minimize etmeyi amaçlamaktadır.
- Bu problemin çözümünde biraz farklı bir yol izlenir.
  - *Öncelikle problem, işlerin r_j zamanlarının olduğu ve C_{max}'ın minimize edilmeye çalışıldığı bir problem gibi düşünülür ve LRPT kuralı yardımıyla paralel makineler üzerinde bir çizelge oluşturularak Gantt şeması çizilir.*
  - *Sonrasında, elde edilen Gantt şeması tersine çevrilir. Böylece, L_{max}'ı minimize eden bir çizelge elde edilmiş olur.*

Örnek: Aşağıda, teslim tarihleri ve işlem süreleri verilen işleri, öne geçmeli halin geçerli olduğu aynı hızdaki iki adet paralel makine üzerinde en büyük gecikmeyi (L_{max}) minimize etmek amacıyla çizelgeleyiniz ($P_2|prmp|L_{max}$).

Jobs (j)	j_1	j_2	j_3	j_4
d_j	4	5	8	9
p_j	3	3	3	8

Paralel Makinelerde Öne Geçmeli Hal (preempitive) – En Büyük Gecikmeyi Minimize Etmek

- İşlerin sanki r_j değerleri varmış gibi, r_j değerleri aşağıdaki şekilde hesaplanır:

$$d^* = \max(d_1, d_2, d_3, d_4) = (4, 5, 8, 9) = 9$$

$$r_4 = d^* - d_4 = 9 - 9 = 0,$$
$$r_3 = d^* - d_3 = 9 - 8 = 1,$$
$$r_2 = d^* - d_2 = 9 - 5 = 4,$$
$$r_1 = d^* - d_1 = 9 - 4 = 5$$

- Burada elde edilen r_j değerleri kullanılarak problem aşağıdaki gibi yeniden formüle edilebilir:

job	j_1	j_2	j_3	j_4
r_j	5	4	1	0
p_j	3	3	3	8

d_j değerleri artık yokmuş gibi davranılır.

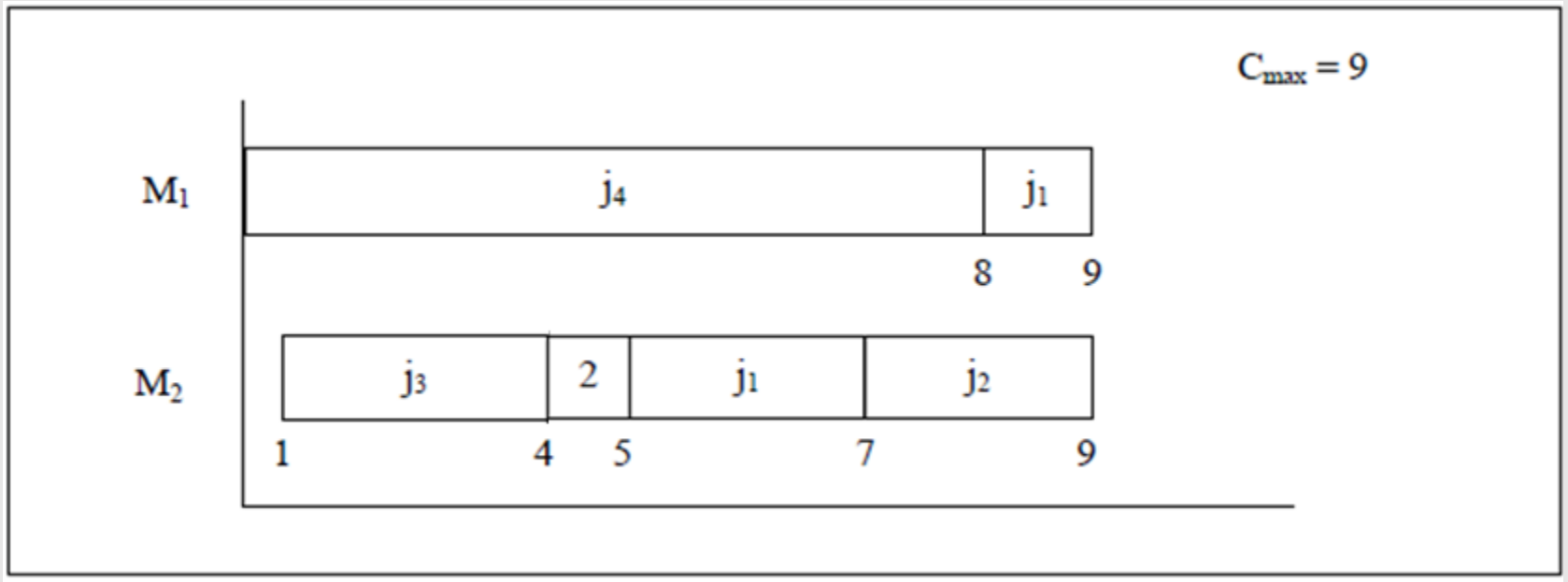
Paralel Makinelerde Öne Geçmeli Hal (preempitive) – En Büyük Gecikmeyi Minimize Etmek

- LRPT kuralıyla elde edilen çizelge aşamalar halinde aşağıda gösterilmiştir.

Start time	Remaining Process Time j_1 $r_1=5$	j_2 $r_2=4$	j_3 $r_3=1$	j_4 $r_4=0$	Schedule Time	Job on M_1	Job on M_2
0	3	3	3	(8)	[0 , 1]	j_4	
1	3	3	(3)	(7)	[1 , 2]	j_4	j_3
2	3	3	(2)	(6)	[2 , 3]	j_4	j_3
3	3	3	(1)	(5)	[3 , 4]	j_4	j_3
4	3	(3)		(4)	[4 , 5]	j_4	j_2
5	(3)	2		(3)	[5 , 6]	j_4	j_1
6	(2)	2		(2)	[6 , 7]	j_4	j_1
7	1	(2)		(1)	[7, 8]	j_4	j_2
8	(1)	(1)			[8 , 9]	j_1	j_2
9					[9 , 10]		

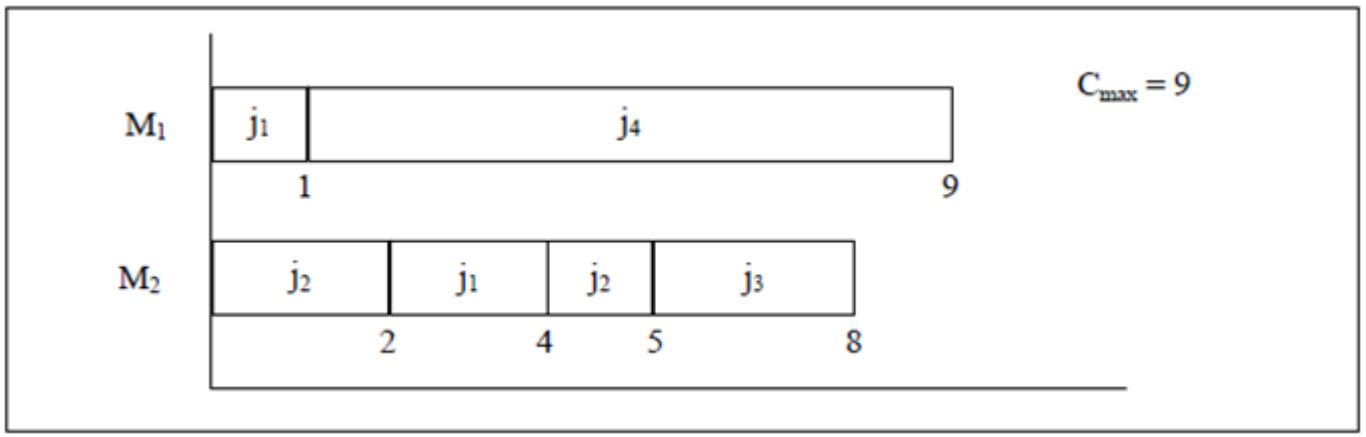
Paralel Makinelerde Öne Geçmeli Hal (preempitive) – En Büyük Gecikmeyi Minimize Etmek

- Görüldüğü üzere hiçbir iş r_j zamanından daha önce başlamamaktadır. Örneğin J1, ancak t=5'ten sonra başlayabilmektedir.



Paralel Makinelerde Öne Geçmeli Hal (preempitive) – En Büyük Gecikmeyi Minimize Etmek

- Gantt şeması ters çevrilerek (aynadaki görüntüsü gibi) işlerin başlama ve bitiş zamanları yeniden düzenlenirse aşağıdaki gibi elde edilecektir.



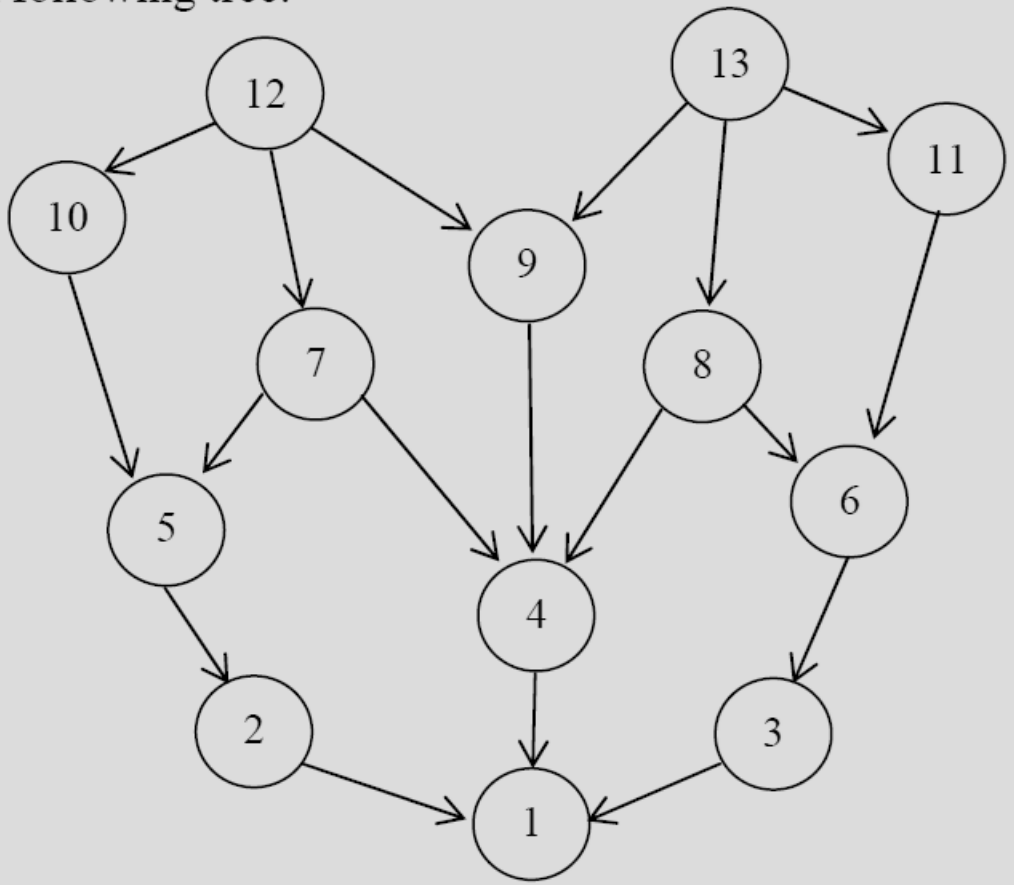
- İşlerin teslim tarihleri ve tamamlanma zamanları dikkate alındığında gecikme süreleri aşağıdaki gibi hesaplanır. Görüldüğü üzere tüm işler tam zamanında tamamlanmıştır, gecikme yoktur ($L_{max}=0$).

J	D_j	C_j	L_j
1	4	4	0
2	5	5	0
3	8	8	0
4	9	9	0

ÖDEV: Aynı problemi, EDD benzeri bir kuralı modifiye ederek nasıl çözebiliriz?

Bölüm Sonu Soruları

- 1. Consider an instance of $P_4 \mid p_j=1, \text{tree} \mid C_{max}$ problem with precedence constraints as shown in following tree.

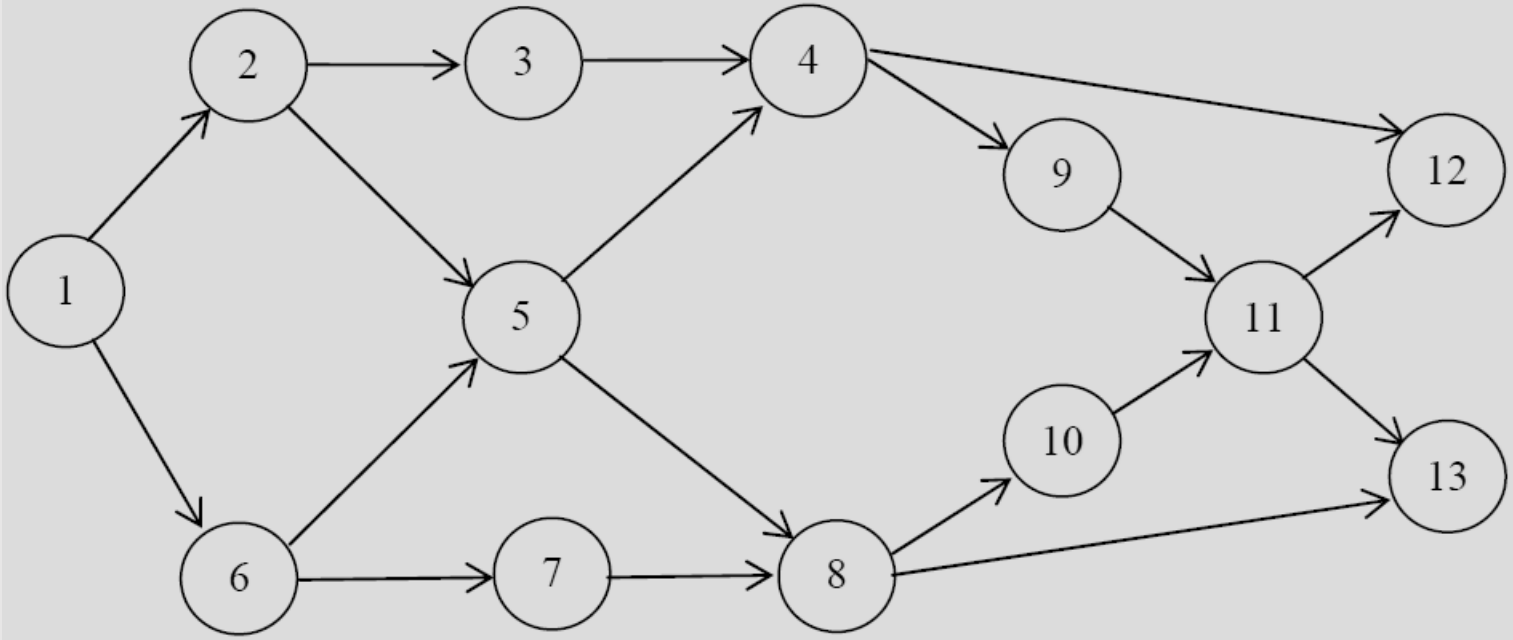


Generate schedule and find C_{max}, using
i) CP rule, ii) LNS rule

2. Consider an instance of the $P_\infty \mid prec \mid C_{max}$ problem with data for 13 jobs as follows:

Job	1	2	3	4	5	6	7	8	9	10	11	12	13
p_j	7	8	4	9	12	5	3	9	5	12	7	5	8

The precedence graph is shown below.



Assume infinite number of machines, and find C_{max} as well as identify critical path.

- 3. Consider an instance of $P_3 \mid\mid C_{max}$ problem with data as under:

Job	1	2	3	4	5	6	7	8
p_j	15	11	9	6	12	18	9	14

i) Apply LPT rule and assign jobs to machines. Is the solution optimal? Why or why not.
ii) Apply load balancing heuristics and find C_{max}.

- 4. The following data represents an instance of $P_2 \mid prmp \mid C_{max}$ Problem

job	1	2	3	4	5
p_j	4	3	13	12	2

Generate schedule to minimize C_{max}. Also draw Gantt chart. Preemptions are allowed at discrete intervals of time

5. Consider $P_4 \mid\mid C_{max}$ problem with the following data:

Job	1	2	3	4	5	6	7	8	9	10	11
p_j	2	2	3	4	5	6	7	8	2	2	3

Find a sequence that minimize C_{max} and compute C_{max}. Then, prove that the optimal sequence found is optimal by a principle.

6. Consider the following data pertaining to problem as an instance of $Q_3 \mid prmp \mid C_{max}$ There are three machines in parallel with varying speeds as follows:

Machine	M_1	M_2	M_3
Speed (v)	2	1	4

Four jobs are to be processed with process time as follows:

Jobs	1	2	3	4
Process time	12	8	16	24

Generate schedule applying LRPT-FM rule in discrete intervals of time.

7. Consider 3-machine and 6-job problem with an instance of $Q_3 \mid prmp \mid \Sigma C_j$.

Machine	M_1	M_2	M_3
Speed (v_i)	6	4	2

Six jobs are to be processed with process time as follows:

Job	1	2	3	4	5	6
P_j	12	16	33	48	56	38

Find optimal schedule using SRPT-FM rule. Preemptions are allowed at discrete point in time

8. The following data is an instance of $P_3 \mid prmp \mid L_{max}$

Job	1	2	3	4	**5**
P_j	4	2	1	6	7
D_j	5	7	3	14	12

Generate a schedule to minimize $L_{max.}$ Preemptions are allowed at discrete intervals of time.

9. Consider $Q_2 \mid prmp \mid \Sigma W_j C_j$ Problem with the following data:

Job	1	2	3	4	5
Process time	17	5	9	12	8
Weight	1	2	4	6	4

In which the two machines processing speeds are as follows:

Machine	M_1	M_2
Processing Speed (v_i)	1	2

Generate a sequence to minimize the total weighted completion time. Then, constructing the Gantt chart for the solution obtained and compute the total weighted completion time.

10. Consider $P_2 \mid \mid n_t$ Problem with the following data:

Job	1	2	3	4	5	6	7	8	9
D_j	8	12	7	3	17	14	16	23	22
P_j	7	5	9	2	8	8	5	2	1

Generate a sequence to minimize the total number of jobs tardy and then compute the total number of jobs tardy and the makespan.

Uygulama

Matematiksel Modelleme & GAMS

- Tek Makine – Min F
- Tek Makine - Min T
- Paralel Makine - Min Cmax

LEKIN

- Tek Makine
- Paralel Makine

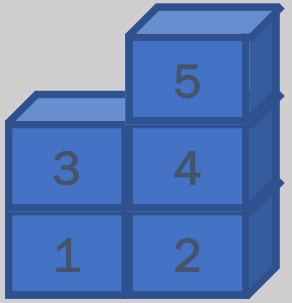
Akış Tipi Atölyelerde Çizelgeleme

-Johnson Algoritması

-Palmer Sezgiseli

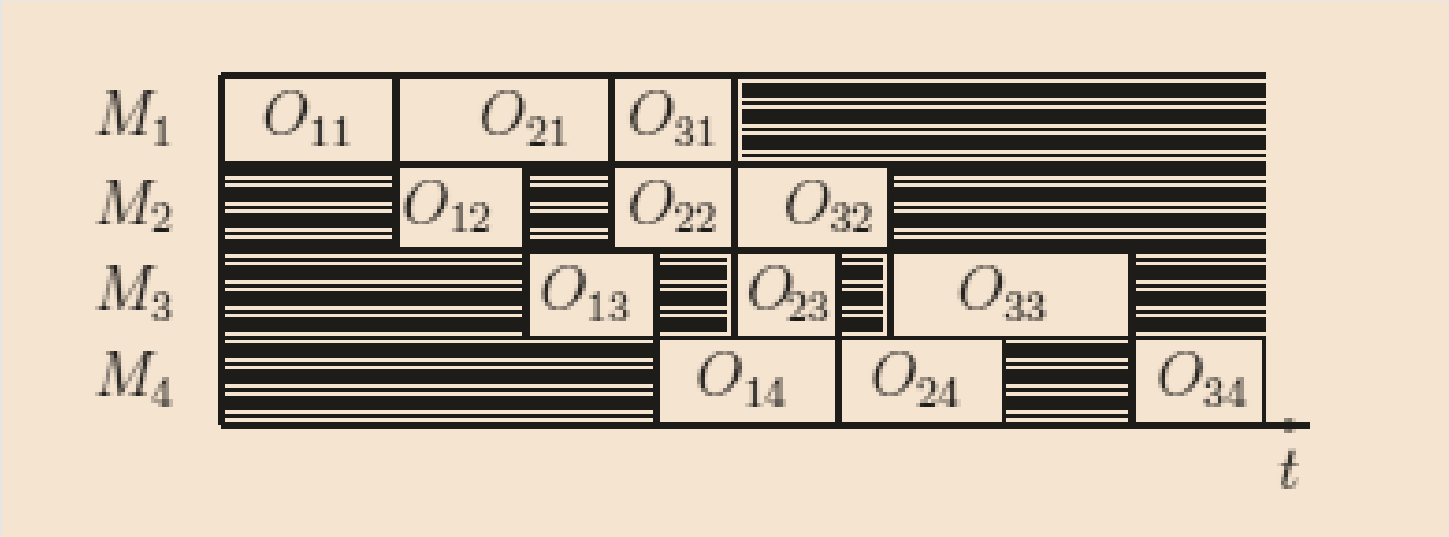
-CDS Algoritması

-NEH Algoritması



Akış Tipi Atölyelerde Çizelgeleme

- **Akış tipi** atölyelerde makinelerin, işlerin yapılış biçiminden kaynaklanan, doğal bir sırası vardır.
- Teknolojik baskılar işlerin makineler arasındaki akışını belirler. Bu akış bütün makineler için benzer şekildedir.
- Şöyle ki; eğer J1 işinin M4 makinesinde M3 makinesinden daha önce işlenmesi gerekiyorsa, bu durum diğer bütün işler için de aynen geçerlidir.



Akış Tipi Atölyelerde Çizelgeleme

- n/m/F/Fmax
- n iş problemi ve m makinede çözüm zordur.
- Akış tipi atölye problemi incelenirse;

  n/2/F/Fmax veya n/2/F/Cmax problemi çözülmelidir.
- $C = r + F$
- $F = Toplam\ F_{ij} = W_{ij} + P_{ij}$

Johnson Algoritması – n iş 2 makine

- Johnson algoritması bir grup işin iki makinede (n iş, iki makine) sırasıyla çalışması sisteminde C_{max}'ı veya F_{max}'ı minimize etmek için optimum çözümü verir.
- Çalışma Prensibi:
  - *Birinci makineye, n işin içinden birinci makinedeki en kısa süreli olanları tahsis edersek, beklemeyi minimize ederiz.*
  - *İkinci makinedeki en kısa süreli işleri de en sona tahsis etmeye çalışırız.*

- **Johnson Sıralama Algoritması** olarak da bilinir.
- Tüm işler için r'ler SIFIR kabul edilir.

Johnson Algoritması – n iş 2 makine

- **1. Adım:** Yapılacak işlerin tümü bir listeye konulur. Buna A listesi diyelim. Ve iki tane boş liste oluşturulur. Bunlara da L1 ve L2 diyelim.
- **2. Adım:** Bütün alt işler arasında işlem süresi en küçük olan iş bulunur.
  - *Eğer bu süre 1. makineye aitse, bu iş L1 listesinin sonuna eklenir ve A listesinden çıkarılır.*
  - *Eğer bu süre 2. makineye aitse, bu iş L2 listesinin başına eklenir ve A listesinden çıkarılır.*
- **3. Adım:** A listesinde iş kalmayana kadar 2. adım tekrar edilir.
- **4. Adım:** L1 ve L2 listeleri birleştirilir.

Johnson Algoritması – Örnek 1

- 7/2/F/Cmax Problemi

İşlem Sırası: M1->M2

İş	M1	M2
1	6	3
2	2	9
3	4	3
4	1	8
5	7	1
6	4	5
7	7	6

L1 listesi: 4..
L2 listesi: ...

L1 listesi: 4...
L2 listesi: ...5

L1 listesi: 4–2 ...
L2 listesi: ...3–5 veya ...1–5

...

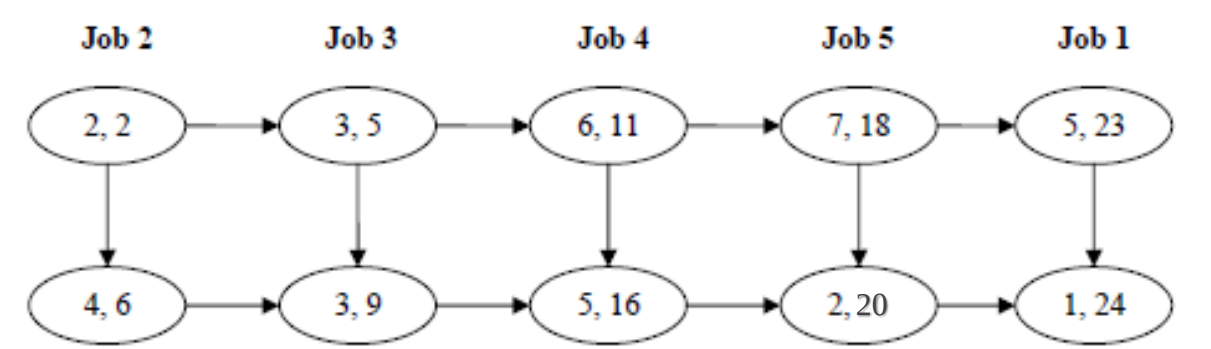
Nihai Sıralama: 4–2–6–7–3–1–5 veya 4–2–6–7–1–3–5

Johnson Algoritması – Örnek 2

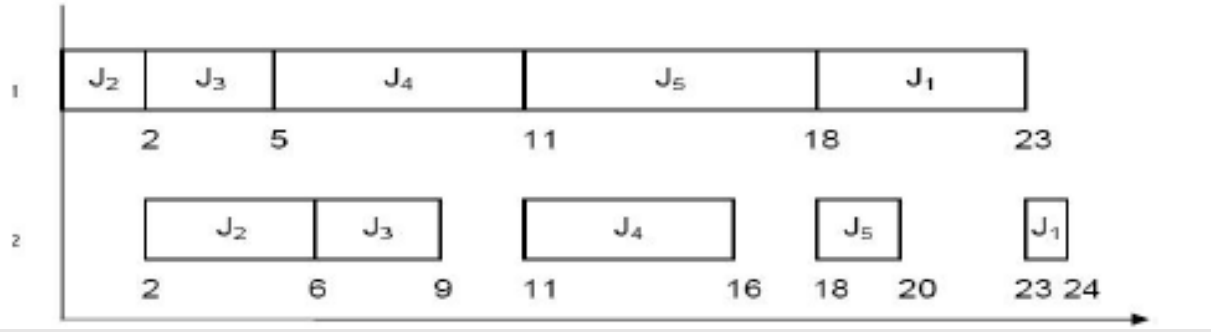
- 5/2/F/Cmax Problemi, İşlem Sırası: M1->M2

Job (j)	j_1	j_2	j_3	j_4	j_5
p_{1j}	5	2	3	6	7
p_{2j}	1	4	3	5	2

5 iş 2 makineden oluşan bu problem Johnson algoritmasıyla çözülmüş ve optimum çözüme ait yönlendirilmiş ağ ve Gantt diyagramları yandaki şekilde verilmiştir.



Şekil: J2, J3, J4, J5, J1 sıralaması için yönlendirilmiş ağ



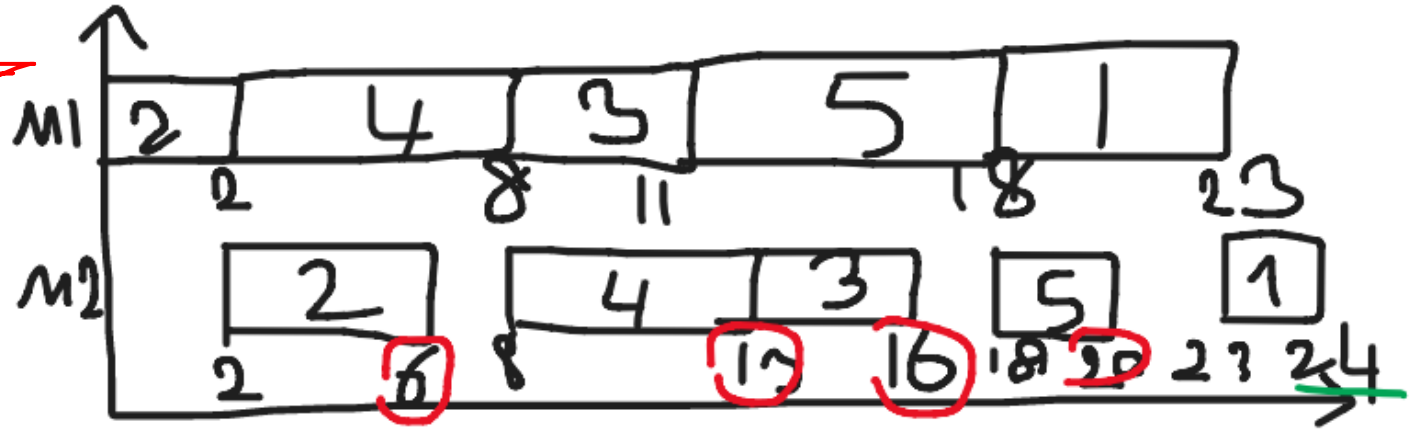
Şekil: J2, J3, J4, J5, J1 sıralaması için Gantt diyagramı

2-3-4-5-1

A1: 75

2 4-3-5-1

A2:



A2: 79

$F = 6 + 13 + 16 + 20 + 24 = 79$

Johnson Algoritması – n iş 3 makine

- n/3/F/Cmax problemi

Özel şartların sağlanması durumunda, n iş 3 makineden oluşan bir akış tipi çizelgeleme probleminde de Johnson algoritması kullanılabilir. Bunun için sağlanması gereken özel şartlar şunlardır:

$$Min_{i=1}^{n}\{P_{i1}\} \geq Max_{i=1}^{n}\{P_{i2}\}$$ VEYA

$$Min_{i=1}^{n}\{P_{i3}\} \geq Max_{i=1}^{n}\{P_{i2}\}$$ olmalı

- Yani Johnson algoritmasının uygulanabilmesi için aslında M2 makinesinin M1 veya M3 makineleri tarafından domine edilmesi (bastırılması) ve darboğaz oluşturmaması gerekiyor.
- Bu şartlar sağlanırsa,
  1. makine işlerinin süreleri, $a_i = P_{i1} + P_{i2}$
  2. makine işlerinin süreleri, $b_i = P_{i3} + P_{i2}$

olarak kabul edilir ve a_i, b_i süreleri kullanılarak Johnson algoritması ile iki kukla makine (M1' ve M2') üzerinden sıralanarak çizelgeleme yapılabilir.

Johnson Algoritması – Örnek 3

- 6/3/F/Cmax problemi

İş	M1	M2	M3
1	4	1	3
2	6	2	9
3	3	1	2
4	5	3	7
5	8	2	6
6	4	1	1

İki makinaya indirgeniyor

M1’	M2’
5	4
8	11
4	3
8	10
10	8
5	2

Johnson uygulanır. Sıralama:

2 – 4 – 5 – 1 – 3 – 6

Ya da

4 – 2 – 5 – 1 – 3 – 6

olarak bulunur.

Elde edilen çizelge için Gantt diyagramını çiziniz.

Johnson Algoritması – Örnek 4

- 4/3/F/Cmax problemi için işlerin makinelerdeki süreleri aşağıdaki tabloda verilmiştir.

Job (j)	Process time (M1)	Process time (M2)	Process time (M3)
1	8	2	4
2	5	4	5
3	6	1	3
4	7	3	2

M1 ve M3 makineleri için minimum işlem süreleri bulunur: 5 ve 2.
Bu sürelerin, M2 makinesi için maksimum işlem süresinden daha büyük olması şartı aranır.

Şart sağlandığı için Johnson algoritması ile çözüm aranabilir. Birleştirilmiş işlem süreleri aşağıdaki tabloda verilmiştir:

Job (j)	Process time (M_1)	Process time (M_2)
1	10	6
2	9	9
3	7	4
4	10	5

Bu halde optimum sıralama J2, J1, J4, J3 olarak elde edilir.

Bu sıralama için elde edilecek çizelgenin yönlendirilmiş ağ diyagramını çiziniz.

Peki n iş 3 makine için özel şartlar sağlanmazsa?

- Hatırlanacağı üzere Johnson algoritmasının kullanılabilmesi için aşağıdaki özel şartların sağlanması gerekmekteydi:

$$Min_{i=1}^{n}\{P_{i1}\} \geq Max_{i=1}^{n}\{P_{i2}\} \quad \text{VEYA} \quad Min_{i=1}^{n}\{P_{i3}\} \geq Max_{i=1}^{n}\{P_{i2}\}$$

- Johnson algoritması, bu özel şartların sağlanmadığı durumlarda da kullanılabilir ve iyi çözümler verir, fakat optimum çözümü garanti etmez.
- Bu tür genelleştirilmiş problemler için Johnson algoritması kullanılarak iyi bir başlangıç çözümü elde edilebilir ve bu çözüm başka prosedürler yardımıyla iyileştirilerek daha iyi çözümler aranabilir.

m>3 iken Cmax'ın minimize edilmesi

- n işin m makinede çizelgelenmesi gerektiği genel bir akış tipi çizelgeleme problemidir. Johnson algoritmasının geliştirilmesinin üzerinden yarım asırdan fazla zaman geçmiş olmasına karşın hala m>3 durumu için akış tipi çizelgeleme probleminde yayılma zamanını optimize edecek bir kural geliştirilememiştir.
- Bu durum kısmen, çalışılan problemin NP-complete (çözülmesi zor) yapısından kaynaklanmaktadır.
- Aşağıdaki algoritmalar, m>3 iken n işin çizelgelenmesi problemi için önerilmiş etkili algoritmalardandır.

ÖDEV

The below table gives the processing times (in hours) of seven jobs to be processed on four machines M1, M2, M3 and M4 in the order M1,M2,M3,M4. Sequence the given jobs using Johnson's method and find the overall processing time.

	M1	M2	M3	M4
A	3	1	4	12
B	8	0	5	15
C	11	3	8	10
D	4	7	3	8
E	5	5	1	10
F	10	2	0	13
G	2	5	6	9

$M_1, M_2, M_3 \cdots M_m$

(1) min processing time on machine M_1 $\geq$ max processing time on machines $M_2, M_3 \ldots M_{(m-1)}$

OR

(2) min processing time on machine M_m $\geq$ maximum processing time on machines $M_2, M_3, \ldots M_{(m-1)}$

A

B

C

Palmer Sezgiseli

- Bu sezgisel iki adımda uygulanır:
  1. *n iş m makineden oluşan bir statik akış tipi atölye sisteminde, her iş için A_j değerini hesaplayınız:*

$$A_j = -\sum_{i=1}^{m} \{m - (2i-1)\}\, p_{ij}$$

  2. *İşleri A_j değerlerine göre azalan şekilde sırala.*

Örnek: Tabloda verilen F3||Cmax problemini Palmer sezgiselini kullanarak çözünüz.

Machines	j_1	j_2	j_3	j_4
M_1	6	8	3	4
M_2	5	1	5	4
M_3	4	4	4	2

Palmer Sezgiseli

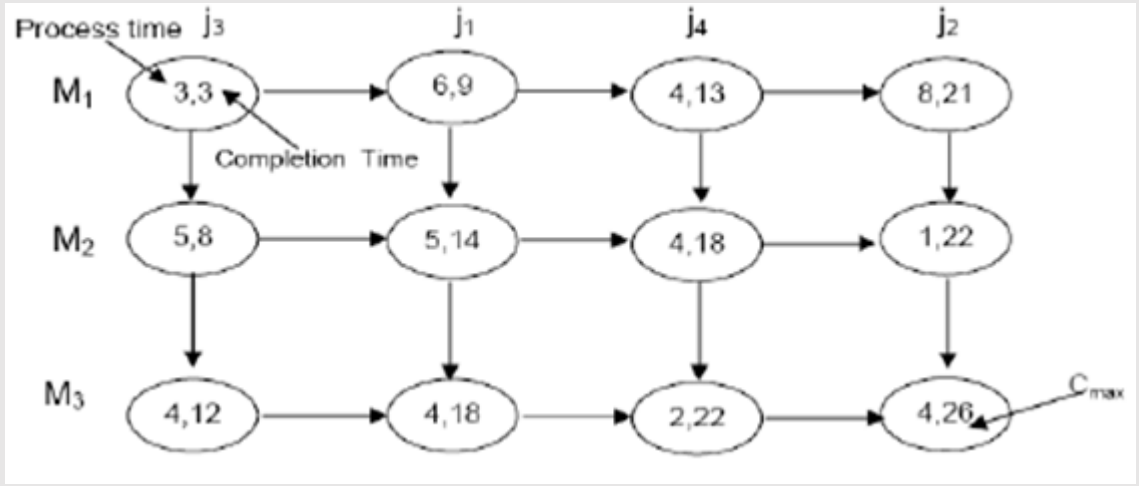
$$A_j = -\sum_{i=1}^{m}\{m-(2i-1)\}\,p_{ij} = -\sum_{i=1}^{3}\{3-(2i-1)\}\,p_{ij}$$

İşlerin A_j değerleri, aşağıdaki şekilde hesaplanır:

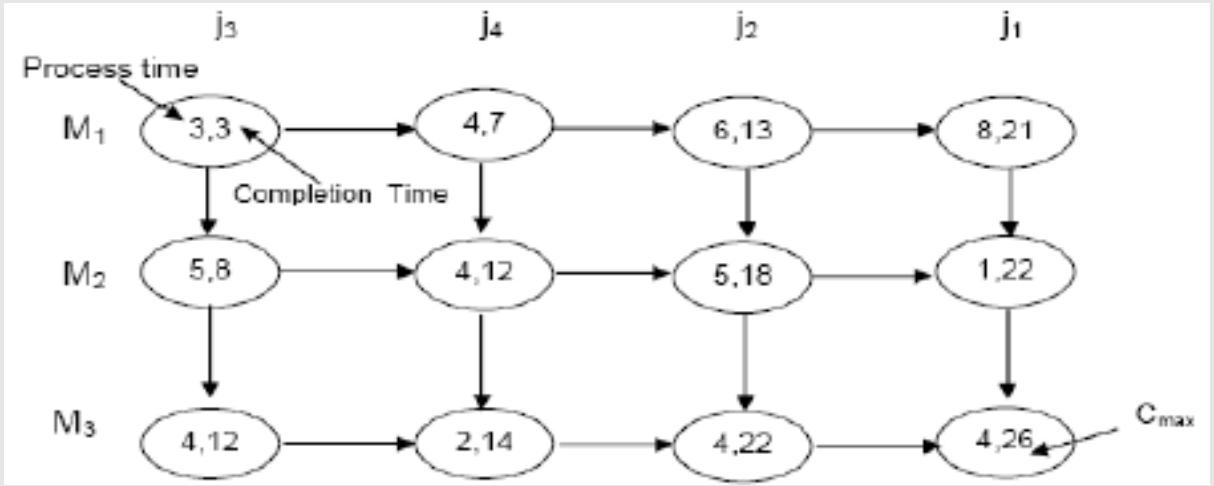
	i=1	i=2	i=3	
	3-(2-1) = 2	3-(2x2-1) = 0	3-(2x3-1) = -2	
Job j	p_{1j}	p_{2j}	p_{3j}	A_j
1	6	5	4	-4
2	8	1	4	-8
3	3	5	4	2
4	4	4	2	-4

A_j değerleri azalan şekilde sıralanırsa, aşağıdaki sıralamalar elde edilir:
Sıralama 1: {J3, J1, J4, J2} Sıralama 2: {J3, J4, J1, J2}

Palmer Sezgiseli



İlk sıralama için, {J3, J1, J4, J2}, elde edilen yönlendirilmiş ağ.



İkinci sıralama için, {J3, J4, J1, J2}, elde edilen yönlendirilmiş ağ.

Görüleceği üzere her iki çözüm için de elde edilen yayılma zamanı, Cmax=26 olmaktadır.

Campbell, Dudek ve Smith (CDS) Algoritması

- Bu algoritma, n iş m makineden (m>2) oluşan bir problemi, p adet n iş alt-problemine dönüştürür (p=m-1). Her bir problem Johnson algoritması kullanılarak çözülür ve Cmax değerleri bulunur.
- Alt problemlerden minimum Cmax değerini veren çizelge, genel çizelge olarak kabul edilir.
- Aşağıdaki 4 iş – 3 makine problemini dikkate alırsak:

Jobs	M_1	M_2	M_3
j_1	P_{11}	P_{21}	P_{31}
j_2	P_{12}	P_{22}	P_{32}
j_3	P_{13}	P_{23}	P_{33}
j_4	P_{14}	P_{24}	P_{34}

2 adet alt problem oluşacaktır.

Campbell, Dudek ve Smith (CDS) Algoritması

- İlk problemde, M1 makinesi M1’ kukla makinesi gibi, M3 makinesi ise M2’ kukla makinesi gibi düşünülerek Johnson algoritması ile çözülür.

Jobs	$M_1' = M_1$	$M_2' = M_3$
j_1	P_{11}	P_{31}
j_2	P_{12}	P_{32}
j_3	P_{13}	P_{33}
j_4	P_{14}	P_{34}

Campbell, Dudek ve Smith (CDS) Algoritması

- İkinci alt problem ise M1+M2'nin M1', M2+M3'ün M2' gibi dikkate alınması ile oluşturulur.

Jobs	$M_1' = M_1+M_2$	$M_2' = M_2+M_3$
j_1	$P_{11} + P_{21}$	$P_{21} + P_{31}$
j_2	$P_{12} + P_{22}$	$P_{22} + P_{32}$
j_3	$P_{13} + P_{23}$	$P_{23} + P_{33}$
j_4	$P_{14} + P_{24}$	$P_{24} + P_{34}$

CDS algoritması için genelleştirilmiş formül:

This implies that the surrogate problems data will be in generated as follows:

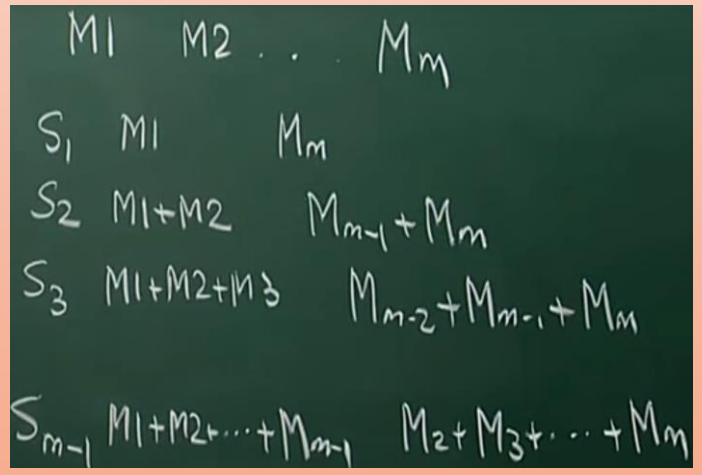
For k = 1, ……, m-1 and j = 1, ……, n then,

$$M_1' = \sum_{i}^{k} P_{ij} \text{ and } M_2' = \sum_{i=m-k+1}^{m} P_{ij}$$

Where:

M_1' = the processing time for the first machine

M_2' = the processing time for the second machine



Campbell, Dudek ve Smith (CDS) Algoritması

Örnek: Aşağıda verilen 4-iş 3-makineden oluşan akış tipi çizelgeleme problemini CDS algoritmasını kullanarak Cmax'ı minimize etmek amacıyla çözünüz.

	j_1	j_2	j_3	j_4
M_1	6	8	3	4
M_2	5	1	5	4
M_3	4	4	4	2

3 adet makine olduğu için m-1=2 adet alt problem oluşacaktır. CDS ile çözüm aşağıdaki gibi yapılır.

- Alt Problem-1: M1 makinesi M1' kukla makinesi, M3 makinesi de M2' kukla makinesi olarak düşünülür.

	j_1	j_2	j_3	j_4
$M_1' = M_1$	6	8	3	4
$M_2' = M_3$	4	4	4	2

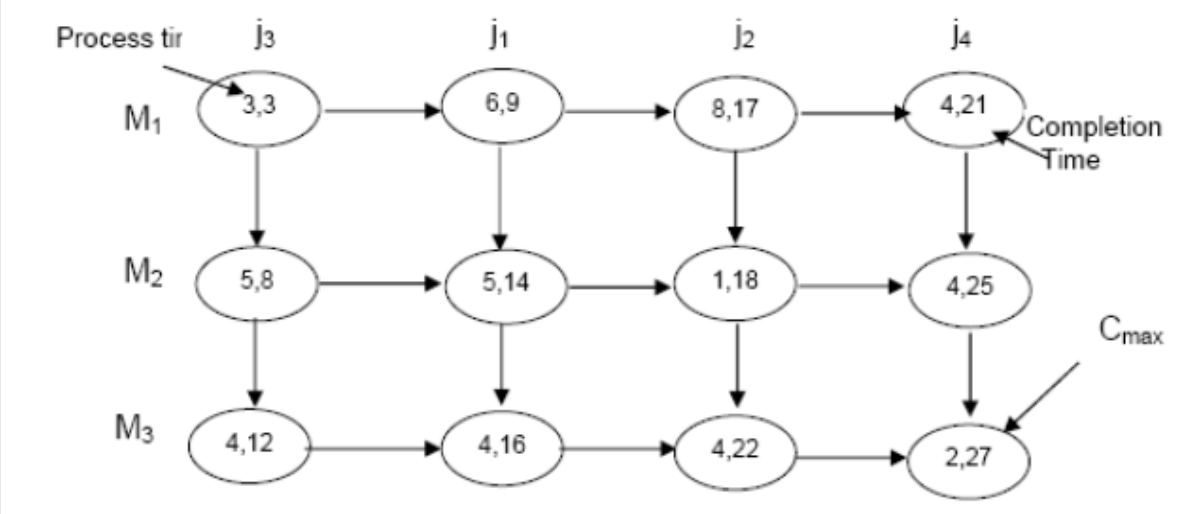
Johnson kuralı uygulanarak sıralama J3-J1-J2-J4 veya J3-J2-J1-J4 olarak elde edilir.

Campbell, Dudek ve Smith (CDS) Algoritması

İlk sıralamaya göre aşağıdaki gibi bir tablo elde edilir:

	j_3	j_1	j_2	j_4
$M_1' = M_1$	3	6	8	4
$M_2' = M_3$	4	4	4	2

Yönlendirilmiş ağ ile ilk sıralamanın çizelge olarak gösterilmesi aşağıdaki gibi olur:

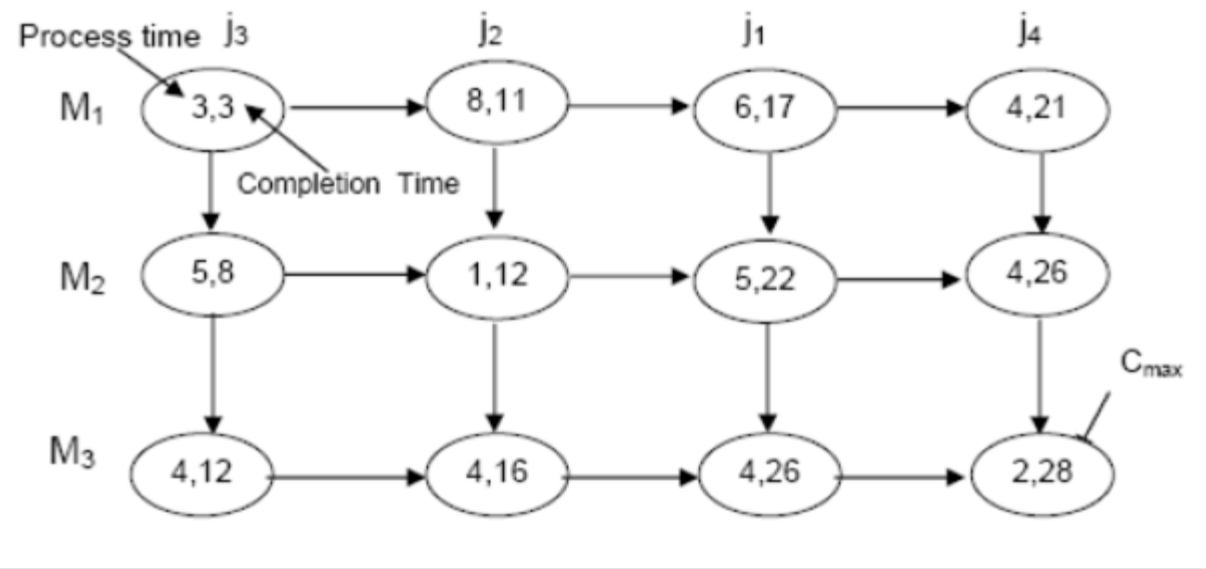


Campbell, Dudek ve Smith (CDS) Algoritması

İkinci sıralamaya göre aşağıdaki gibi bir tablo elde edilir:

	J_3	j_2	j_1	j_4
$M_1' = M_1$	3	8	6	4
$M_2' = M_3$	4	4	4	2

Yönlendirilmiş ağ ile ikinci sıralamanın çizelge olarak gösterilmesi ise aşağıdaki gibi olur:



Campbell, Dudek ve Smith (CDS) Algoritması

- Alt problem-2: Orijinal problemdeki veriler kullanılarak ikinci alt problem aşağıdaki gibi elde edilir:

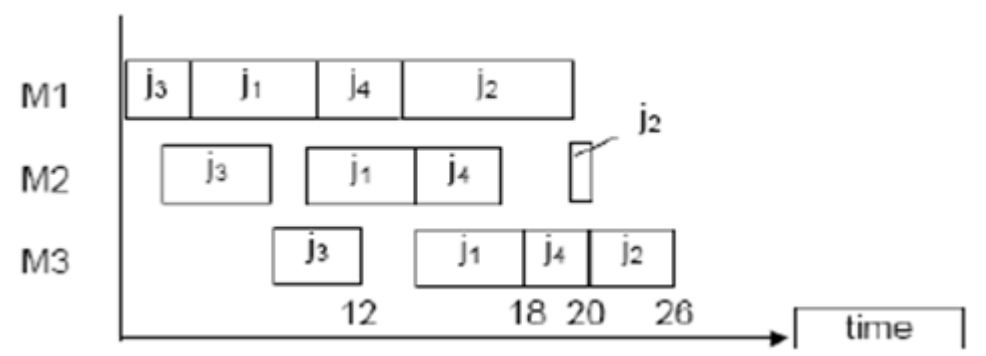
	j_1	j_2	j_3	j_4
$M_1'=M_1+M_2$	11	9	8	8
$M_2'=M_2+M_3$	9	5	9	6

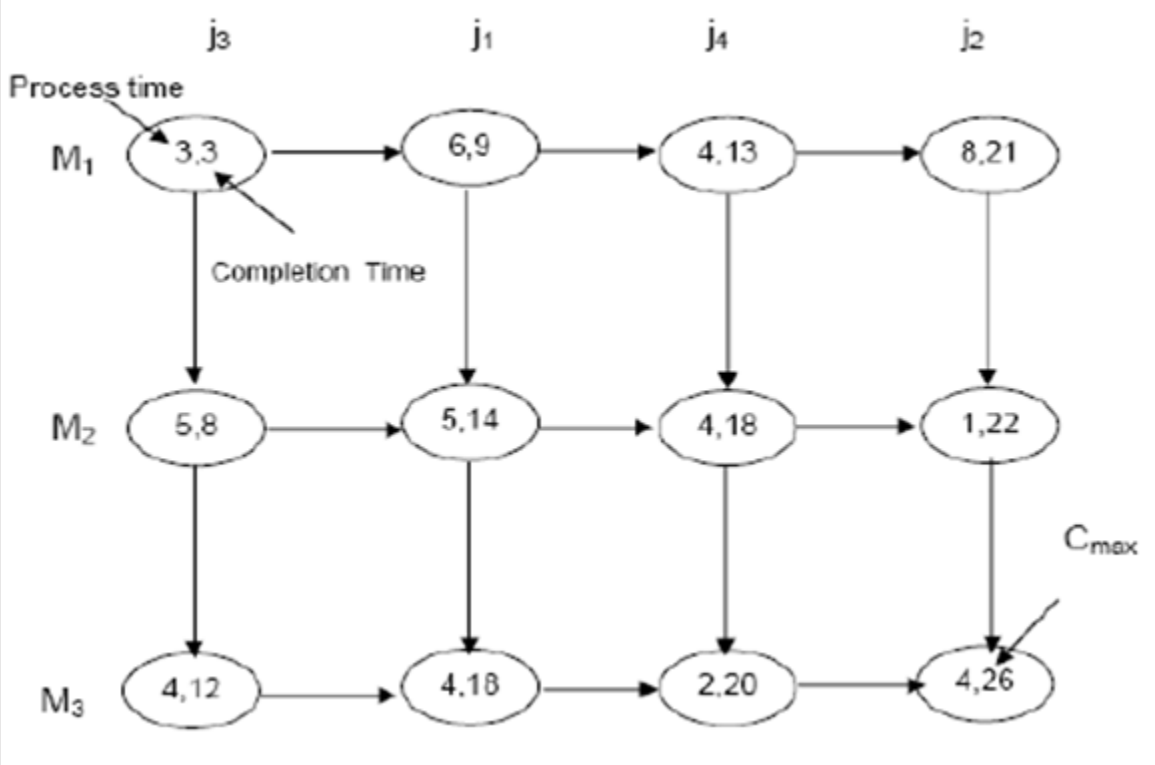
Bu durumda Johnson kuralı ile elde edilecek olan sıralama ise J3-J1-J4-J2 olur. Cmax ise tabloda gösterildiği gibi 26 zb olarak elde edilir.

Machine	j_3	j_1	j_4	j_2	C_3	C_1	C_4	C_2	C_{max}
M_1	3	6	4	8	3	9	13	21	
M_2	5	5	4	1	8	14	18	22	
M_3	4	4	2	4	12	18	20	26	26

Campbell, Dudek ve Smith (CDS) Algoritması

- J3-J1-J4-J2 sıralaması alt problemlerde en iyi Cmax değerini veren sıralama olduğu için bu sıralama en iyi sıralama olarak kabul edilir ve orijinal problem için çizelge oluşturulur.





Sonuç olarak minimum Cmax değeri J3-J1-J4-J2 sıralaması kullanılarak 26 zb olarak bulunmuştur.

Nawaz, Enscor ve Ham (NEH) Algoritması

- NEH algoritması iterasyonlar şeklinde ilerler. Algoritmanın çalışma prensibini açıklayan adımlar aşağıdaki gibidir:
- Adım-1: Her bir işin, toplam iş süresi (tüm makinelerde yapılması gereken sürelerin toplamı) bulunur: $T_j = \sum_{i=1}^{m} p_{ij}$
- Adım-2: İşler T_j değerlerine göre azalan şekilde bir listede sıralanır.
- Adım-3: Listedeki ilk iki iş seçilir ve yerleri değiştirilerek iki alternatif kısmi sıralama oluşturulur. Bu durumda oluşturulan kısmi sıralamaların çizelgelenmesi sonucu oluşacak Cmax değerleri bulunur. Cmax değeri yüksek olan sıralama göz ardı edilir. Cmax değeri düşük olan sıralama '*mevcut sıralama*' olarak adlandırılır.
- Adım-4: Listeden bir sonraki iş seçilir ve mevcut sıralamadaki tüm yerlere koyularak deneme yapılır. Elde edilecek tüm sıralamaların Cmax değerleri hesaplanır.
- Adım-5: Cmax değeri yüksek olan sıralama göz ardı edilir. Cmax değeri düşük olan sıralama '*mevcut sıralama*' olarak adlandırılır.
- Adım-6: Listede sıralanmamış iş kaldıysa Adım-4'e dönülür, yoksa algoritma sonlandırılır.

Nawaz, Enscor ve Ham (NEH) Algoritması

Örnek: Aşağıdaki F3||Cmax poblemini NEH algoritması yardımıyla çözünüz:

	j_1	j_2	j_3	j_4
M_1	6	8	3	4
M_2	5	1	5	4
M_3	4	4	4	2

Çözüm: İşlerin T_j değerleri hesaplanır.

	j_1	j_2	j_3	j_4
M1	6	8	3	4
M2	5	1	5	4
M3	4	4	4	2
T_j	**15**	**13**	**12**	**10**

T_j değerlerine göre azalan liste: J1-J2-J3-J4

Nawaz, Enscor ve Ham (NEH) Algoritması

- Listenin ilk iki sırasındaki işler (J1 ve J2)'den oluşan kısmi sıralamalar: $S_{12**} = \{J_1, J_2, *, *\}$ veya $S_{21**} = \{J_2, J_1, *, *\}$ şeklinde olabilir.
- S_{12**} kısmi sıralaması için oluşan kısmi çizelgenin Cmax değeri **19** olur:

	j_1	j_2	C_1	C_2	C_{max}
M_1	6	8	6	14	
M_2	5	1	11	15	
M_3	4	4	15	19	19

- S_{21**} kısmi sıralaması için oluşan kısmi çizelgenin Cmax değeri ise **23** olur:

	j_2	j_1	C_1	C_2	C_{max}
M_1	8	6	8	14	
M_2	1	5	9	19	
M_3	4	4	13	23	23

Nawaz, Enscor ve Ham (NEH) Algoritması

- İki sıralama karşılaştırılırsa, S_{12**} sıralamasının diğerine göre üstün olduğu görülür. Dolayısıyla bundan sonraki araştırmalar S_{12**} sıralaması üzerinden devam edecek, S_{21**} sıralaması göz ardı edilecektir.

Schedule	C_{max}
S_{21**}: (j_2 , j_1 , * , *)	23
S_{12**}: (j_1 , j_2 , * , *)	19

Nawaz, Enscor ve Ham (NEH) Algoritması

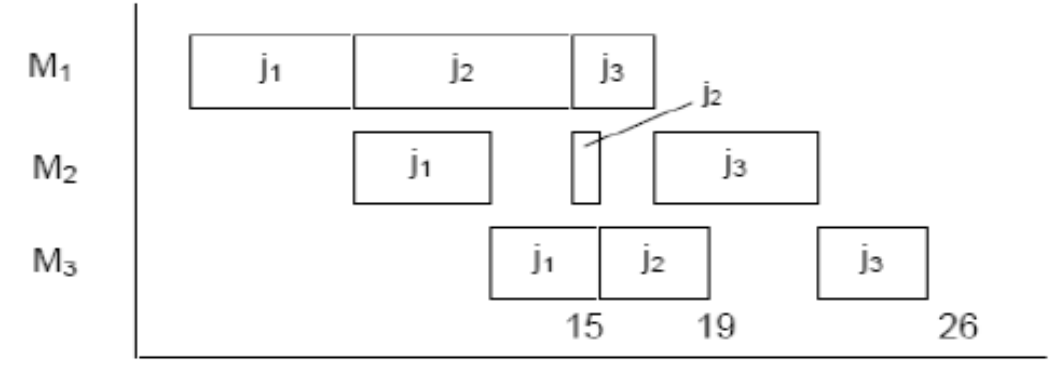
- Şimdi, listede J1 ve J2 işlerinden sonra gelen, J3 işi için denemeler yapılacaktır.
- J3 işi, mevcut sıralamada 3 farklı şekilde yerleştirilebilir:
  - *J1 işinden önce: Yeni kısmi sıralama* $S_{312*} = \{J_3, J_1, J_2, *\}$
  - *J1 işinden sonra: Yeni kısmi sıralama* $S_{132*} = \{J_1, J_3, J_2, *\}$
  - *J2 işinden sonra: Yeni kısmi sıralama* $S_{123*} = \{J_1, J_2, J_3, *\}$

Nawaz, Enscor ve Ham (NEH) Algoritması

$S_{123*} = \{J_1, J_2, J_3, *\}$ kısmi sıralaması için Cmax hesaplaması aşağıdaki gibidir:

	j_1	j_2	j_3	C_1	C_2	C_3	C_{max}
M_1	6	8	3	6	14	17	
M_2	5	1	5	11	15	22	
M_3	4	4	4	15	19	26	26

$S_{123*} = \{J_1, J_2, J_3, *\}$ kısmi sıralaması için Gantt şeması:



Nawaz, Enscor ve Ham (NEH) Algoritması

$S_{312*} = \{J_3, J_1, J_2, *\}$ kısmi sıralaması için Cmax hesaplaması aşağıdaki gibidir:

	j_3	j_1	j_2	C_3	C_1	C_2	C_{max}
M_1	3	6	8	3	9	17	
M_2	5	5	1	8	14	18	
M_3	4	4	4	12	18	22	22

Son olarak, $S_{132*} = \{J_1, J_3, J_2, *\}$ kısmi sıralaması için Cmax hesaplaması aşağıdaki gibidir:

	j_1	j_3	j_2	C_1	C_3	C_2	C_{max}
M_1	6	3	8	6	9	17	
M_2	5	5	1	11	16	18	
M_3	4	4	4	15	20	24	24

Nawaz, Enscor ve Ham (NEH) Algoritması

- Elde edilen kısmi çizelgeler kıyaslandığında, $S_{312*} = \{J_3, J_1, J_2, *\}$ kısmi çizelgesinin minimum Cmax oluşturduğu gözlenir.

Partial Schedule	C_{max}
S_{123*} $(j_1, j_2, j_3, *)$	26
S_{312*} $(j_3, j_1, j_2, *)$	22
S_{132*} $(j_1, j_3, j_2, *)$	24

- Dolayısıyla araştırmaya $S_{312*} = \{J_3, J_1, J_2, *\}$ kısmi çizelgesi üzerinden devam edilir.

Nawaz, Enscor ve Ham (NEH) Algoritması

- Şimdi, listede J1, J2 ve J3işlerinden sonra gelen, son iş olan J4 için denemeler yapılacaktır.
- J4 işi, mevcut sıralamada 4 farklı şekilde yerleştirilebilir:
  - *J3 işinden önce: Yeni kısmi sıralama* $S_{4312} = \{J_4, J_3, J_1, J_2\}$
  - *J3 işinden sonra: Yeni kısmi sıralama* $S_{3412} = \{J_3, J_4, J_1, J_2\}$
  - *J2 işinden önce: Yeni kısmi sıralama* $S_{3142} = \{J_3, J_1, J_4, J_2\}$
  - *J2 işinden sonra: Yeni kısmi sıralama* $S_{3124} = \{J_3, J_1, J_2, J_4\}$

$S_{4312} = \{J_4, J_3, J_1, J_2\}$ kısmi sıralaması için Cmax hesaplaması aşağıdaki gibidir.

	j_4	j_3	j_1	j_2	C_4	C_3	C_1	C_2	C_{max}
M_1	4	3	6	8	4	7	13	21	
M_2	4	5	5	1	8	13	18	22	
M_3	2	4	4	4	10	17	22	26	26

Nawaz, Enscor ve Ham (NEH) Algoritması

$S_{3412} = \{J_3, J_4, J_1, J_2\}$ kısmi sıralaması için Cmax hesaplaması aşağıdaki gibidir.

	j_3	j_4	j_1	j_2	C_3	C_4	C_1	C_2	C_{max}
M_1	3	4	6	8	3	7	13	21	
M_2	5	4	5	1	8	12	18	22	
M_3	4	2	4	4	12	14	22	26	26

$S_{3142} = \{J_3, J_1, J_4, J_2\}$ kısmi sıralaması için Cmax hesaplaması aşağıdaki gibidir.

	j_3	j_1	j_4	j_2	C_3	C_1	C_4	C_2	C_{max}
M_1	3	6	4	8	3	9	13	21	
M_2	5	5	4	1	8	14	18	22	
M_3	4	4	2	4	12	18	20	26	26

Nawaz, Enscor ve Ham (NEH) Algoritması

Son olarak, $S_{3124} = \{J_3, J_1, J_2, J_4\}$ kısmi sıralaması için Cmax hesaplaması aşağıdaki gibidir.

	j_3	j_1	j_2	j_4	C_3	C_1	C_2	C_4	C_{max}
M_1	3	6	8	4	3	9	17	21	
M_2	5	5	1	4	8	14	18	25	
M_3	4	4	4	2	12	18	19	27	27

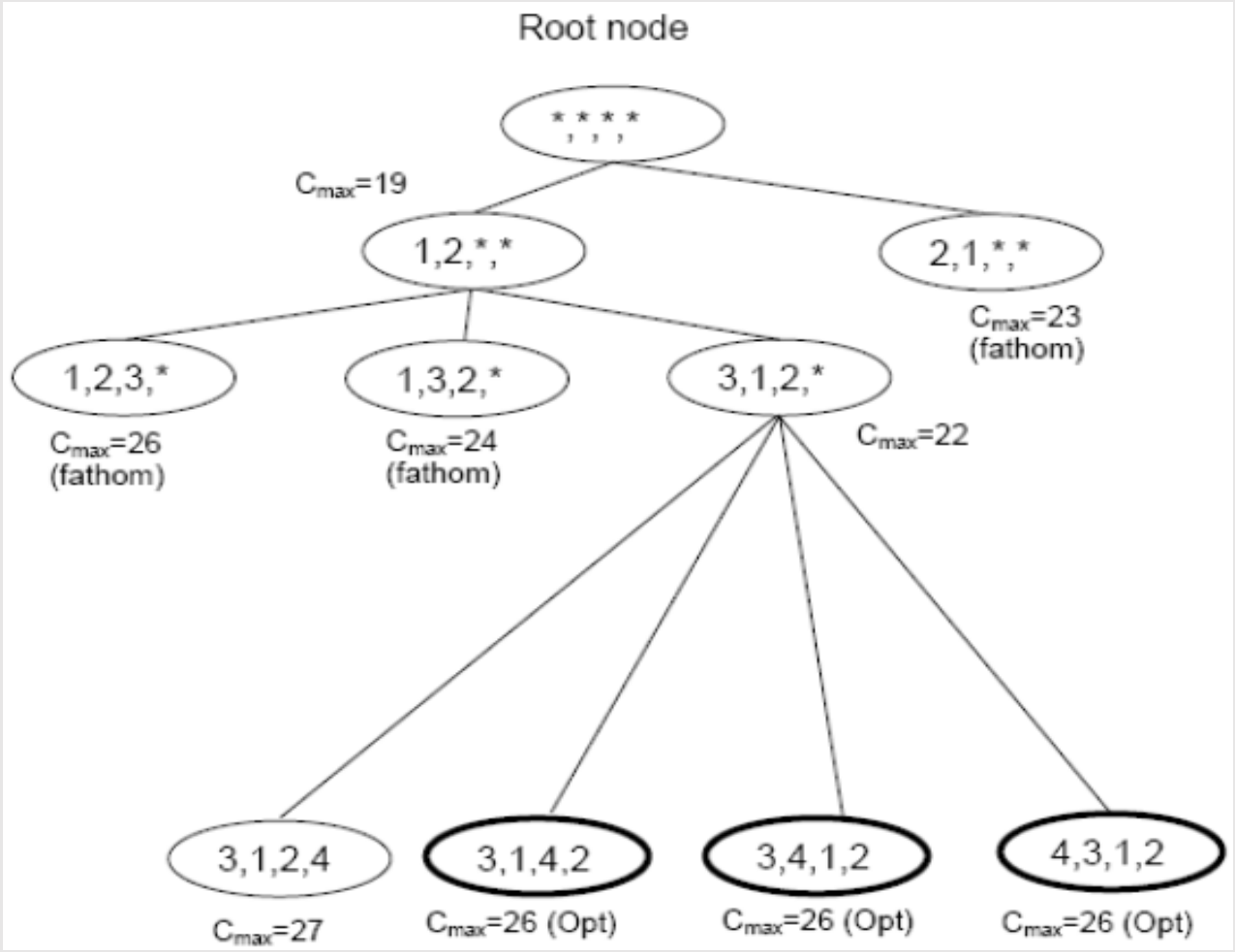
Elde edilen kısmi çizelgeler kıyaslandığında, üç alternatif sıralama için aynı Cmax değerinin (26 zb) elde edildiği görülür.

Schedule	C_{max}
S_{3124} : (j_3, j_1, j_2, j_4)	27
S_{3124} : (j_4, j_3, j_1, j_2)	26
S_{3412} : (j_3, j_4, j_1, j_2)	26
S_{3142} : (j_3, j_1, j_4, j_2)	26

Bu durumda üç alternatif çözüm vardır:
$\{J_4, J_3, J_1, J_2\}$; $\{J_3, J_4, J_1, J_2\}$ veya $\{J_3, J_1, J_4, J_2\}$

Nawaz, Enscor ve Ham (NEH) Algoritması

- NEH algoritması için elde edilen çözümün aşamalarını gösteren arama ağacı aşağıdaki gibi gösterilebilir.



Bölüm Çalışma Soruları

Soru 1

F3||Cmax probleminin optimum sonucunu bulunuz ve optimum olduğunu matematiksel olarak gösteriniz.

Job	1	2	3	4
M_1	3	2	1	8
M_2	5	1	8	7
M_3	7	4	2	2

Soru 2

Aşağıdaki 3 makine 4 iş problemini CDS ve NEH algoritmaları ile çözünüz.

Job	1	2	3	4
M_1	4	7	3	1
M_2	9	6	3	8
M_3	7	4	8	5

Soru 3

Aşağıdaki F3||Cmax problemini; (i) Johnson algoritmasını ve (ii) CDS algoritmasını kullanarak ayrı ayrı çözünüz ve Cmax'ı bulunuz.

Job	1	2	3	4
M_1	6	8	3	4
M_2	4	1	2	3
M_3	5	6	4	7

-Eğer teslim tarihleri aşağıdaki gibi olsaydı yukarıdaki yöntemlerden hangisi en büyük gecikmeyi (Lmax) minimize eder?

Job	1	2	3	4
Due Date	17	13	11	18

Soru 4

Aşağıdaki F3||Cmax problemini CDS ve NEH algoritmaları ile çözünüz.

	Job					
Machines	1	2	3	4	5	6
M_1	2	23	25	5	15	10
M_2	29	3	20	7	11	2
M_3	19	8	11	14	7	4

Soru 5

Aşağıdaki F4||Cmax problemi için en iyi sıralamayı CDS algoritması ile bulunuz.

Job	1	2	3	4	5
1	1	10	17	12	11
2	13	12	9	17	3
3	6	18	13	2	5
4	2	18	4	6	16

Soru 6

Aşağıdaki F3||Cmax problemi için en iyi sıralamayı bulunuz.

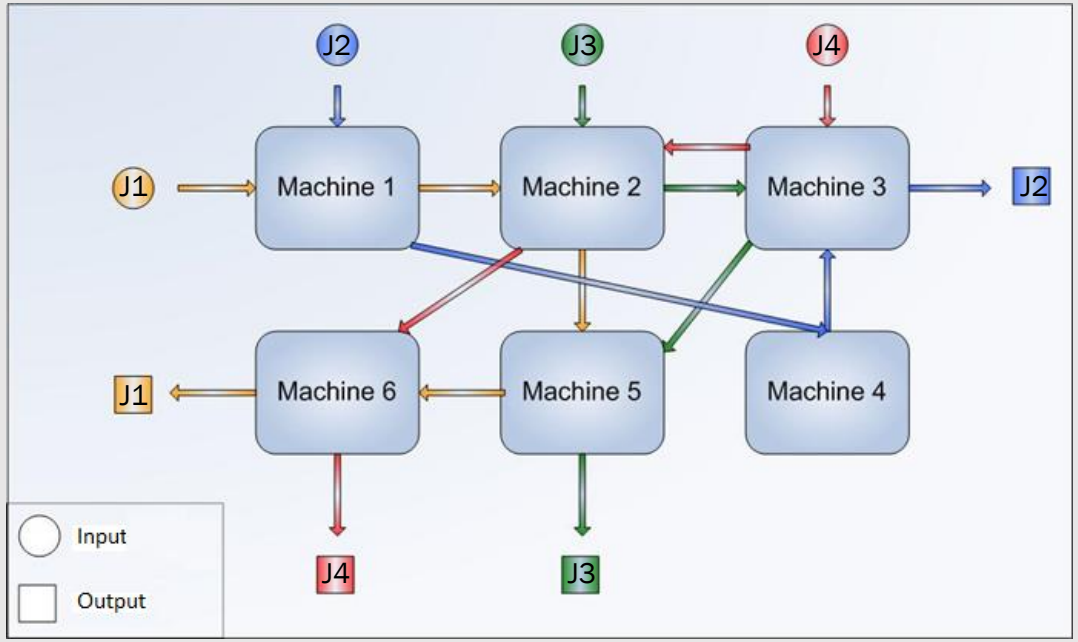
Job	1	2	3	4
M1	6	8	3	4
M2	3	1	3	3
M3	4	4	4	2

Atölye Tipi Çizelgeleme

-Shifting Bottleneck Sezgiseli

Atölye Tipi Çizelgeleme

- Atölye tipi çizelgeleme problemi, n iş - m makineden oluşan bir sistemde, **birbirlerinden farklı rotalar izleyebilen** işlerin çizelgelenmesi problemidir.
- Örneğin yanda verilen 4 iş – 6 makineden oluşan bir sistemde, J1 işi sırasıyla M1-M2-M5-M6 numaralı makinelerde işlem görürken; J3 işi M2-M3-M5 numaralı makinelerde işlem görmektedir. Yine diğer işler de (J2 ve J4) farklı rotalar izlemektedir.
- Bu örnekte işler aynı sayıda operasyondan geçmektedir (her biri toplamda üç makinede işlem görüyor). Dikkat edilmesi gereken husus, her bir iş için işlem görmesi gereken operasyonların sayıları da farklı olabilir.
- İşlerin izleyeceği rotalar ve her bir makine için işlem süreleri önceden bilinmektedir.



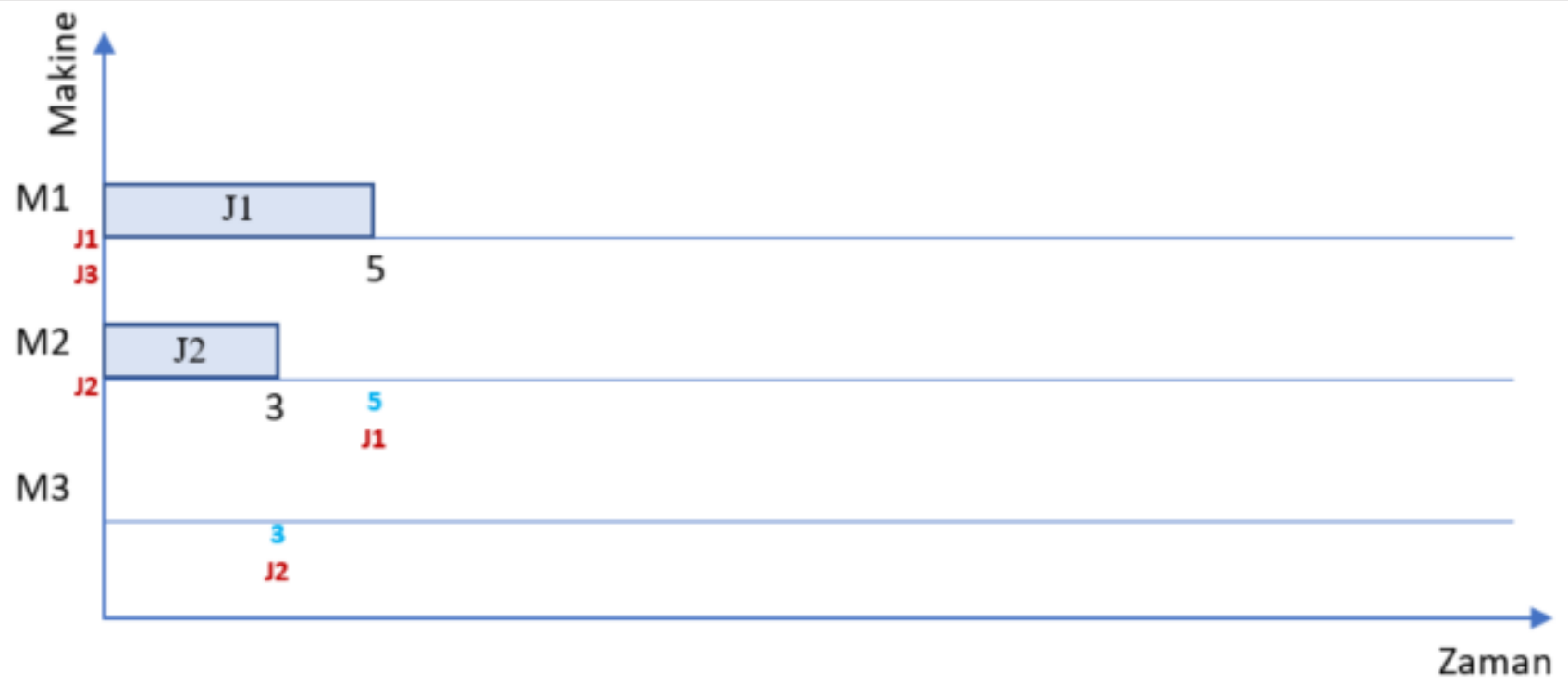
Örnek 1

- Atölye tipi çizelgeleme problemlerinin çözümü için ders kapsamında yararlandığımız öncelik kurallarından (örneğin SPT, LPT, SRPT gibi) yararlanılabilir.

- Yanda verilen örneği inceleyelim ve **Earliest Starting Time - EST** (En Erken Başlama) kuralını işleterek çözelim. Eşitlik durumunda (aynı anda birden fazla iş başlayabilecek durumda olursa eşitliği bozmak için SPT kuralı işletilebilir.

İş	Rota ve İşlem Zamanları		
J1	M1(5)	M2(2)	M3(4)
J2	M2(3)	M3(7)	M1(3)
J3	M1(8)	M3(6)	M2(5)

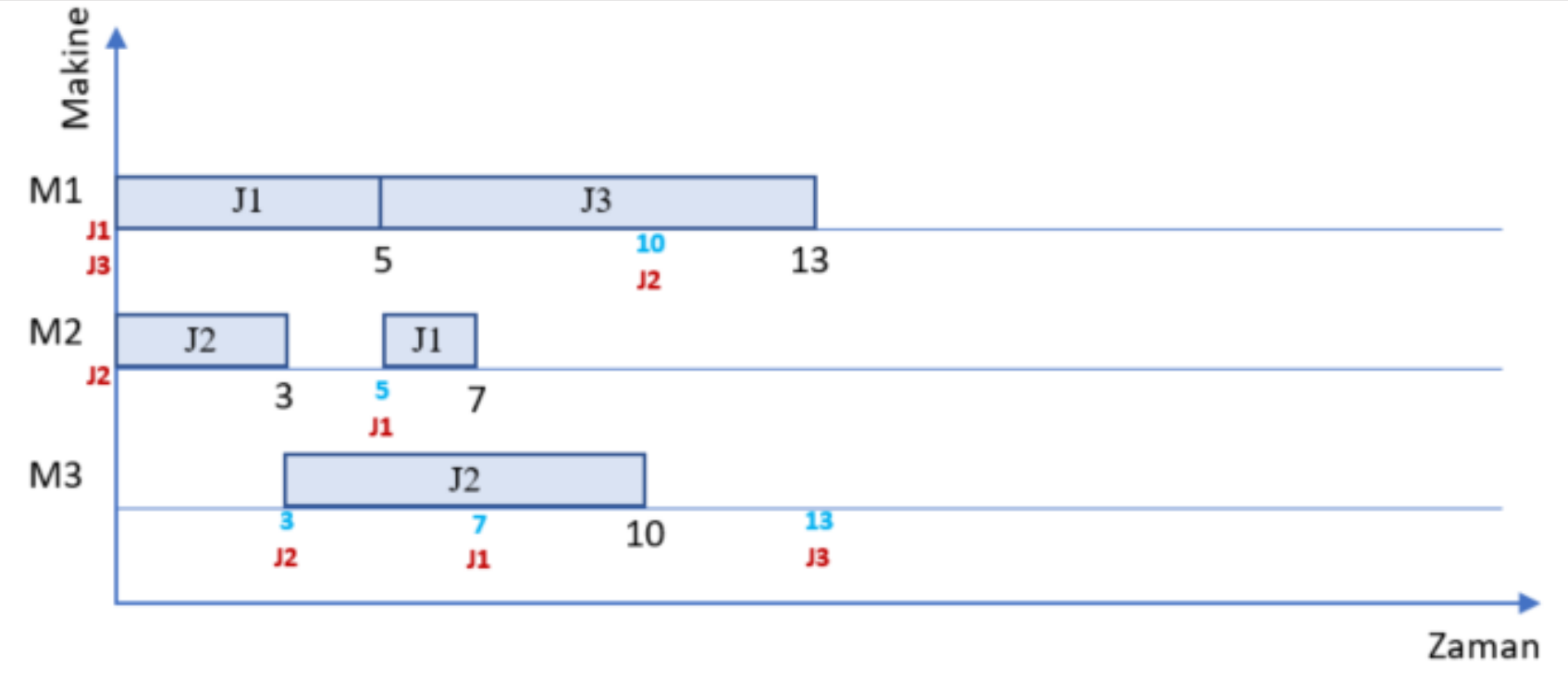
Örnek 1



- t=0 iken J1 ve J3 numaralı işler M1 de işlem görmeye hazırdır. Her iki iş de aynı anda aynı makinede başlayabilecek durumda olmasından dolayı eşitliği bozmak için aralarından daha kısa süreye sahip olan J1 seçilir ve M1'e atanır.

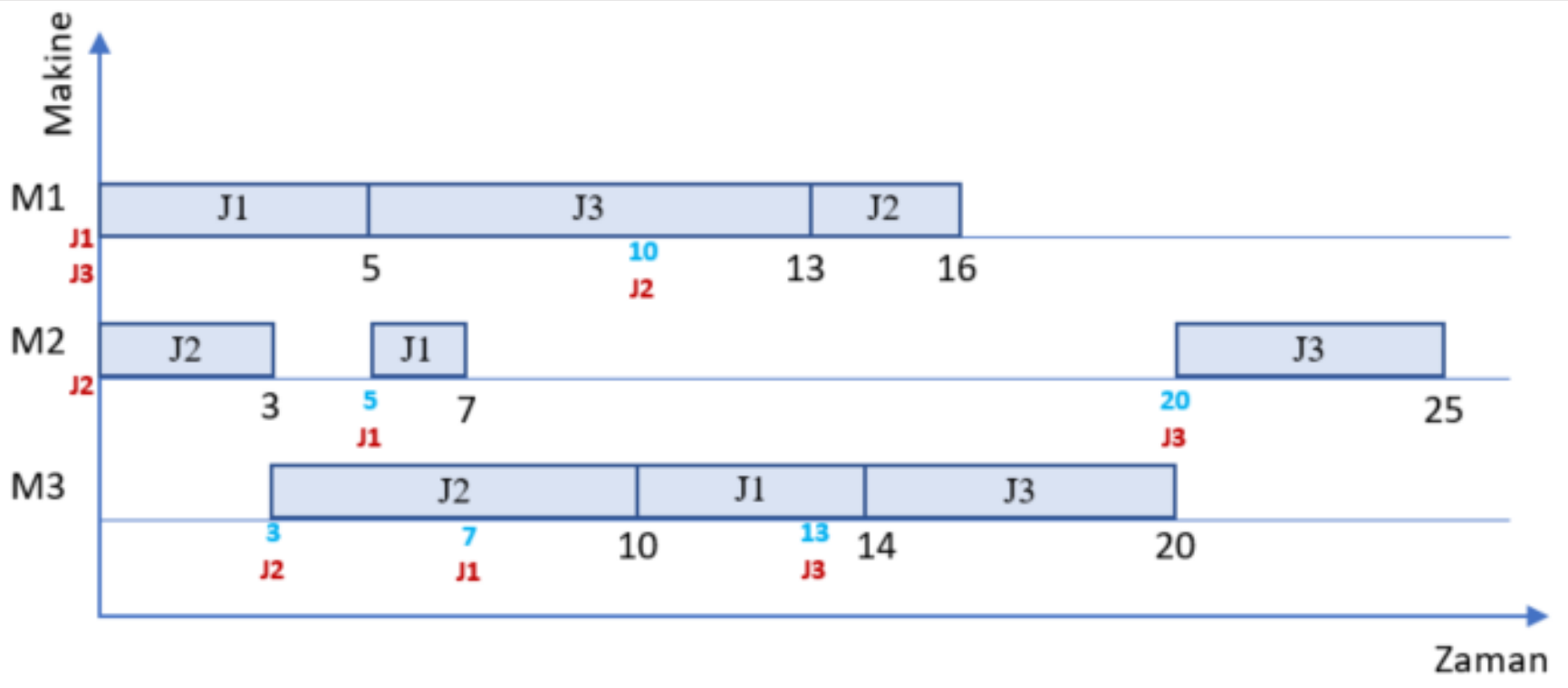
- t=0 iken J2 numaralı iş M2'de işlenebilir durumdadır. Başka iş olmadığı için j2 başlatılır.

Örnek 1



- t=3 iken J2 numaralı işin M2'de işlemi tamamlanır ve M3'te hazır pozisyonda sadece kendisi olduğu için M3'te başlatılır.
- t=5 iken J1 numaralı işin M1'de işlemi tamamlanır ve M2'de işlenmeye başlar (başka bekleyen yok).
- t=7 iken J1 numaralı işin M2'de işlemi tamamlanır ve M3'te beklemeye başlar. M3'te halen J2 numaralı iş devam etmektedir.

Örnek 1



- t=10 iken M3'te bitirilen J2 işi M1'de beklemeye başlar ve M3'te halihazırda beklemekte olan J1 işi başlatılır.
- t=13 iken M1'de tamamlanan J3 işi M3'te beklemeye başlar ve M1'de beklemekte olan J2 işi başlatılır.
- t=14 iken M3'te J1 işi bitirilir ve beklemekte olan J3 işi başlatılır.
- t=20 iken J3 işi M3'te bitirilir ve M2'de başlatılır.
- t=25 iken tüm işler tamamlanmıştır.

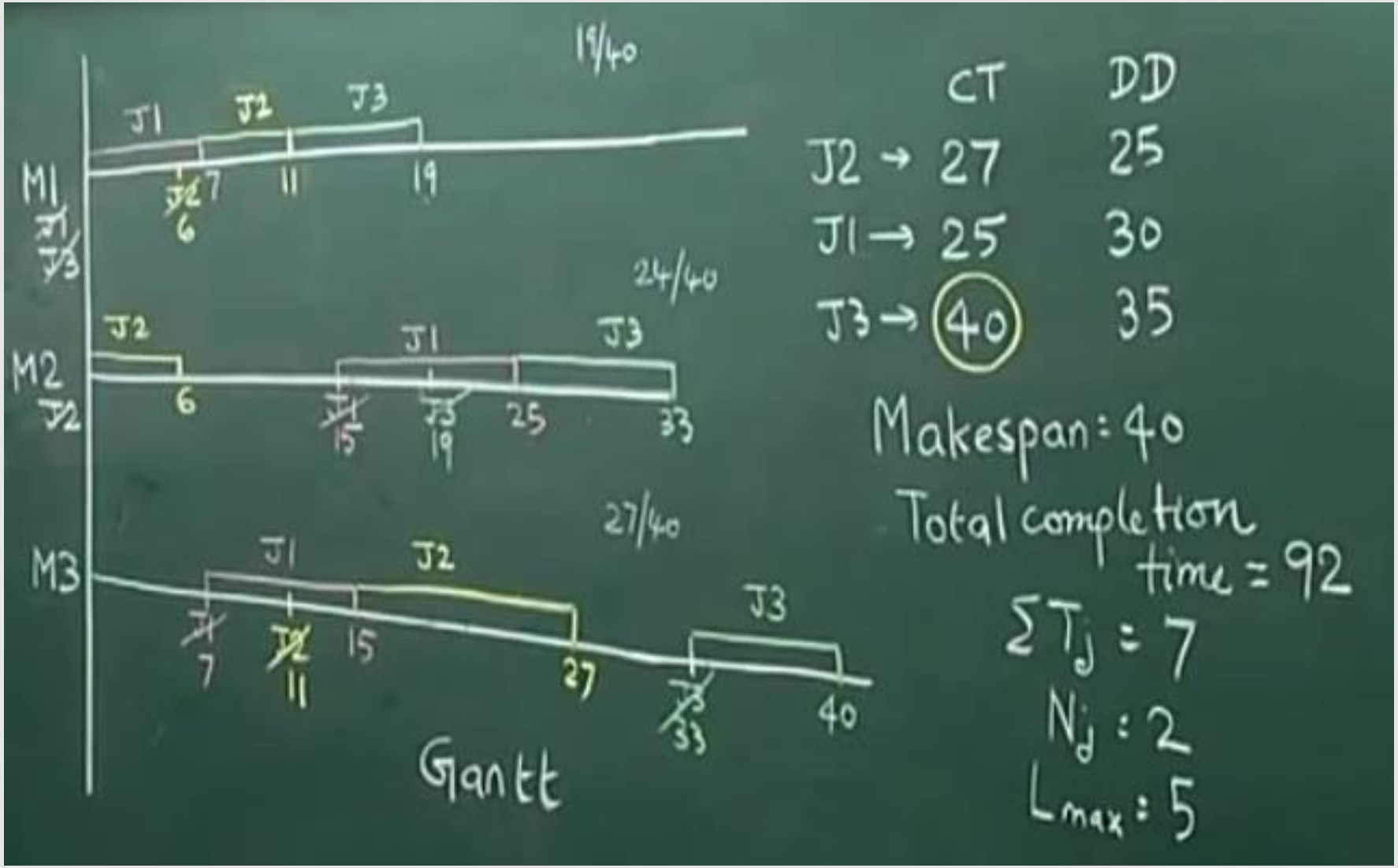
Örnek 2

İş	Rota ve İşlem Zamanları		
J1	M1(7)	M3(8)	M2(10)
J2	M2(6)	M1(4)	M3(12)
J3	M1(8)	M2(8)	M3(7)

- Rotaları ve işlem zamanları yanda verilen işleri **SPT** kuralından yararlanarak çözelim.
- Makinelerde işlem görmek için bekleyen işler dururken, makineleri boş bırakmamaya özen gösterilmelidir.

https://www.youtube.com/watch?v=xZs7WsNPJXY

Örnek 2


19/40
J1
J2
J3
M1
7
11
19
6
24/40
J2
J1
J3
M2
6
15
19
25
33
27/40
J1
J2
J3
M3
7
11
15
27
33
40
Gantt
CT
DD
J2 → 27
25
J1 → 25
30
J3 → 40
35
Makespan: 40
Total completion time = 92
ΣTj = 7
Nj : 2
Lmax : 5

Sıralama İçin Kullanılabilecek Kurallar

İşlem Zamanına Bağlı

- **SPT**
- LPT
- STPT (Total)
- LTPT
- SRPT (including current)
- LRPT

İşlem Sayısına Bağlı

- Min NoP
- Max NoP
- Min RemOp
- Max RemOp

Diğer:

- **FCFS**
- LCFS
- Random

Teslim Tarihine Bağlı

- **EDD**
- MinSlack (Slack=DD-CurrentTime-RPT)
- Min Slack/RnoP

Ağırlıklı Birleşim:

- $\alpha SRPT + \beta MinRemOp$
- $\alpha SRPT + \beta MinTPT$

Shifting Bottleneck (SB) Sezgiseli

- Atölye tipi çizelgeleme problemleri için önerilen farklı yaklaşımlar bulunmakla birlikte, Shifting Bottleneck (SB) sezgiseli son zamanlarda geliştirilmiş ve iyi çözüm üreten algoritmalardan birisidir.
- SB sezgiseli, 1988 yılında **Adams, Balas ve Zawack** tarafından önerilmiş ve 1993 yılında Dauzere-Peres ve Lassere tarafından modifiye edilmiştir. Son olarak 1995'te SB sezgiseli Balas, Lenstra ve Vazacopoulos tarafından genişletilmiştir.
- SB algoritması atölye tipi çizelgeleme problemlerinde darboğaz oluşturan makinelerin iyi yönetimini simüle eden tek algoritma olarak bilinmektedir.

Shifting Bottleneck (SB) Sezgiseli

- Atölye tipi çizelgeleme problemleri için önerilen tüm sezgisellerden daha üstün olduğu bilinmektedir.
- SB tarafından bulunan çözümler, diğer geliştirilen algoritmaların performanslarının ölçülmesi için referans olarak kullanılmaktadır.
- SB sezgiselinin **Cmax** ve **Lmax** kriterlerini minimize etmek açısından iyi sonuçlar verdiği, farklı sayıda iş ve makineden oluşan test problemleri ile yapılan deneysel çalışmalarda gösterilmiştir.
- Örneğin, Adams, Balas ve Zawack tarafından SB sezgiselinin performansı, farklı öncelik kuralları (SPT, LPT, Random, FCFS, vb.) ile kıyaslanmıştır. Bu amaçla çözülen 40 test probleminde SB sezgiseli diğer sezgisellere oranla üstünlük sağlamıştır.
- Ayrıca, aynı ekip tarafından daha büyük boyutlu test problemleri de çözülmüştür (örneğin 500 iş - 10 makine). Yapılan karşılaştırmalar sonucunda, SB sezgiselinin 30 iş - 10 makineden büyük problemler için optimum sonuçları verdiği gözlenmiştir.

Shifting Bottleneck (SB) Sezgiseli

- SB sezgiseli iterasyonlar halinde ilerleyen bir algoritmadır. Her iterasyonda tek makine çizelgeleme problemi gibi düşünülen alt problemler oluşturulur ve işlerin rj zamanları varmış gibi Lmax'ı minimize edecek sıralamalar oluşturulur. Algoritmanın işleyiş prensibi aşağıdaki gibidir:
- **Adım 1:**

  $M_0 = \emptyset$ olarak ayarlanır ve CPM ağı çizilerek Cmax bulunur.
- **Adım 2:**

  Her bir $m \in M_0$ makinesi için $1|rj|Lmax$ problemi çözülür ve m makinesinde en iyi sıralama bulunur. İşlerin rj ve dj değerleri, çizilen CPM ağı üzerinden aşağıdaki şekilde hesaplanır:

  rj: m makinesi üzerindeki "m, j" düğümü için en erken başlama zamanı

  dj: m makinesi üzerindeki "m, j" düğümü için en geç tamamlanma zamanı
- **Adım 3:**

  Darboğaz oluşturan makine tespit edilir. Bunun için her bir $m \in M_0$ makinesi için elde edilen çözümlerin Lmax değerleri karşılaştırılır. En büyük Lmax değerinin oluştuğu makine darboğaz oluşturmaktadır.

  M_0 kümesi, yeni bulunan darboğaz makinesi eklenerek güncellenir.

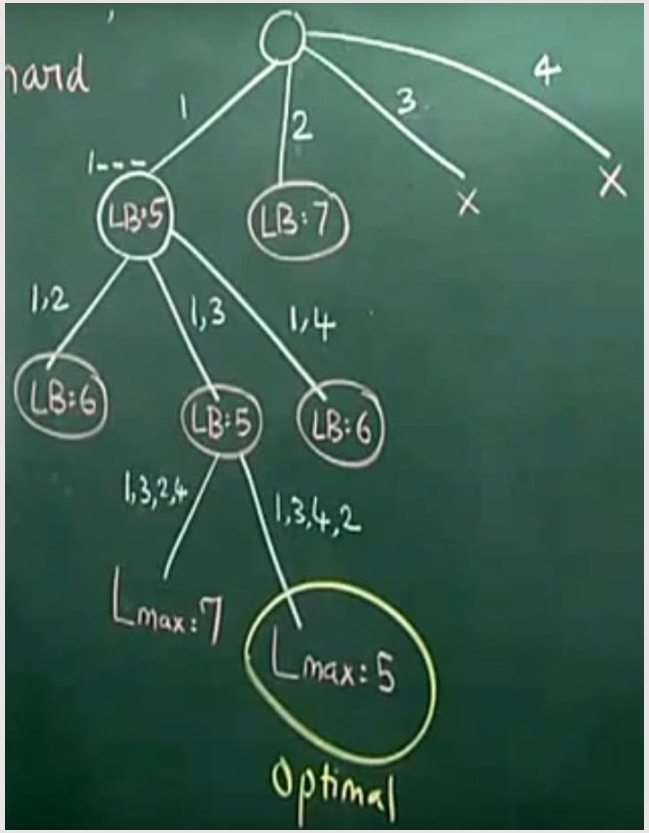
  CPM ağı, üzerine darboğaz makinesinde elde edilen sıralamanın da eklenmesiyle güncellenir. Cmax değeri de böylece güncellenir.

  Eğer tüm makineler M_0 içindeyse dur, değilse Adım 2'ye dön.

Shifting Bottleneck (SB) Sezgiseli

- SB sezgiselinde Adım 2'de her bir makine için oluşturulan 1|rj|Lmax probleminin çözümünde EDD kuralından yararlanılabilir. Fakat işlerin rj zamanları 0 olmadığı için, bu kural kullanılırken çok dikkatli olunmalıdır. Çünkü işlerin hazır olma zamanları da dikkate alınmalıdır. Bunun için tüm ihtimaller göz önünde bulundurularak bir çözüm yapılmalıdır.
- Tüm ihtimallerin göz önünde bulundurulmasıyla yapılacak çözüm, aşağıdaki 1|rj|Lmax örnek problemi çözümünden de görüleceği üzere aslında bir dal-sınır prosedürü uygulamasıdır.

İş	1	2	3	4
pj	4	2	6	5
rj	0	1	3	5
dj	8	12	11	10



	1	3	4	2
Cj	4	10	15	17
dj	8	11	10	12
L	-4	-1	5	5

Shifting Bottleneck (SB) Sezgiseli - Örnek

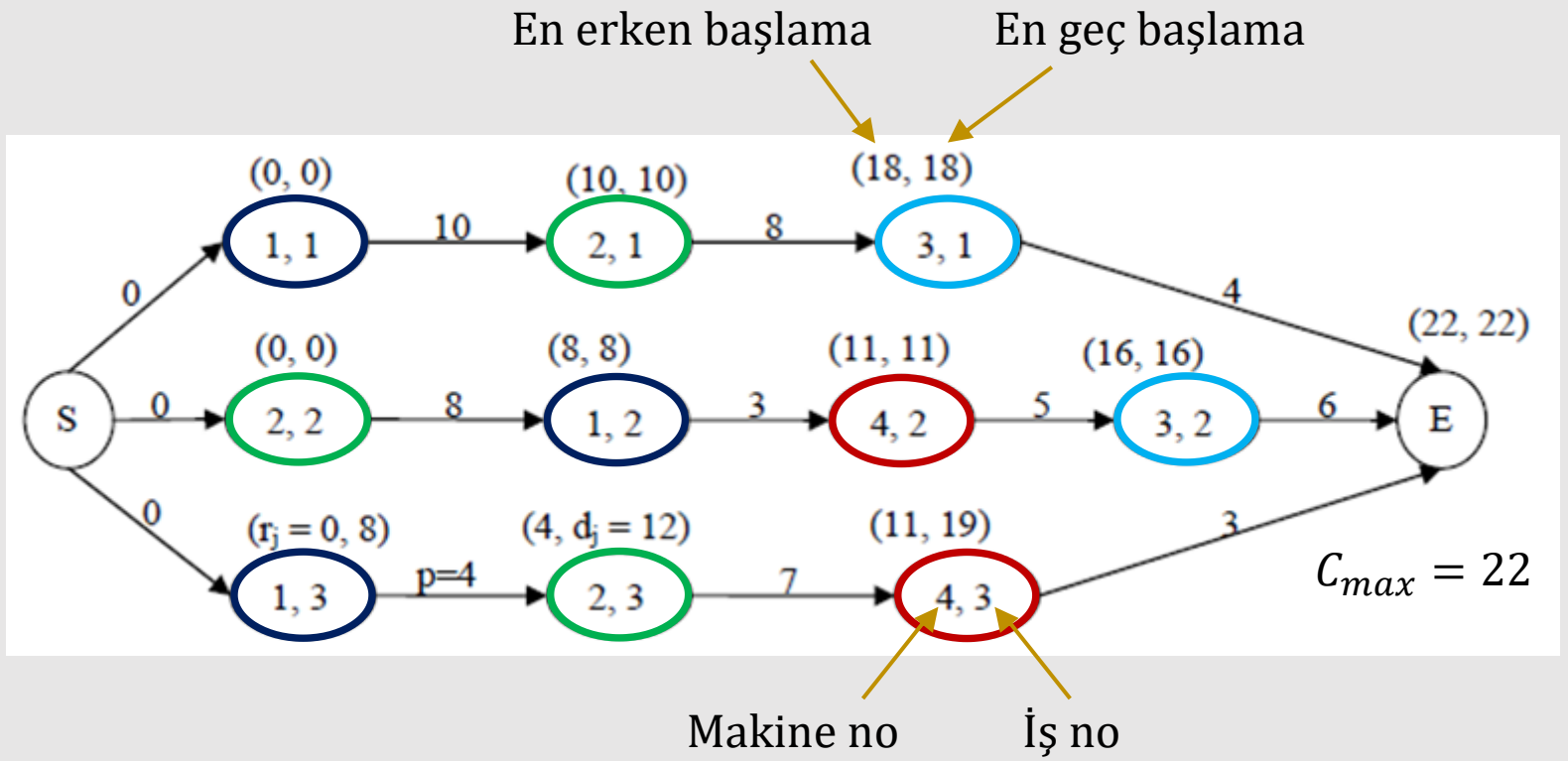
- Aşağıda verilen J4||Cmax problemini SB sezgiseli ile çözelim.

İş	Rota ve İşlem Zamanları			
J1	M1(10)	M2(8)	M3(4)	
J2	M2(8)	M1(3)	M4(5)	M3(6)
J3	M1(4)	M2(7)	M4(3)	

Çözüme Geçmeden Önce Ekstra Soru
Bu örnekte, Cmax'ın alabileceği minimum değer nedir ve nasıl hesaplanır?

Shifting Bottleneck (SB) Sezgiseli - Örnek

- İterasyon 1: $M_0 = \emptyset$ olarak ayarlanır ve CPM ağı çizilir.



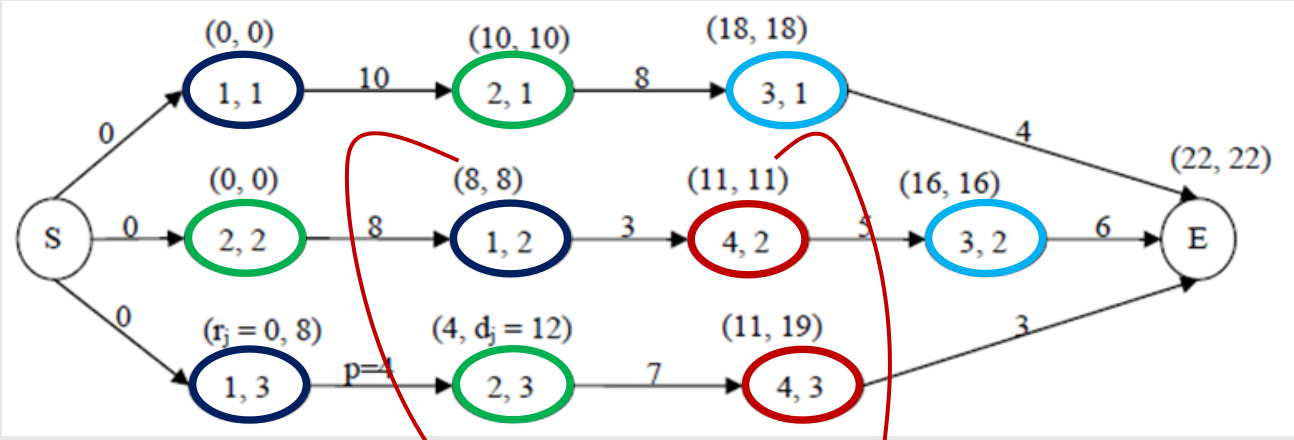
Bir işin en erken ve en geç başlama zamanları hesaplanırken;
En erken başlamada öncüllerden en büyük en erken başlama zamanı,
en geç başlamada ise ardıllardan en küçük en geç başlama zamanı dikkate alınır.

Shifting Bottleneck (SB) Sezgiseli - Örnek

- 1|rj|Lmax problemi her bir makine (M1, M2, M3, M4) için aşağıdaki şekilde çözülür.

Makine 1:

1|rj|Lmax problemi için EDD kuralı uygulanması sonucu minimum Lmax değerini verecek olan sıralama J1-J2-J3 şeklinde elde edilir.



Burada salt olarak EDD sıralamasına değil, aynı zamanda işlerin rj değerlerine de bakılması gerekir. Örneğin bir işin rj değeri 0 değilse ve rj değeri 0 olan başka işler varsa, bu iş dj değeri en küçük olsa bile ilk sıraya koyulmaz (çünkü bu durumda makine boşta bekleyecek ve dolayısıyla Lmax değeri artacaktır).

Job (j)	1	2	3
Operation	(1,1)	(1,2)	(1,3)
p_{1j}	10	3	4
r_{1j}	0	8	0
d_{1j}	10	11	12

Shifting Bottleneck (SB) Sezgiseli - Örnek

- M1'de Lmax'ı minimize eden çizelge aşağıdaki gibidir:

Job (j)	P_{1j}	r_{1j}	S_{1j}	C_{1j}	d_{1j}	L_{1j}
1	10	0	0	10	10	0
2	3	8	10	13	11	2
3	4	0	13	17	12	5

- Böylece M1'deki maksimum gecikme 5'tir, yani Lmax(1)=5.

- Sonrasında, sıradaki makine olan M2'ye geçilir ve tek makine çizelgeleme problemi gibi çözülür.

Shifting Bottleneck (SB) Sezgiseli - Örnek

- **Makine 2:**
- M2 üzerinde 1|rj|Lmax problemi için EDD kuralı uygulanması sonucu minimum Lmax değerini verecek olan sıralama J2-J3-J1 şeklinde elde edilir.

Job (j)	1	2	3
Operation	(2,1)	(2,2)	(2,3)
p_{2j}	8	8	7
r_{2j}	10	0	4
d_{2j}	18	8	19

NOT: Burada dikkat edilmesi gereken husus daha önce de vurgulandığı üzere EDD uygulanırken işlerin rj değerlerine dikkat edilmesi gerektiğidir.
J2-J1-J3 sıralamasının seçilmemesinin sebebi J1'in rj değeridir. Çünkü J2 işi bittiği sırada J1 hazır değildir. Şayet J2-J1-J3 sıralaması seçilmiş olsaydı Lmax değeri 5 değil 6 olacaktı.

M2 üzerinde elde edilen sıralamaya göre oluşacak olan çizelgeleme aşağıdaki tablodaki gibi olur:

Job (j)	P_{2j}	r_{2j}	S_{2j}	C_{2j}	d_{2j}	L_{2j}
2	8	0	0	8	8	0
3	7	4	8	15	19	-4
1	8	10	15	23	18	5

Shifting Bottleneck (SB) Sezgiseli - Örnek

- **Makine 3:**

- M3 üzerinde 1|rj|Lmax problemi için EDD kuralı uygulanması sonucu minimum Lmax değerini verecek olan sıralama J2-J1 şeklinde olacaktır.

Job (j)	1	2	3
Operation	(3,1)	(3,2)	(3,3)
p_{3j}	4	6	-
r_{3j}	18	16	-
d_{3j}	22	22	-

- Elde edilen sıralamanın M3 için uygulanması sonucu maksimum gecikme 4 olacaktır, yani Lmax(3)=4.

Job (j)	p_{3j}	r_{3j}	S_{3j}	C_{3j}	d_{3j}	L_{3j}
2	6	16	16	22	22	0
1	4	18	22	26	22	4

Shifting Bottleneck (SB) Sezgiseli - Örnek

- **Makine 4:**
- 1|rj|Lmax probleminin M4 için EDD kuralı ile çözülmesi sonucu minimum Lmax değerini verecek olan sıralama J2-J3 şeklinde olacaktır.

job (j)	1	2	3
Operation	(4,1)	(4,2)	(4,3)
p_{4j}	-	5	3
r_{4j}	-	11	11
d_{4j}	-	16	22

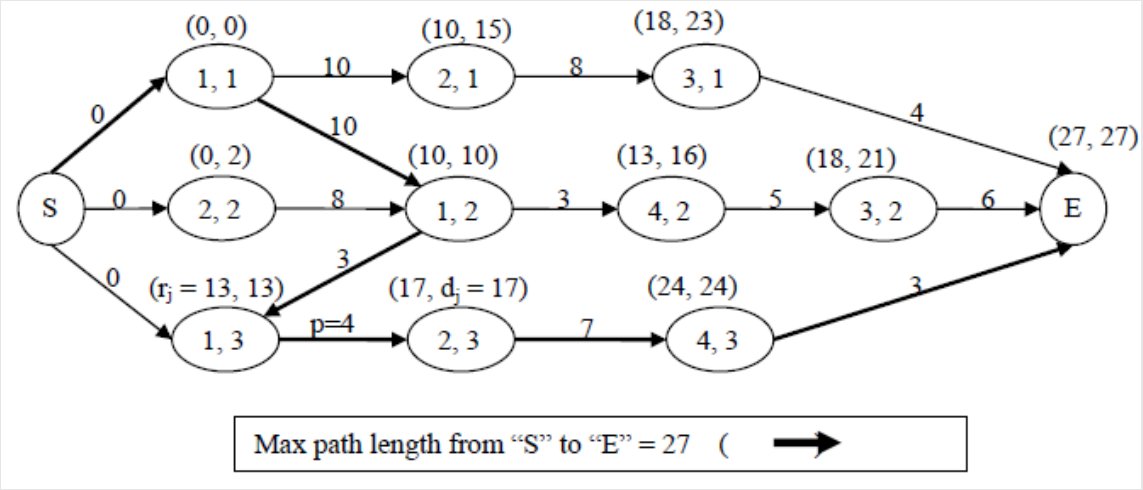
- Elde edilen sıralamanın M4 için uygulanması sonucu maksimum gecikme 0 olacaktır, yani Lmax(4)=0.

Job (j)	p_{4j}	r_{4j}	S_{4j}	C_{4j}	d_{4j}	L_{4j}
2	5	11	11	16	16	0
3	3	11	16	19	22	-3

Shifting Bottleneck (SB) Sezgiseli - Örnek

- Başka makine kalmadığı için darboğaz oluşturan makine tespit edilebilir.
- Darboğaz oluşturan makine, maksimum Lmax değerine sahip olan makinedir.
- Örneğimizde maksimum Lmax değeri Lmax(1)=Lmax(2)=5 ile M1 ve M2 oluşturuyor.
- Eşitliği bozmak için rastgele M1 makinesi darboğaz oluşturan makine olarak seçilir ve iterasyon 1 burada sonlandırılır (M1 makinesi üzerinde sıralama J1-J2-J3 şeklindeydi).

- **İterasyon 2: $M_0 = \{M1\}$ olarak ayarlanır ve CPM ağı güncellenir.**



Cmax'=Cmax+Lmax(1)=22+5=27 oldu.

Şimdi yeniden, güncellenmiş CPM ağı kullanılarak M2, M3 ve M4 makineleri için Lmax değerleri hesaplanır.

Shifting Bottleneck (SB) Sezgiseli - Örnek

- **Makine 2:**
- Aşağıda verilen güncellenmiş tek makine (1|rj|Lmax) problem verisi için minimum Lmax değerini verecek sıralama J2-J1-J3 olarak elde edilir.

Job (j)	1	2	3
Operation	(2,1)	(2,2)	(2,3)
p_{2j}	8	8	7
r_{2j}	10	0	17
d_{2j}	23	10	24

- Elde edilen sıralamaya göre yapılacak çizelgeleme sonrası Lmax değerinin 1 olduğu bulunur, yani Lmax(2)=1.

Job (j)	p_{2j}	r_{2j}	S_{2j}	C_{2j}	d_{2j}	L_{2j}
2	8	0	0	8	10	-2
1	8	10	10	18	23	-5
3	7	17	18	25	24	1

Shifting Bottleneck (SB) Sezgiseli - Örnek

- **Makine 3:**

- Aşağıda verilen güncellenmiş tek makine (1|rj|Lmax) problem verisi için minimum Lmax değerini verecek sıralama J1-J2 olarak elde edilir.

Job (j)	1	2	3
Operation	(3,1)	(3,2)	(3,3)
p_{3j}	4	6	-
r_{3j}	18	18	-
d_{3j}	27	27	-

- Elde edilen sıralamaya göre yapılacak çizelgeleme sonrası Lmax değerinin 1 olduğu bulunur, yani Lmax(3)=1.

Job (j)	p_{3j}	r_{3j}	S_{3j}	C_{3j}	d_{3j}	L_{3j}
1	4	18	18	22	27	-5
2	6	18	22	28	27	1

Shifting Bottleneck (SB) Sezgiseli - Örnek

- **Makine 4:**
- Aşağıda verilen güncellenmiş tek makine (1|rj|Lmax) problem verisi için minimum Lmax değerini verecek sıralama J2-J3 olarak elde edilir.

Job (j)	1	2	3
Operation	(4,1)	(4,2)	(4,3)
p_{4j}	-	5	3
r_{4j}	-	13	24
d_{4j}	-	21	27

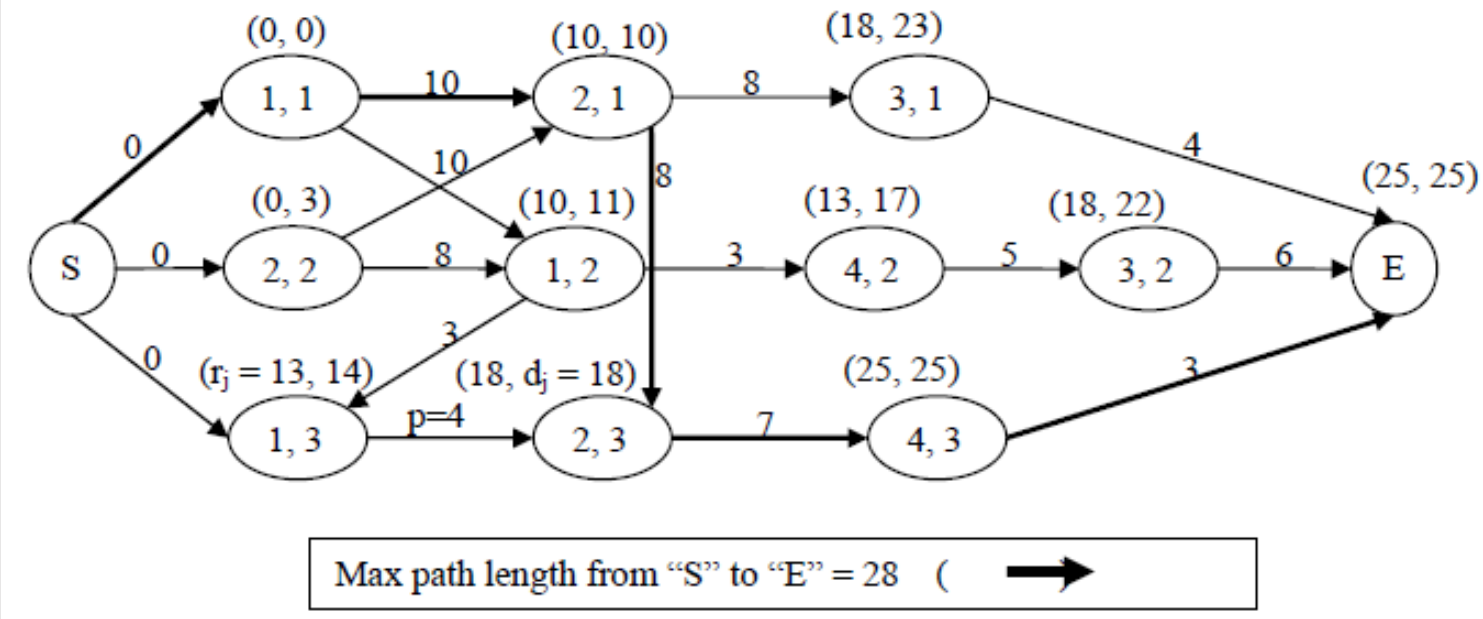
- Elde edilen sıralamaya göre yapılacak çizelgeleme sonrası Lmax değerinin 0 olduğu bulunur, yani Lmax(4)=0.

Job (j)	p_{4j}	r_{4j}	S_{4j}	C_{4j}	d_{4j}	L_{4j}
2	5	13	13	18	21	-3
3	3	24	24	27	27	0

- Başka makine kalmadığı için tekrar darboğaz oluşturan makine bulunur. Burada M2 ve M3 makineleri maksimum Lmax=1 değerine sahiptir. Rastgele M2 seçilir ve iterasyon 2 burada sonlanır (M2 için sıralama J2-J1-J3 olarak bulunmuştu).

Shifting Bottleneck (SB) Sezgiseli - Örnek

- İterasyon 3: $M_0 = \{M1, M2\}$ olarak ayarlanır ve CPM ağı tekrar güncellenir.



Cmax''=Cmax'+Lmax(2)=27+1=28 oldu.

Şimdi yeniden, güncellenmiş CPM ağı kullanılarak M3 ve M4 makineleri için Lmax değerleri hesaplanır.

Shifting Bottleneck (SB) Sezgiseli - Örnek

- **Makine 3:**

- Aşağıda verilen güncellenmiş tek makine (1|rj|Lmax) problem verisi için minimum Lmax değerini verecek sıralama J1-J2 olarak elde edilir.

Job (j)	1	2	3
operation	(3,1)	(3,2)	(3,3)
p_{3j}	4	6	-
r_{3j}	18	18	-
d_{3j}	28	28	-

- Elde edilen sıralamaya göre yapılacak çizelgeleme sonrası Lmax değerinin 0 olduğu bulunur, yani Lmax(3)=0.

Job (j)	p_{3j}	r_{3j}	S_{3j}	C_{3j}	d_{3j}	L_{3j}
1	4	18	18	22	28	-6
2	6	18	22	28	28	0

Shifting Bottleneck (SB) Sezgiseli - Örnek

- **Makine 4:**
- Aşağıda verilen güncellenmiş tek makine (1|rj|Lmax) problem verisi için minimum Lmax değerini verecek sıralama J2-J3 olarak elde edilir ve bu sıralama için Lmax değeri de 0 olarak bulunur.

Job (j)	1	2	3
operation	(4,1)	(4,2)	(4,3)
p_{4j}	-	5	3
r_{4j}	-	13	25
d_{4j}	-	22	28

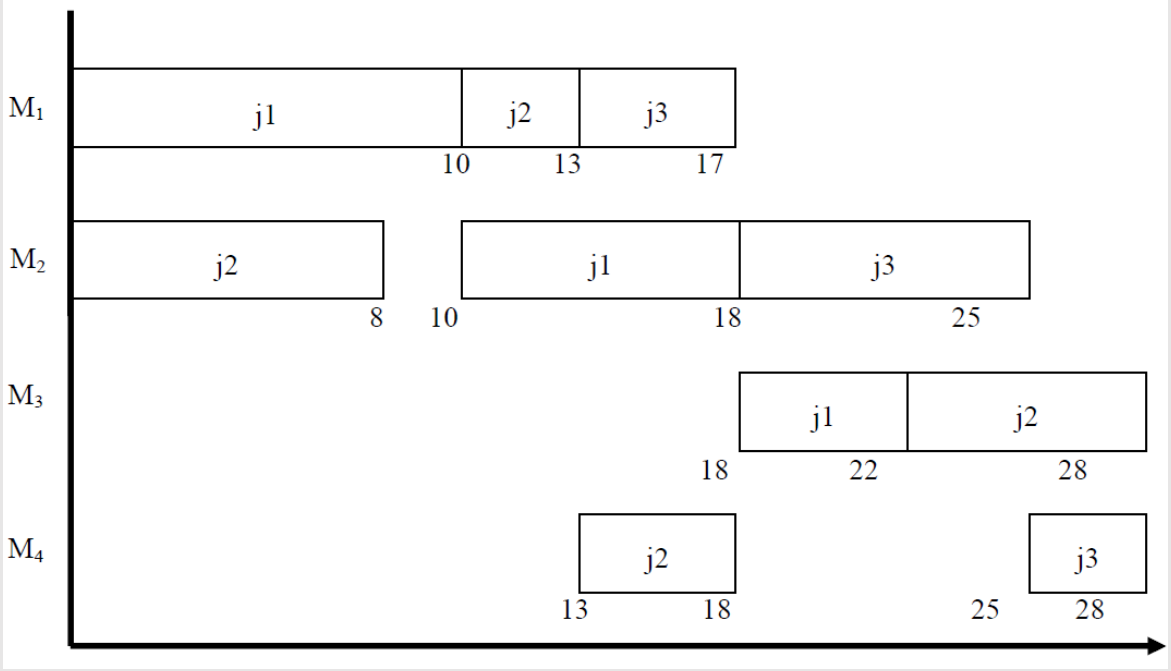
Job (j)	p_{4j}	r_{4j}	S_{4j}	C_{4j}	d_{4j}	L_{4j}
2	5	13	13	18	22	-4
3	3	25	25	28	28	0

Shifting Bottleneck (SB) Sezgiseli - Örnek

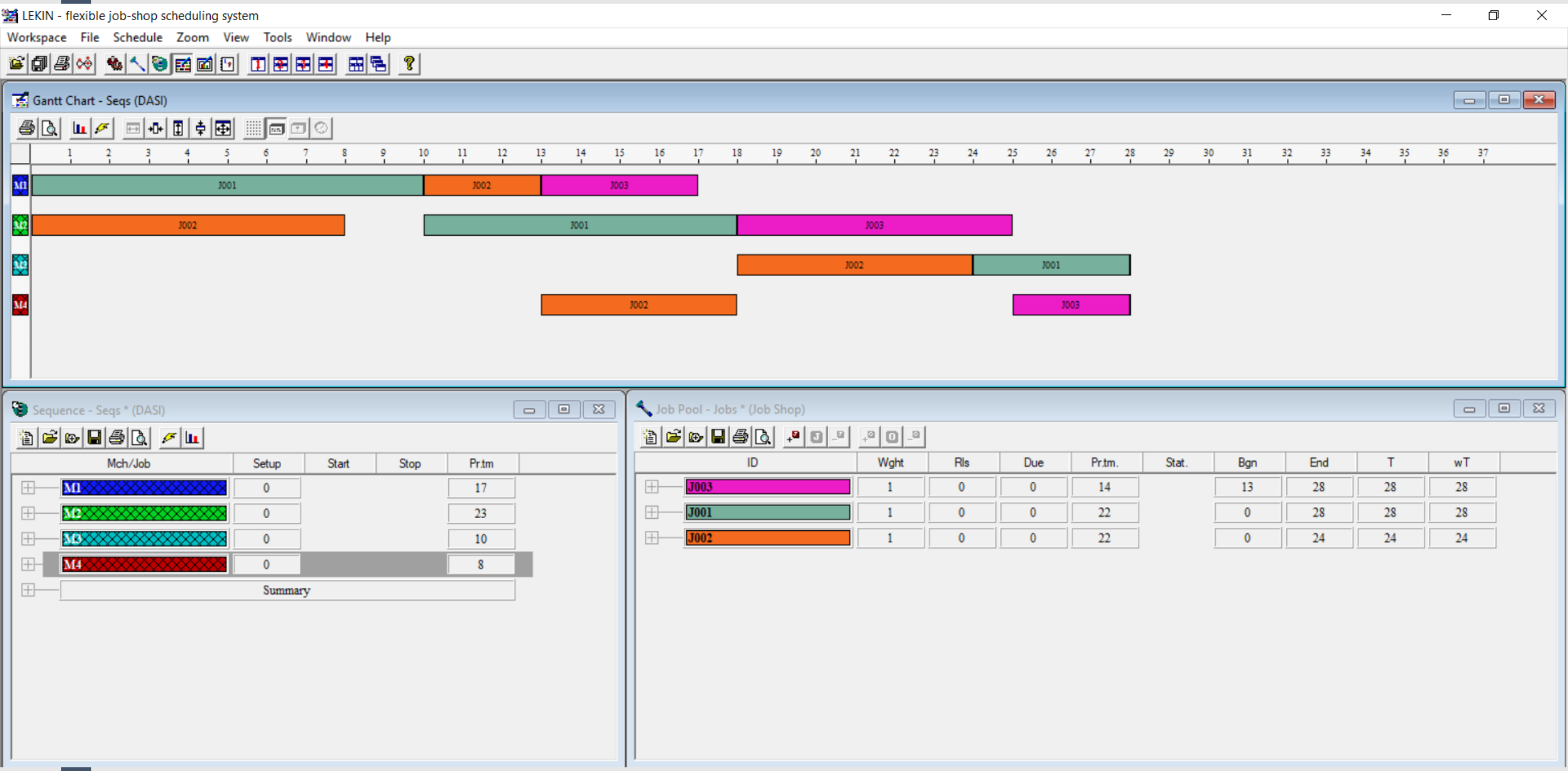
- En büyük gecikme (Lmax) değerleri M3 ve M4 mekineleri için 0 olduğundan, optimum Cmax değeri değişmez ve işlerin makinelerdeki sıralamaları aşağıdaki şekilde bulunur.

Machine	Job Sequence	Machine	Job Sequence
M_1	$\{j_1, j_2, j_3\}$	M_2	$\{j_2, j_1, j_3\}$
M_3	$\{j_1, j_2\}$	M_4	$\{j_2, j_3\}$

- Bu sıralamalara göre çizilen Gantt diyagramı da yanda verilmiştir:



LEKIN® Çıktısı



Bölüm Çalışma Soruları

Soru 1

Aşağıdaki problem için öncelikle uygun bir çizelge elde ediniz.

Job	Jobs routing	Processing times
1	1 - 2 - 3	5 - 10 - 12
2	1 - 3 - 2	4 - 3 - 8
3	3 - 2 - 1	9 - 6 - 7
4	2 - 3 - 1	7 - 5 - 11

- Tüm işler için teslim tarihi 27 olsaydı; yayılma zamanı, akış zamanları toplamı, toplam gecikme (tardiness) ve en büyük gecikme (Tmax) ne olurdu?

Soru 2

Consider $J_4 \mid\mid C_{max}$ problem with the following data:

Job	Jobs routing	Processing times
1	1 - 2 - 4	6 - 8 - 5
2	2- 3 - 4	4 - 4 - 3
3	4 - 2 - 1	8 - 6 - 4
4	2 - 3 - 4	5 - 10 - 15

Find the best makespan using the shifting bottleneck algorithm. Then, compute total flow time, total waiting time, and the utilization.

Soru 3

Consider $J_4 \mid\mid C_{max}$ problem using the following data:

Job	Jobs routing	Processing times
1	1 - 2 - 3	5 - 10 - 12
2	1 - 3 - 2	4 - 3 - 8
3	3 - 2 - 1	9 - 6 - 7
4	2 - 3 - 1	7 - 5 - 11

Find the best makespan using the shifting bottleneck algorithm

Soru 4

Using the SPT rule, develop a feasible schedule that minimizes makespan for the following jobs in a job shop problem:

Job	Jobs routing	Processing times
1	2 - 4 - 5	7 - 11 - 2
2	5 - 3	3 - 6
3	4 - 1 - 5 - 2	15 - 3 - 4 - 6
4	3 - 2 - 1 - 4 - 5	3 - 1 - 6 - 4 - 2
5	5 - 1 - 2 - 3 - 4	3 - 7 - 6 - 1 - 3
6	3 - 1 - 5 - 4	4 - 6 - 5 - 3

Soru 5

Using the EDD rule, develop a feasible schedule that minimizes the total lateness penalty for the following jobs in a job shop problem $J_5 \mid \mid \Sigma x_j$ problem:

Job	Job routing	Processing times	Due date	Late Penalty (PT)	Early Penalty (PE)
1	2-4-5	7-11-2	33	11	4
2	5-3	3-6	11	4	1
3	4-1-5-2	15-3-4-6	32	1	0
4	3-2-1-4-5	3-1-6-4-2	47	2	2
5	5-1-2-3-4	3-7-6-1-3	27	1	0
6	3-1-5-4	4-6-5-3	28	4	3

Where the total lateness penalty $\sum_{j=1} X_j$ is computed as follows:

$$x_j = \begin{cases} (C_j - d_j) * PT_j & if \ldots C_j - d_j > 0 \\ (d_j - C_j) * PE & if \ldots C_j - d_j \leq 0_j \end{cases} \cdots j = 1 \cdots 6$$

Where:

PT_j is the late penalty for job j

PE_j is the early penalty for job j.

Then, construct the Gantt chart to show your solution and then compute the total lateness penalty.

Soru 6

Consider $J_4 \mid\mid C_{max}$ problem and use the most work remaining (MWKR) dispatching rule to find the best makespan using the following data:

Job	Job routing	Processing times
1	1 - 2 - 3 - 4	6 - 8 - 13 - 5
2	1 - 2 - 3 - 4	4 - 1 - 4 - 3
3	4 - 2 - 1 - 3	3 - 8 - 6 - 4
4	2 - 1 - 3 - 4	5 - 10 - 15 - 4
5	1 - 2 - 4 - 3	3 - 4 - 6 - 4
6	3 - 1 - 2 - 4	4 - 2 - 4 - 5

Soru 7

Consider the following instance of the $J_3 \mid\mid C_{max}$ problem.

Job	Machine Sequence	Process Times
1	1-2-3	p11=5, p21=4, p31=6
2	2-1	p22=5,p_{12}=7
3	2-3	p23 =5, p_{33} =5

Apply the Shifting bottleneck heuristic for this instance and find C_{max}. Also, draw Gantt chart for your solution.

Soru 8

Consider $J_3 \mid\mid C_{max}$ with the following data:

Job	M/C routing	Processing times	Due date
1	1 - 2 - 3	$P_{11} = 4, P_{12} = 3, P_{13} = 2$	9
2	2 - 1 - 3	$P_{22} = 4, P_{21} = 1, P_{23} = 4$	9
3	3 - 2 - 1	$P_{33} = 3, P_{32} = 2, P_{31} = 4$	9
4	2 - 3 - 1	$P_{42} = 3, P_{43} = 1, P_{41} = 5$	9

Use the data above to solve the job shop problem according to SPT, Slack time, and CR rules. In each case, show your feasible solution on Gantt chart and compute C_{max}, T_{max} $\overline{\mathbf{F}}$, and $\overline{\mathbf{T}}$. According to our objective function which schedule is better?

LABORATUVAR UYGULAMASI

Çizelgeleme - Laboratuvar Uygulaması, Soru 1

- Verilen problem için Shifting Bottleneck (SB) sezgiseli ile yayılma zamanını (Cmax) minimize eden çizelgeyi önce manuel olarak ve sonrasında LEKIN programıyla oluşturunuz.

İş	Rota	İşlem Zamanları
1	1, 2, 3, 4	p_{11}=9, p_{21}=8, p_{31}=4, p_{41}=4
2	1, 2, 4, 3	p_{12}=5, p_{22}=6, p_{42}=3, p_{32}=6
3	3, 1, 2, 4	p_{33}=10, p_{13}=4, p_{23}=9, p_{43}=2

Çizelgeleme - Laboratuvar Uygulaması, Soru 2

a) Aşağıda verilen F4||Cmax problemini öncelikle CDS algoritması ile çözünüz.

Job	1	2	3	4	5
1	1	10	17	12	11
2	13	12	9	17	3
3	6	18	13	2	5
4	2	18	4	6	16

b) Aynı problemi LEKIN üzerinden öncelik kuralları ve sezgisel kullanarak çözünüz ve elde ettiğiniz sonuçları kıyaslayınız.

CDS algoritması için genelleştirilmiş formül:

This implies that the surrogate problems data will be in generated as follows:

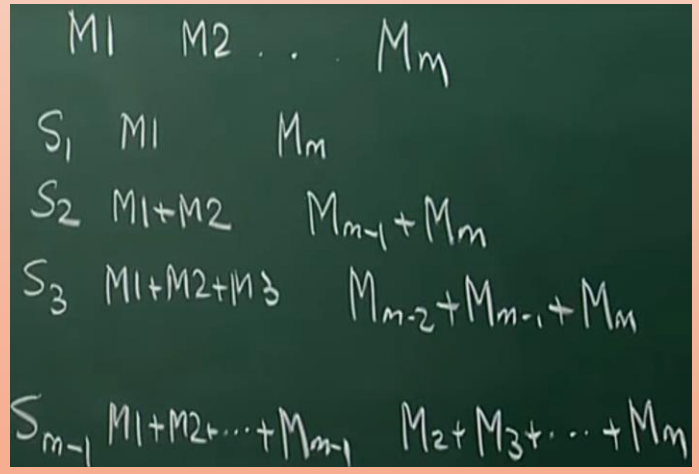
For k = 1, ……, m-1 and j = 1, ……, n then,

$$M_1' = \sum_{i}^{k} P_{ij} \quad \text{and} \quad M_2' = \sum_{i=m-k+1}^{m} P_{ij}$$

Where:

M_1' = the processing time for the first machine

M_2' = the processing time for the second machine



INTRODUCTION TO LEKIN

Gareth Beddoe

Introduction to LEKIN

- What is LEKIN?
- Machine Environments
- Methods Employed
- Graphical User Interface
- Setting up the Environment
- 2 Examples
  - *Single Machine Environment*
  - *Flow Shop Environment*

What is LEKIN?

- Interactive scheduling system for machine environments
- Ideal for research and teaching
  - *Graphical Interface*
  - *Built in dispatching rules and simple heuristic methods*
  - *User-defined algorithms can be added*
- Educational Version:
  - *50 jobs, 20 work-centres maximum*
  - *Windows 98 or NT*

Who wrote LEKIN?

- Stern School of Business, NYU
  - *Michael Pinedo et. al.*
    - http://www.stern.nyu.edu/om/pinedo/

- Download (educational version):
  - *http://www.stern.nyu.edu/om/pinedo/lekin*

- Reference:

  Pinedo M, *Scheduling: Theory, Algorithms, and Systems (2nd Edition)*, Prentice Hall 2002: pp 493-499

Machine Environments

- Single Machine
- Parallel Machines

- Flow Shop
- Job Shop

- Flexible Flow Shop
- Flexible Job Shop

Generalisations: more than one machine of each type

Methods: Dispatching Rules

- EDD, MS, LPT, SPT, WSPT
- FCFS: (F)irst (C)ome (F)irst (S)erve
- ATCS: Apparent Tardiness Cost (with Setups).
  - *Optimizes the Total Weighted Tardiness.*
  - *Trade-off between MS and WSPT*
- CR : Critical Ratio rule.
  - *Schedules jobs according to the ratio of the time left until the due date and the remaining processing time.*
  - *Trade-off between EDD and LPT.*

Methods: Built-in Heuristics

- Shifting Bottle-neck Heuristics
  - *General SB Routine (most objectives)*
  - *Objective Specific routines:*
    - SB/sum wT: Total Weighted Tardiness
    - SB/Tmax: Maximum Tardiness, Makespan
- Local Search Heuristic
  - *For all objectives*
- Hybrid Method:
  - *SB-LS: Combination of Shifting Bottle-neck and Local Search heuristics*

Methods: User-defined Heuristics

- Users can write new heuristics methods and use the “plug-in” feature
- Operation as external executables with standardised input and ouput parameters
- Allows researchers to test and develop new algorithms in an interactive environment.
- Facilitates comparison between various methods

Objectives

- Makespan
- The Maximum Tardiness
- The Total Number of Late Jobs
- The Total Flow Time
- The Total Tardiness
- The Total Weighted Flow Time
- The Total Weighted Tardiness

$$C_{\max}$$

$$T_{\max}$$

$$\sum U_j$$

$$\sum C_j$$

$$\sum T_j$$

$$\sum w_j C_j$$

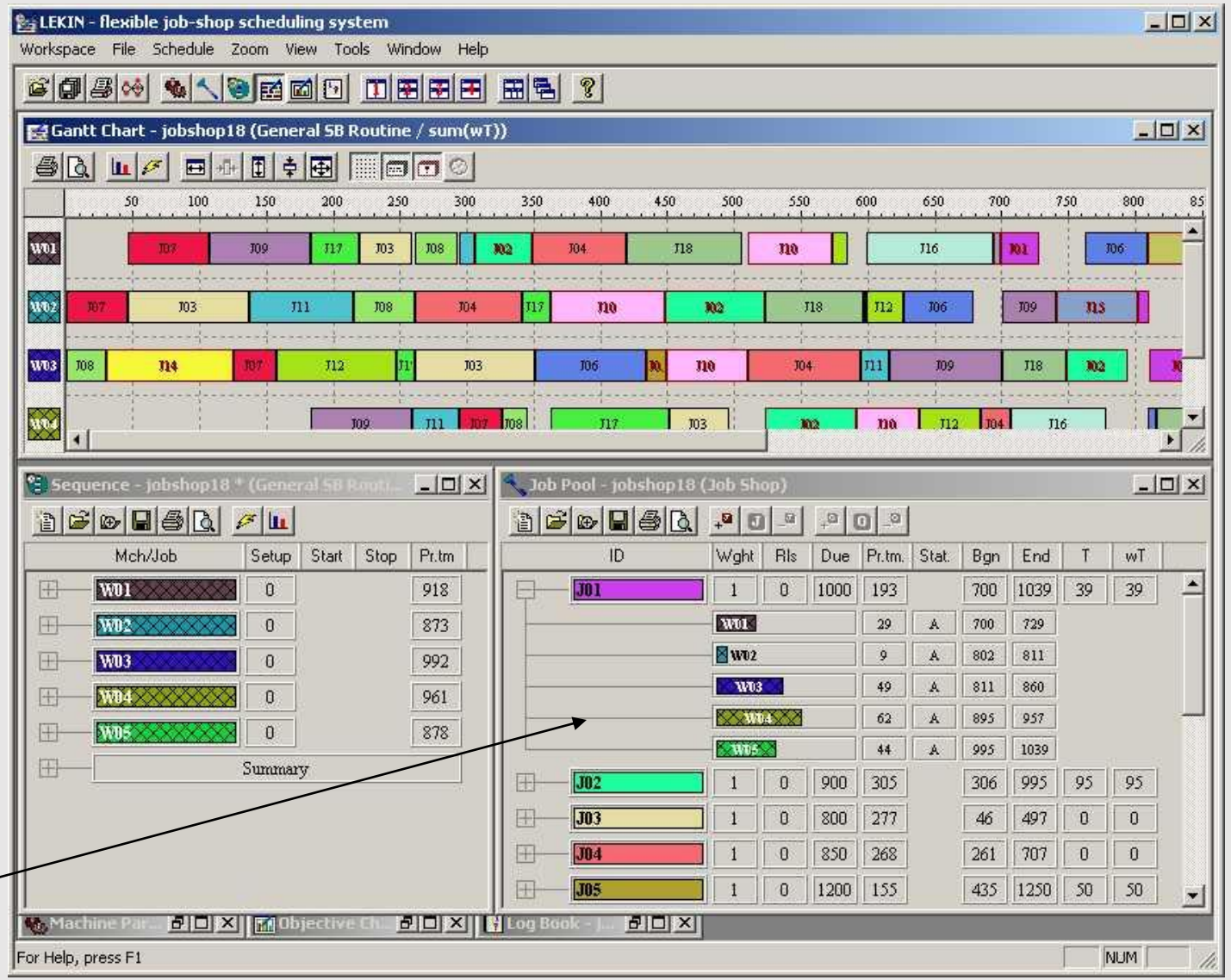
$$\sum w_j T_j$$

Graphical User Interface

Solution Schedule

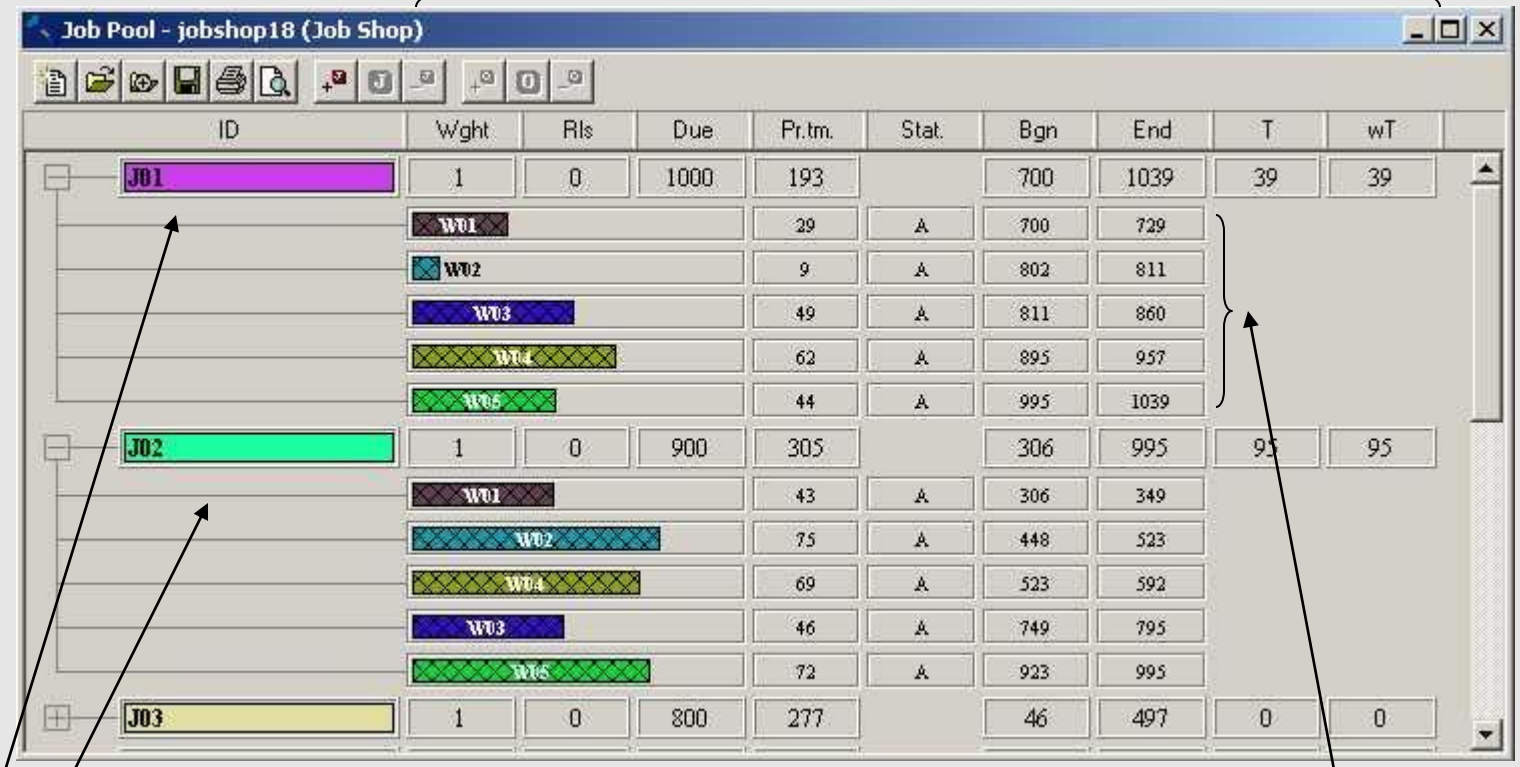
Machine Information

Job Information



Job Pool Window

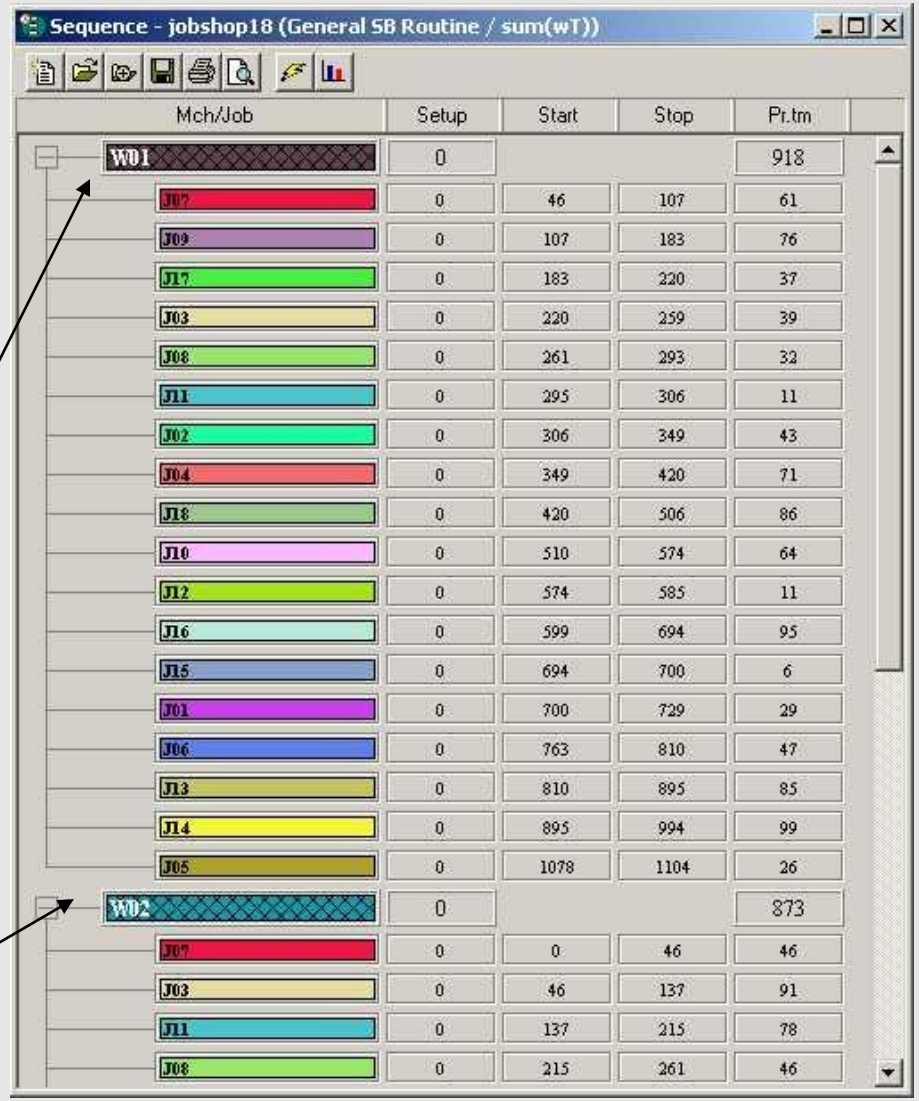
Job statistics and settings



Jobs

Sequence through machines and start and end times for each machine

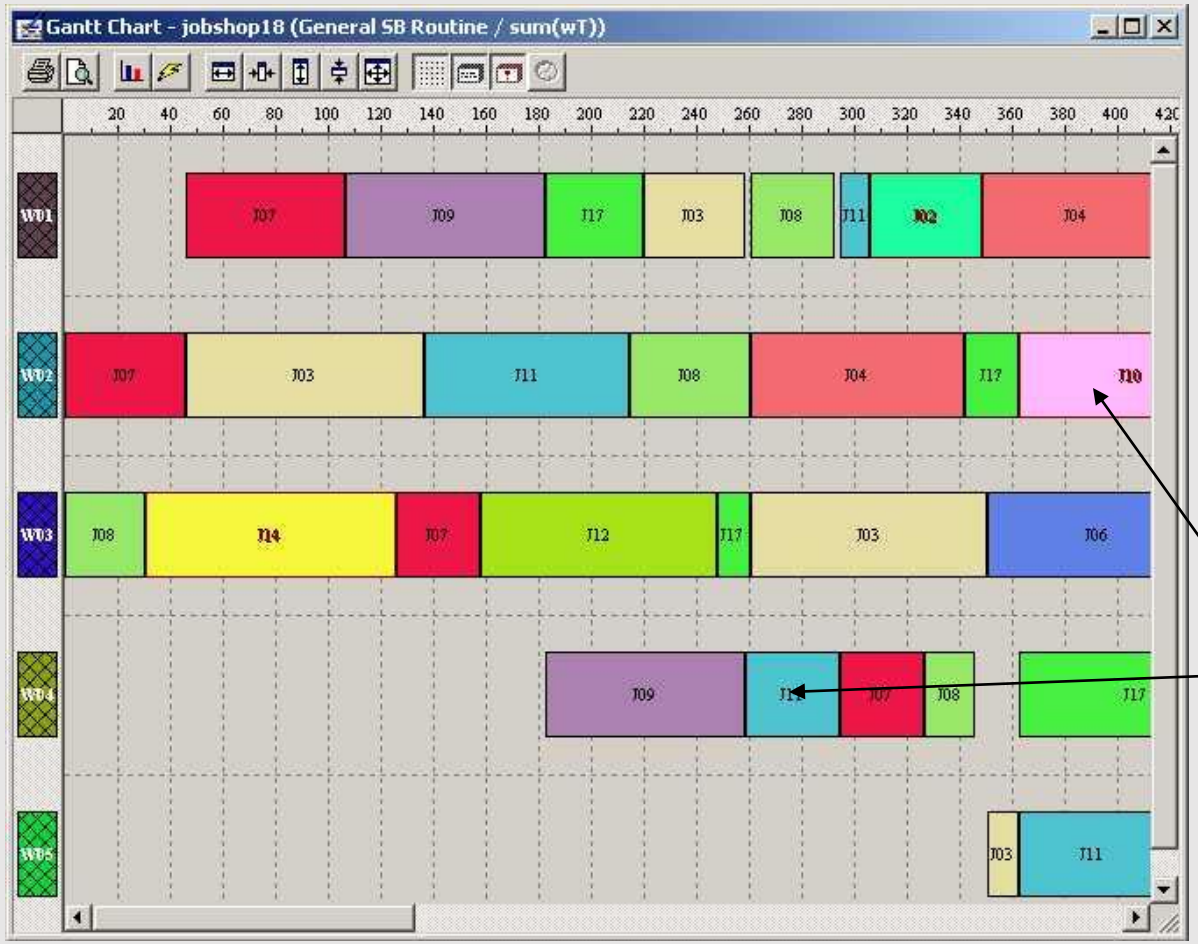
Sequence Window



Machines (Workcenters)

Sequence of jobs through machine and start and end times

Gantt Chart (Schedule) Window

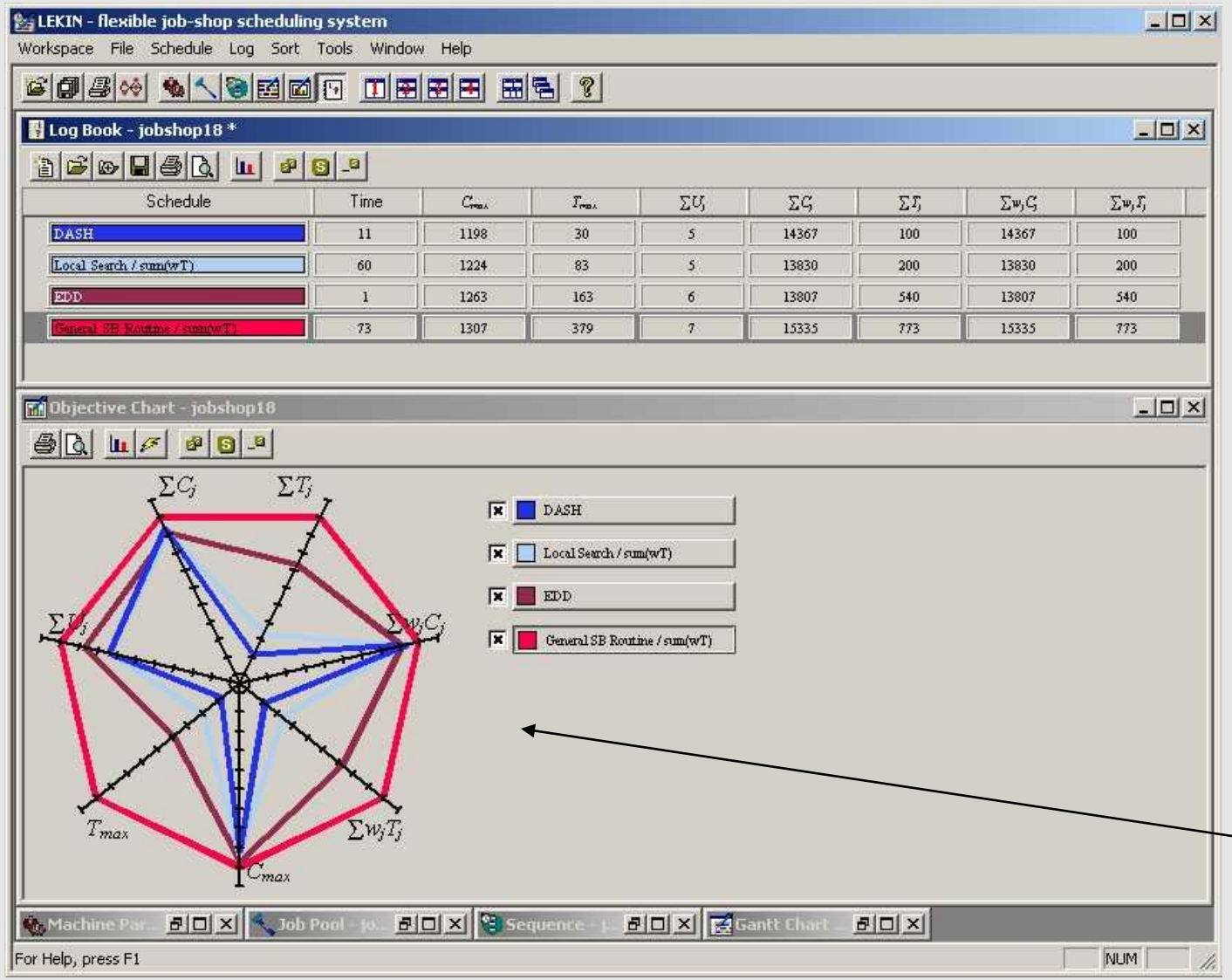


Machines
(Workcenters)

Jobs

Time

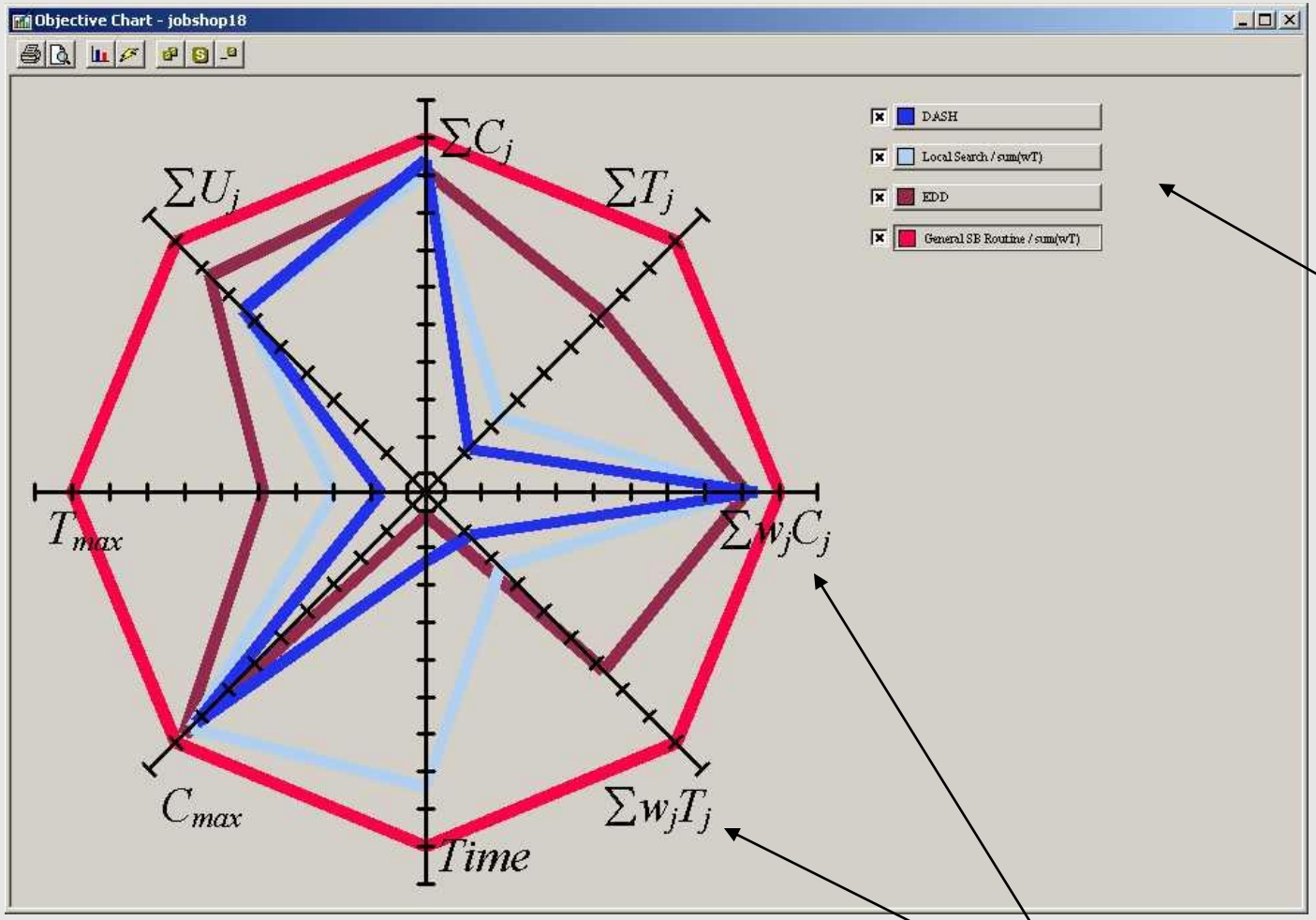
Displaying Results



Log of previous solutions

Objective Performance

Performance Comparisons



Various Solutions

Objectives

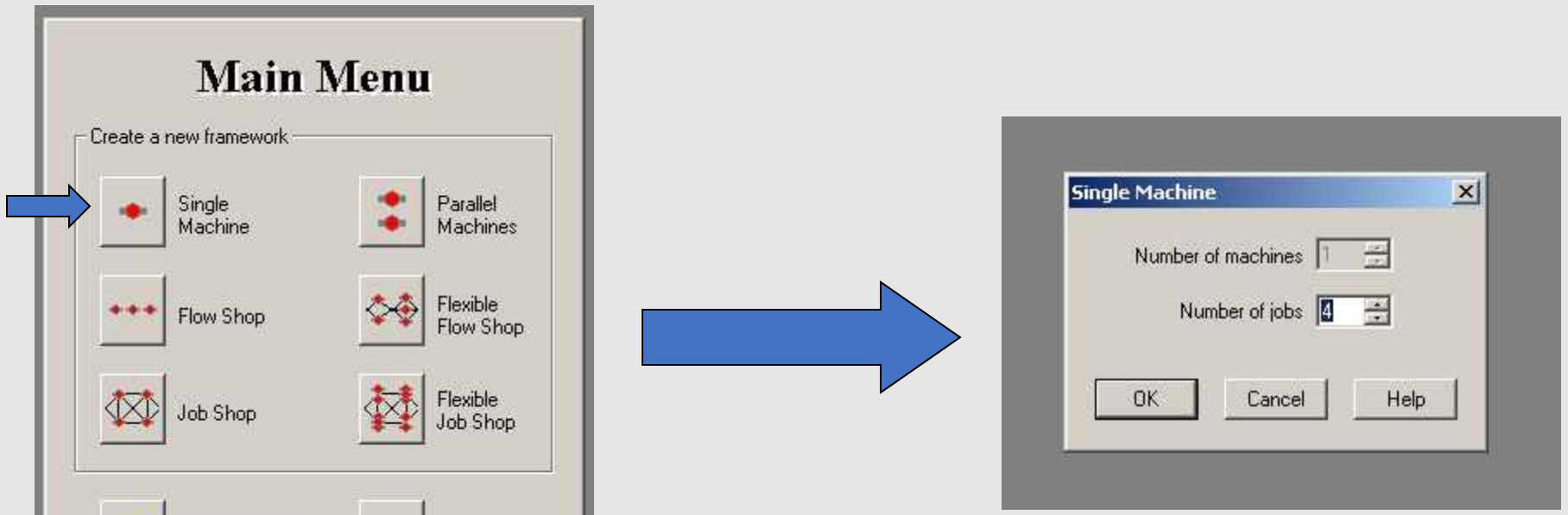
Basic Setup Procedure

1) Enter Machine Information
   - *Number of Machines*
   - *Availability Time*
   - *Setup Time Information*
2) Enter Job Information
   - *Number of Jobs*
   - *Release Dates, Due Dates, Weight, and Route*
3) Select a dispatching rule or heuristic and generate schedule

Example 1: Single Machine

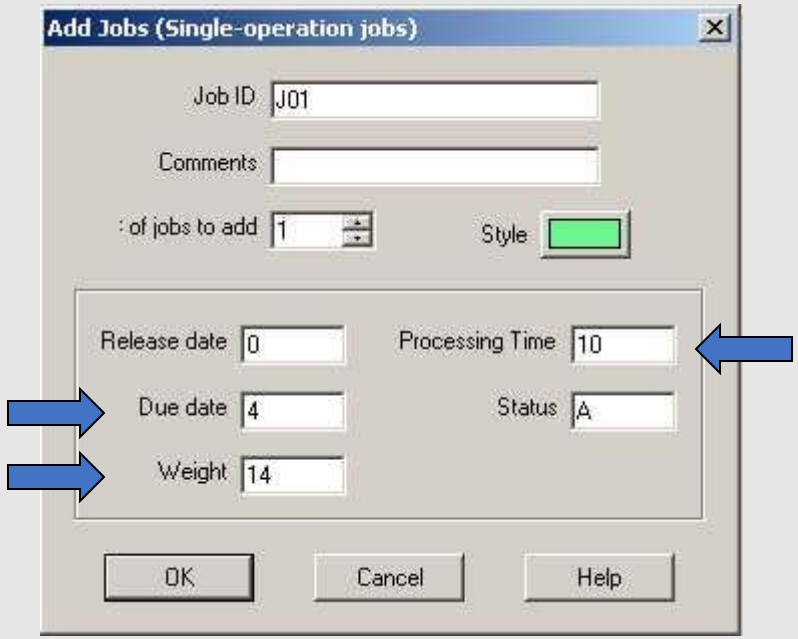
jobs	1	2	3	4
p_j	10	10	13	4
d_j	4	2	1	12
w_j	14	12	1	12

Setting up the problem (1)



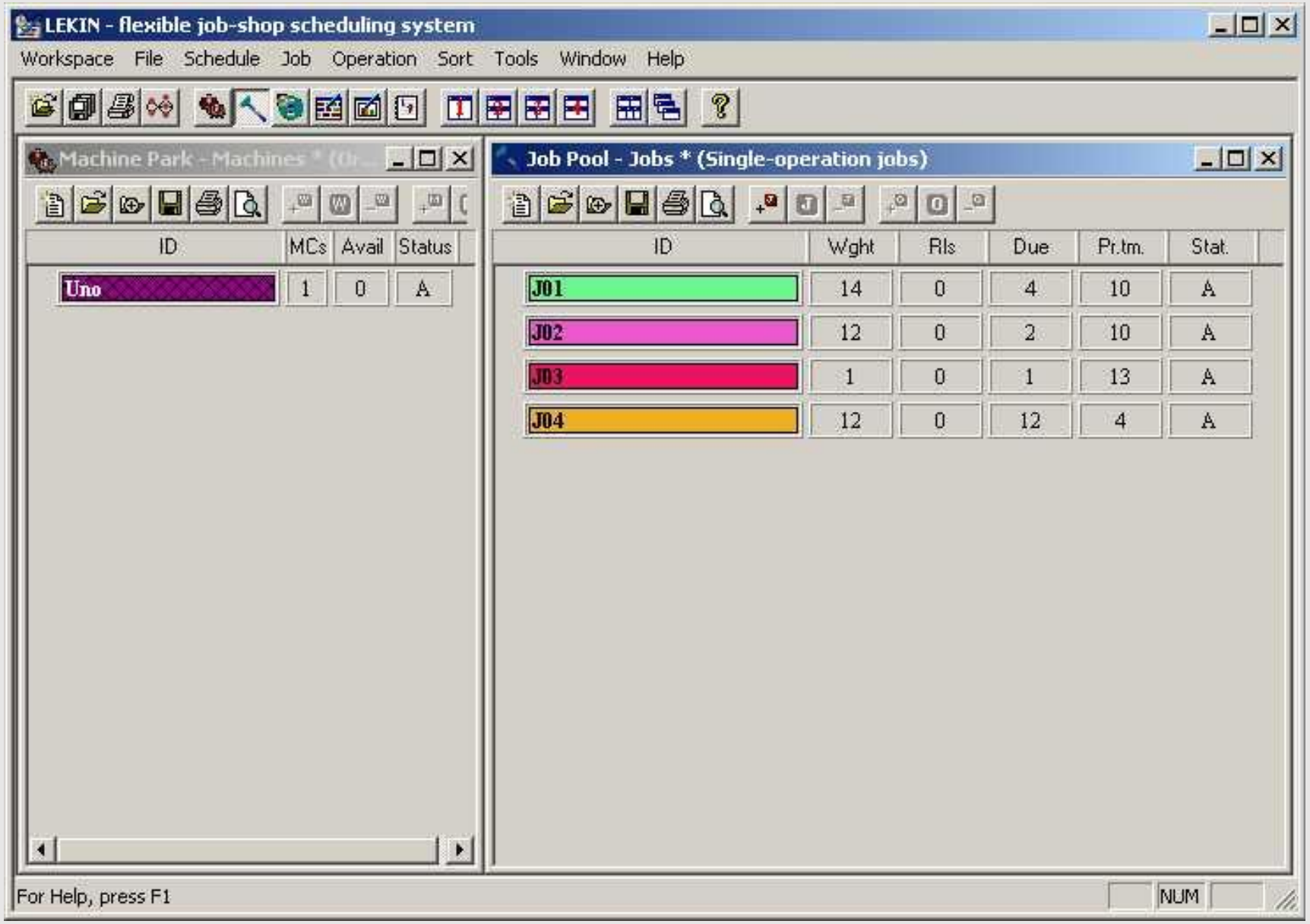
1) Choose Single Machine Environment
2) Number of machines already set (= 1)
3) Choose number of jobs (= 4)

Setting up the problem (2)

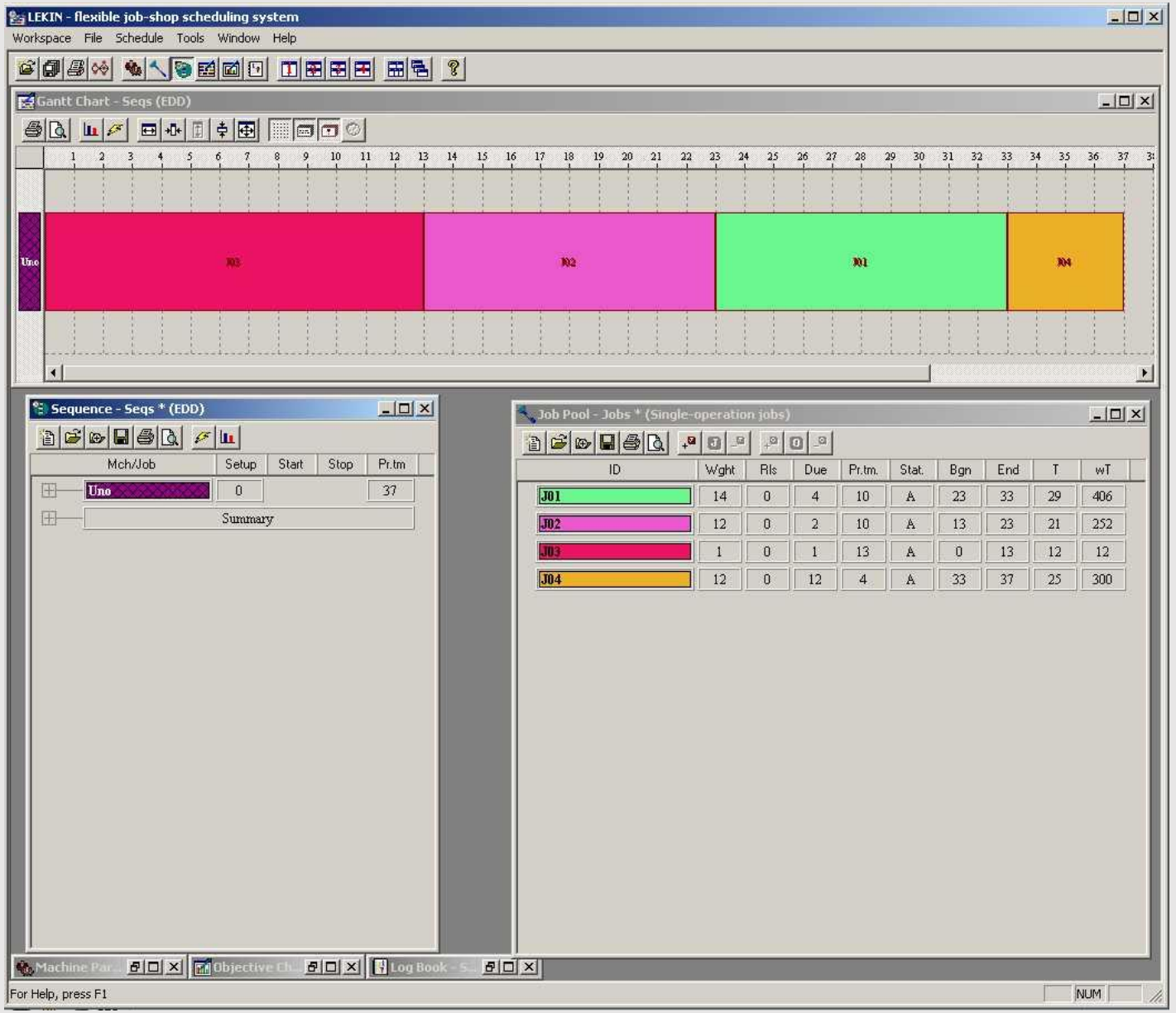


- For each job:
  - *Enter Due Date, Processing Time, and Weight*
  - *Click OK*

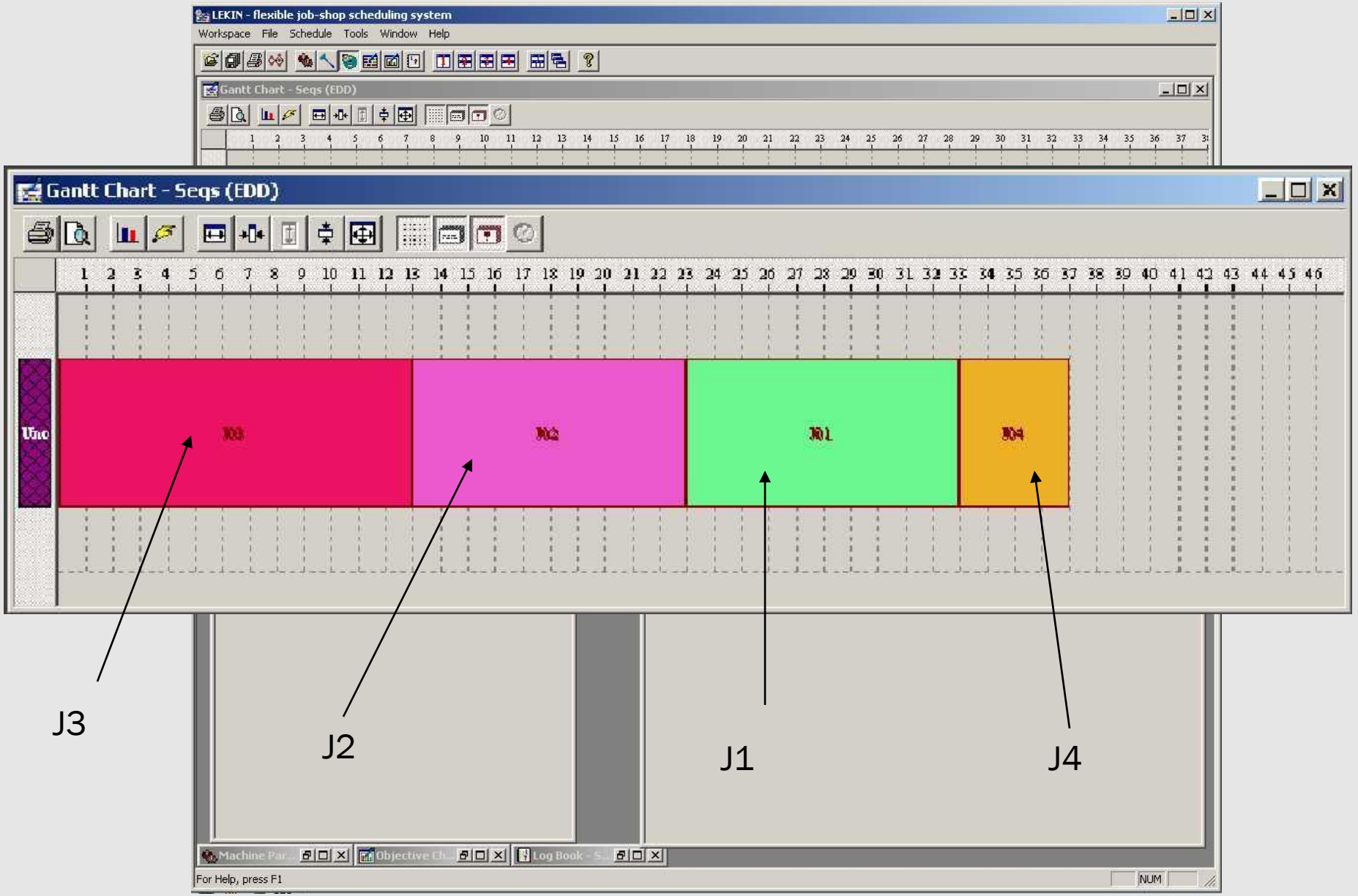
Environment Display



Schedule!


LEKIN - flexible job-shop scheduling system
Workspace File Schedule Tools Window Help
Gantt Chart - Seqs (EDD)
Uno
J03
J02
J01
J04
Sequence - Seqs * (EDD)
Mch/Job Setup Start Stop Pr.tm
Uno 0 37
Summary
Job Pool - Jobs * (Single-operation jobs)
ID Wght Rls Due Pr.tm. Stat. Bgn End T wT
J01 14 0 4 10 A 23 33 29 406
J02 12 0 2 10 A 13 23 21 252
J03 1 0 1 13 A 0 13 12 12
J04 12 0 12 4 A 33 37 25 300
Machine Par...
Objective Ch...
Log Book - S...
For Help, press F1
NUM

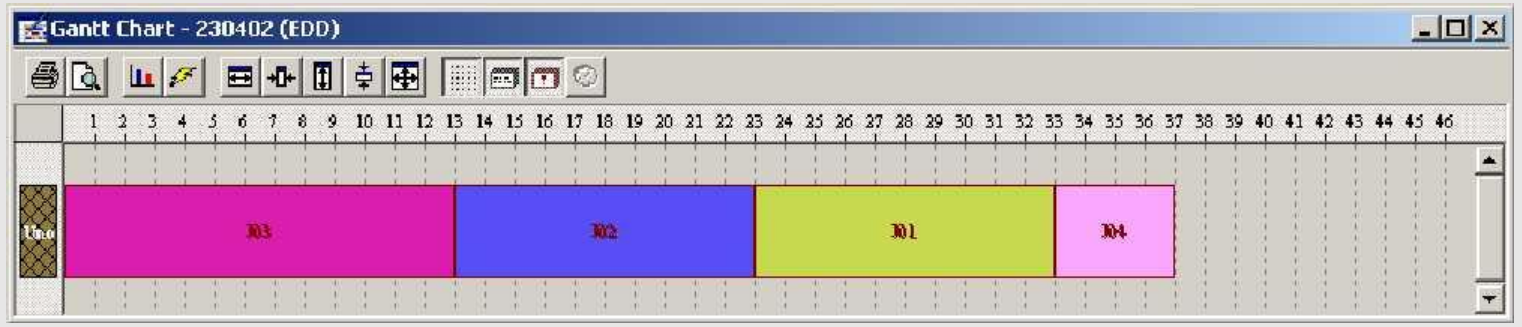
Schedule!


LEKIN - flexible job-shop scheduling system
Workspace File Schedule Tools Window Help
Gantt Chart - Seqs (EDD)
Gantt Chart - Seqs (EDD)
J3
J2
J1
J4
Machine Par...
Objective Ch...
Log Book - S...
For Help, press F1
NUM

More Solutions

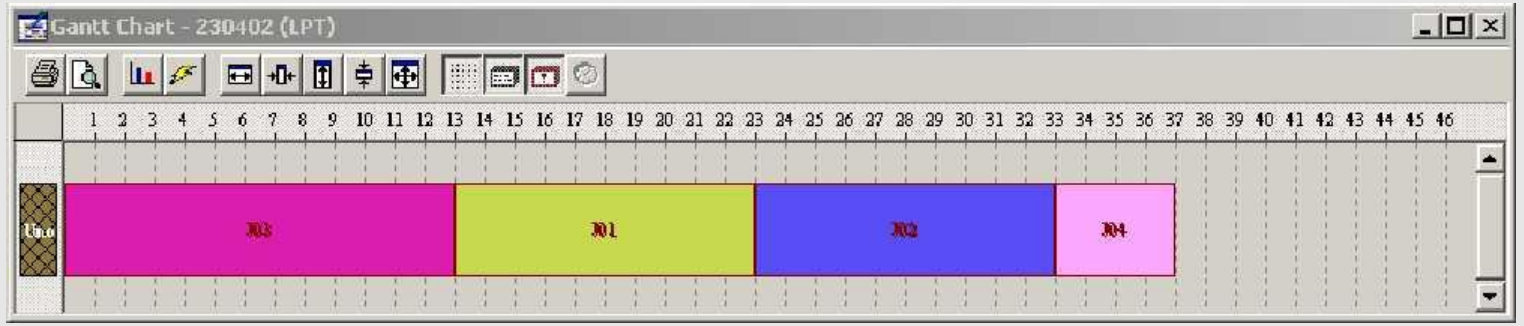
EDD:

(3214)



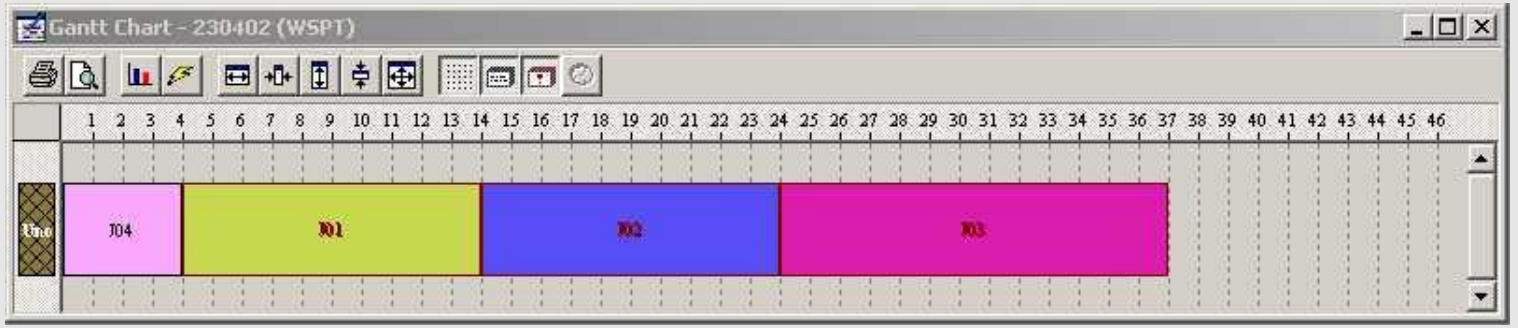
LPT:

(3124)



WSPT:

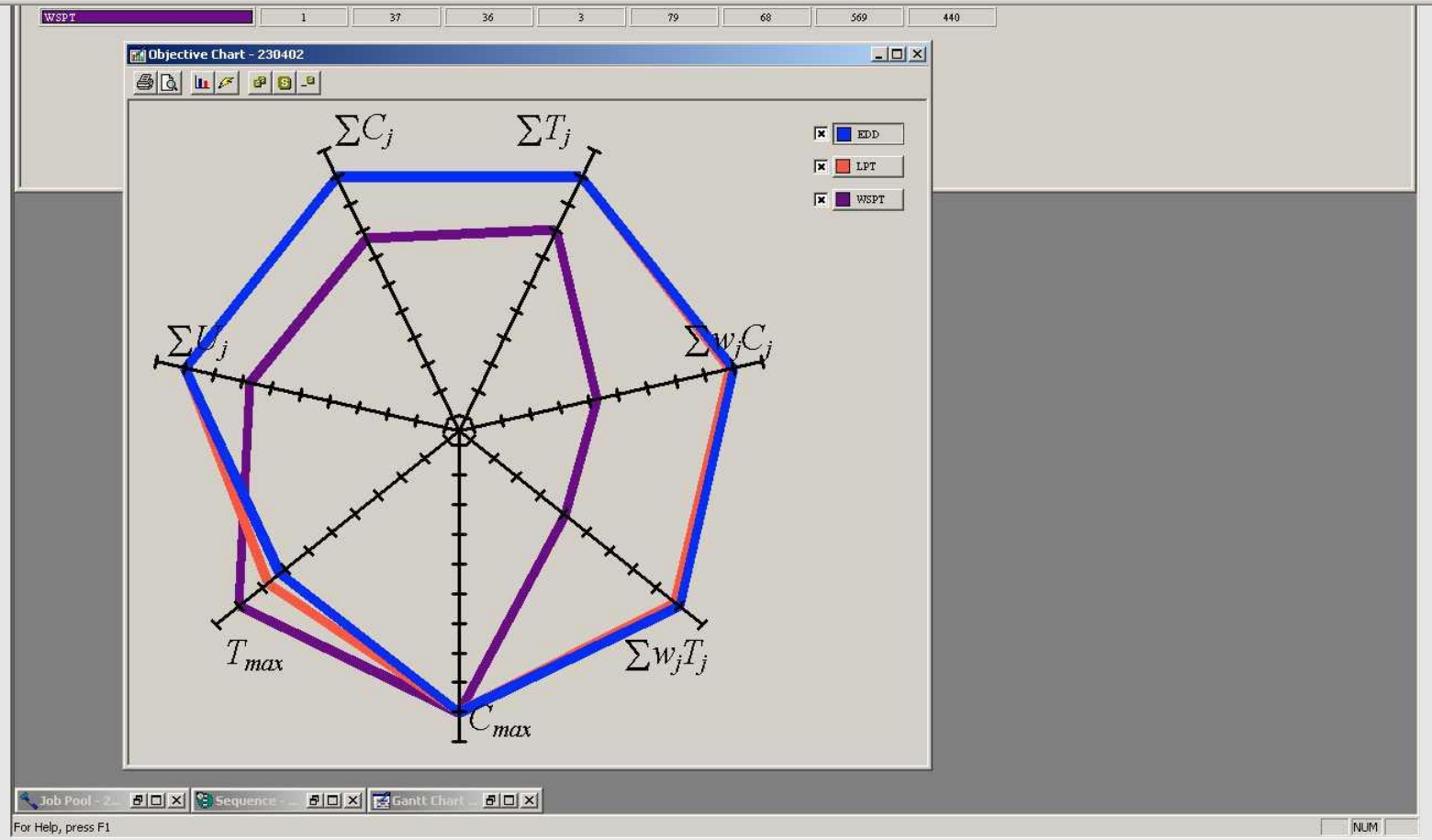
(4123)



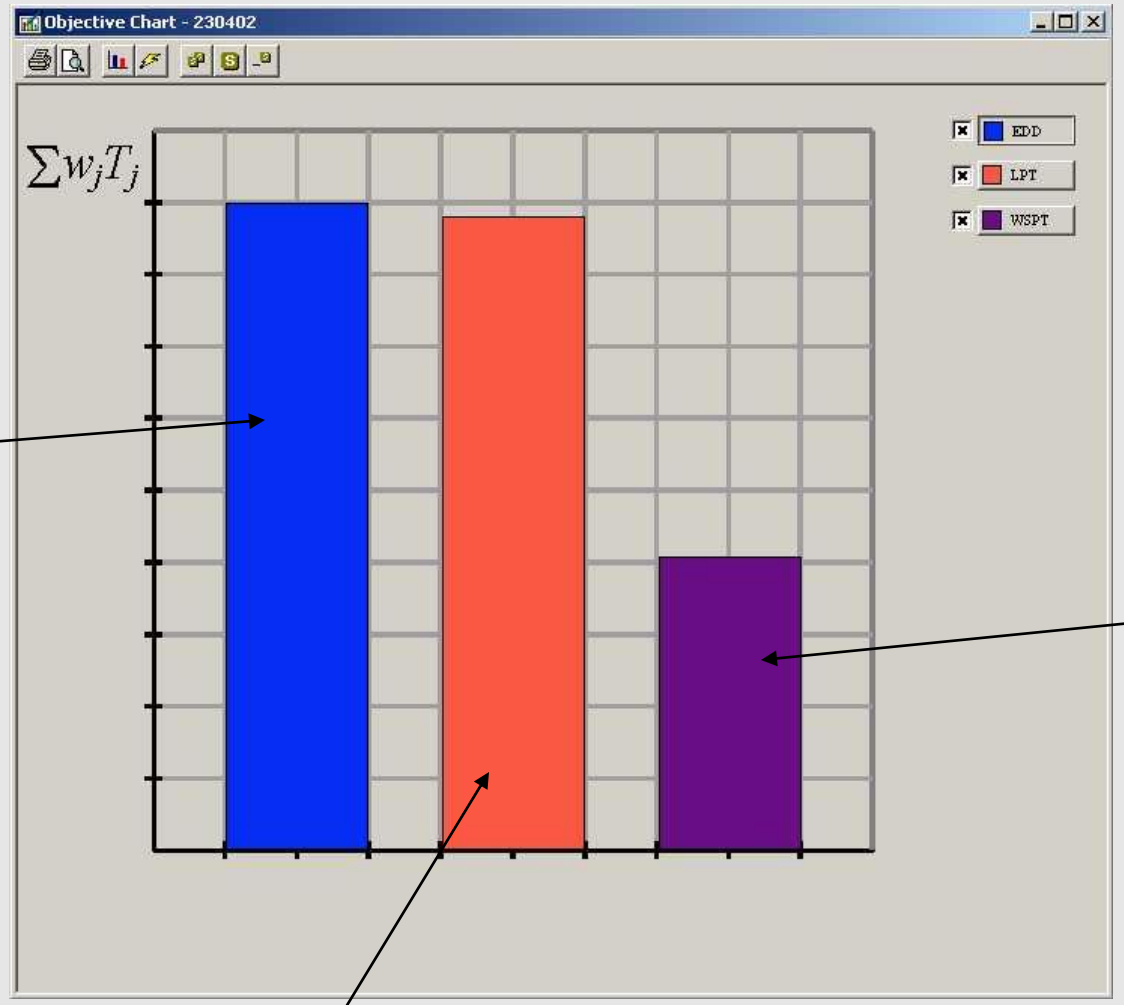
Comparison

Log Book - 230402

Schedule	Time	C_{max}	T_{max}	$\sum U_j$	$\sum C_j$	$\sum T_j$	$\sum w_j C_j$	$\sum w_j T_j$
EDD	1	37	29	4	106	87	1195	970
LPT	1	37	31	4	106	87	1175	950
WSPT	1	37	36	3	79	68	569	440



More Comparison



EDD

WSPT

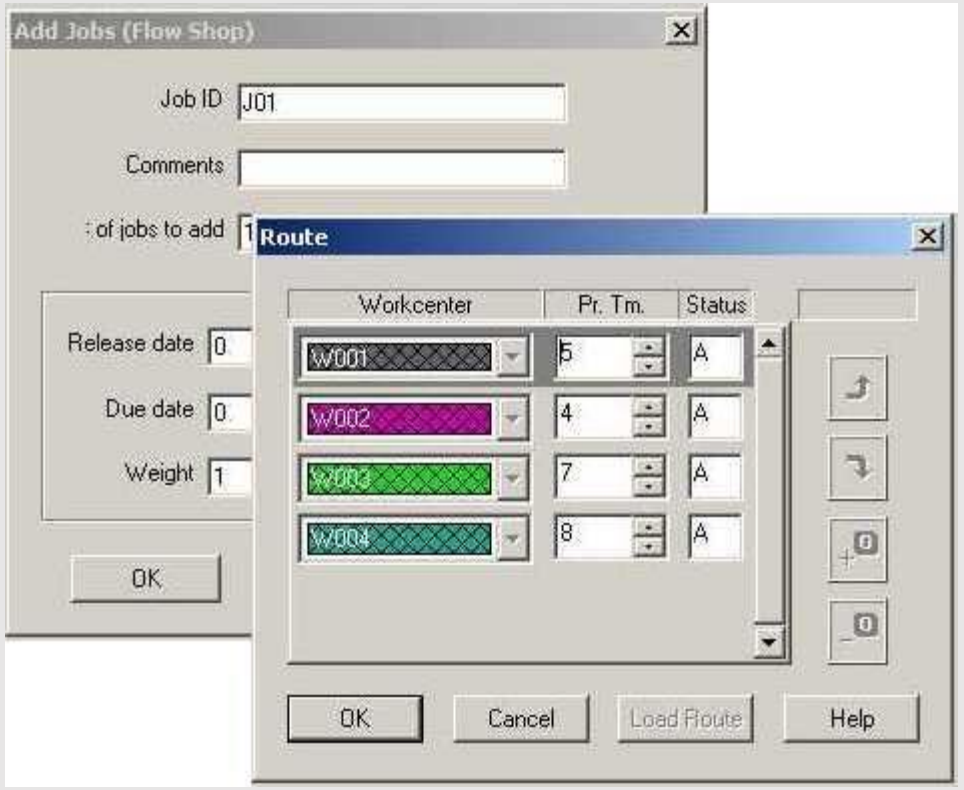
LPT

Example 2: Flow Shop

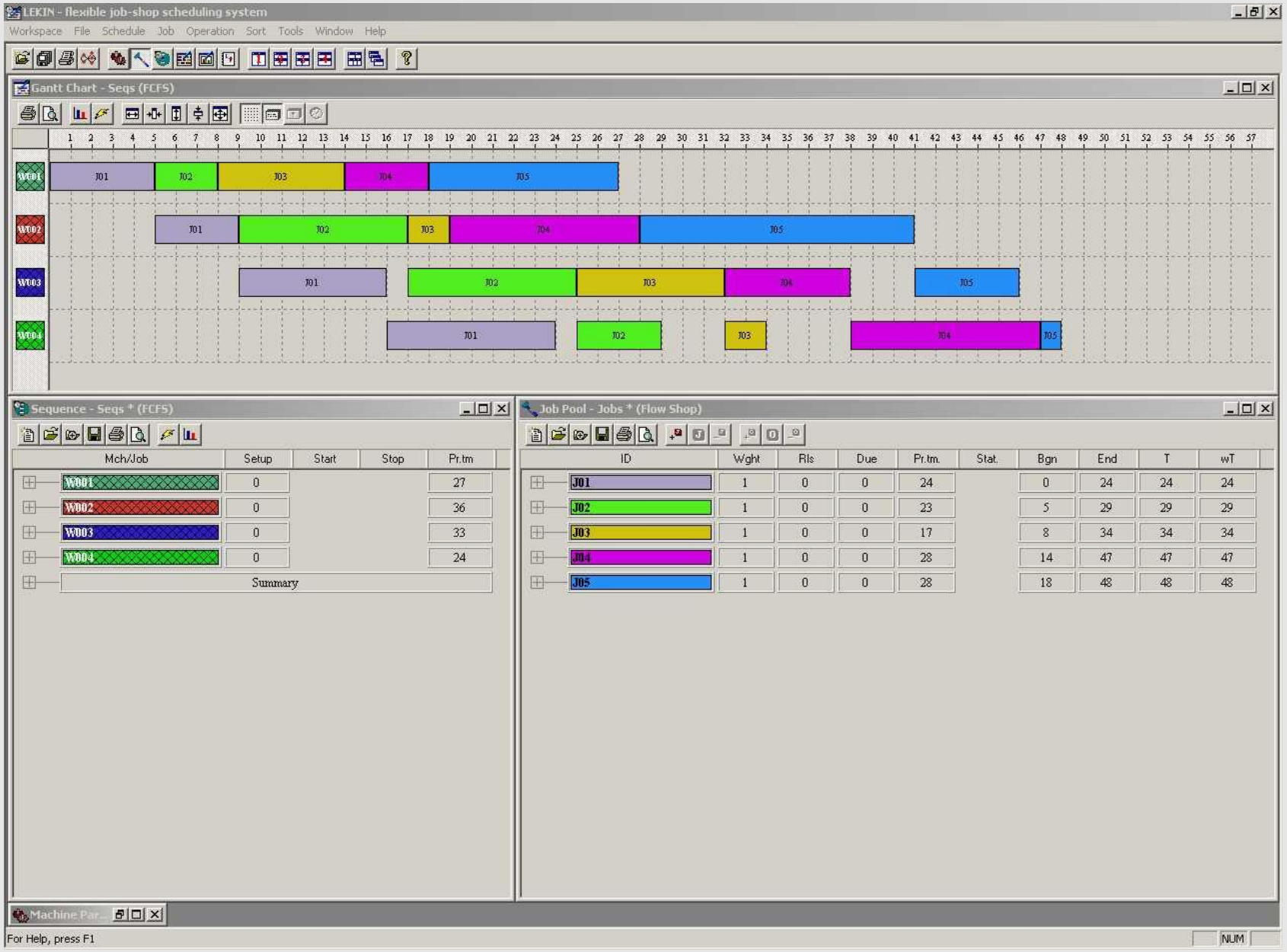
jobs	1	2	3	4	5
p_{1j}	5	3	6	4	9
p_{2j}	4	8	2	9	13
p_{3j}	7	8	7	6	5
p_{4j}	8	4	2	9	1

Setting up the problem

- Machine (Workcenter) setup
- Establishing machine route for jobs



Schedule!


LEKIN - flexible job-shop scheduling system
Workspace File Schedule Job Operation Sort Tools Window Help
Gantt Chart - Seqs (FCFS)
Sequence - Seqs * (FCFS)
Mch/Job Setup Start Stop Pr.tm
W001 0 27
W002 0 36
W003 0 33
W004 0 24
Summary
Job Pool - Jobs * (Flow Shop)
ID Wght Rls Due Pr.tm. Stat. Bgn End T wT
J01 1 0 0 24 0 24 24 24
J02 1 0 0 23 5 29 29 29
J03 1 0 0 17 8 34 34 34
J04 1 0 0 28 14 47 47 47
J05 1 0 0 28 18 48 48 48
Machine Par...
For Help, press F1
NUM

Other LEKIN features

- Manual Schedule Adjustment
  - *useful for determining neighbourhood definitions in local search development*
- Large library of standard problems included in package
- Industrial version currently in development
  - *will be able to handle a much larger machine environment*
  - *will include considerably more dispatching rules and built in heuristics*

Summary

- Graphics based interactive machine shop scheduling system
- Ability to schedule a number of different machine environments
- Valuable as an educational and research tool
- Extendible with new heuristic techniques

Kaynaklar

- Principles of Sequencing and Scheduling, Kenneth R. Baker, Dan Trietsch, John Wiley & Sons, New Jersey, 2009.
- Algorithms for Sequencing and Scheduling, Ibrahim M. Alharkan, King Saud University.
- Scheduling: Theory, Algorithms, and Systems, Michael Pinedo, Springer, 2012.
- Üretimde Sıralama ve Çizelgeleme Ders Notları, Yrd.Doç.Dr. A. Ayça Supçiller, Pamukkale Üniversitesi, 2014.
- İş Sıralama ve Çizelgeleme Ders Notları, Prof.Dr. Hüseyin Başlıgil, Yıldız Teknik Üniversitesi, 2013.
- Üretim Çizelgeleme Ders Notları, Yrd.Doç.Dr. Mert Topoyan, Dokuz Eylül Üniversitesi, 2017.
- Üretim ve Servis Sistemlerinde Planlama ve Çizelgeleme, Yad. Doç. Dr. Zehra Kamışlı Öztürk, Anadolu Üniversitesi, 2012