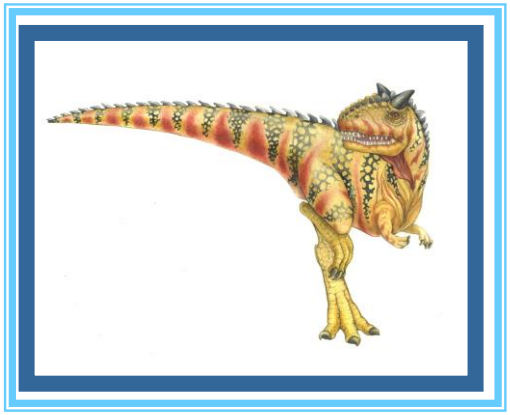
Bölüm 5: Süreç İş Planlaması

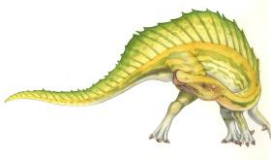


Süreç İş Planlaması

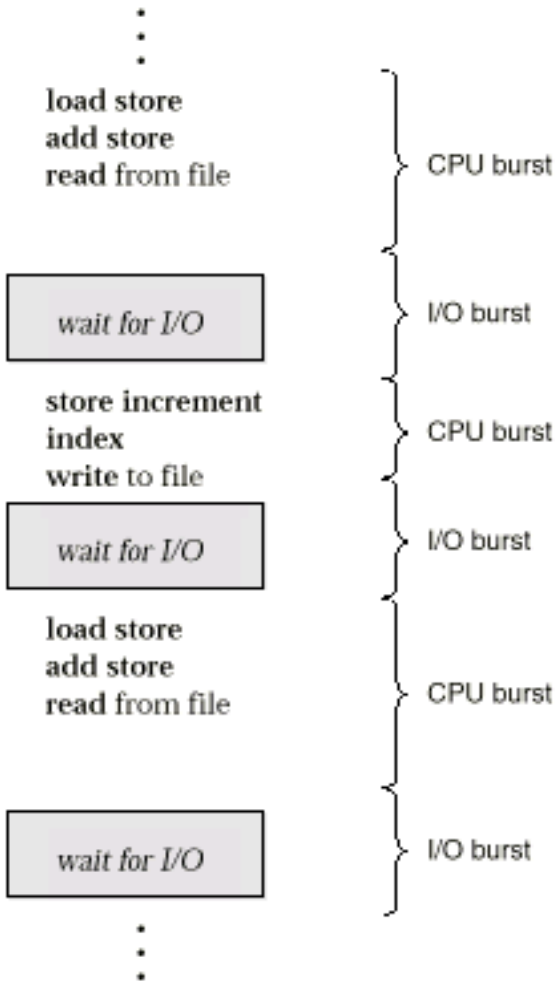
- Temel Kavramlar
- Görev Yönetim Kriterleri
- Görev Yönetim Algoritmaları

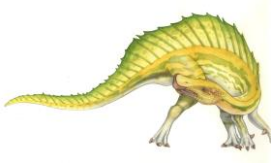
Temel Kavramlar

- En üst düzeyde işlemci kullanımı çoklu iş düzeni ile sağlanır
- CPU-I/O Burst Cycle (İşlemci-G/Ç Çoğuşma Çevrimi/Döngüsü)– süreç işletimi, işlemci çoğuşması ve G/Ç bekleme çevriminden oluşur
- İşlemci çoğuşma dağıtımı işletim sisteminin temel görevlerinden biridir

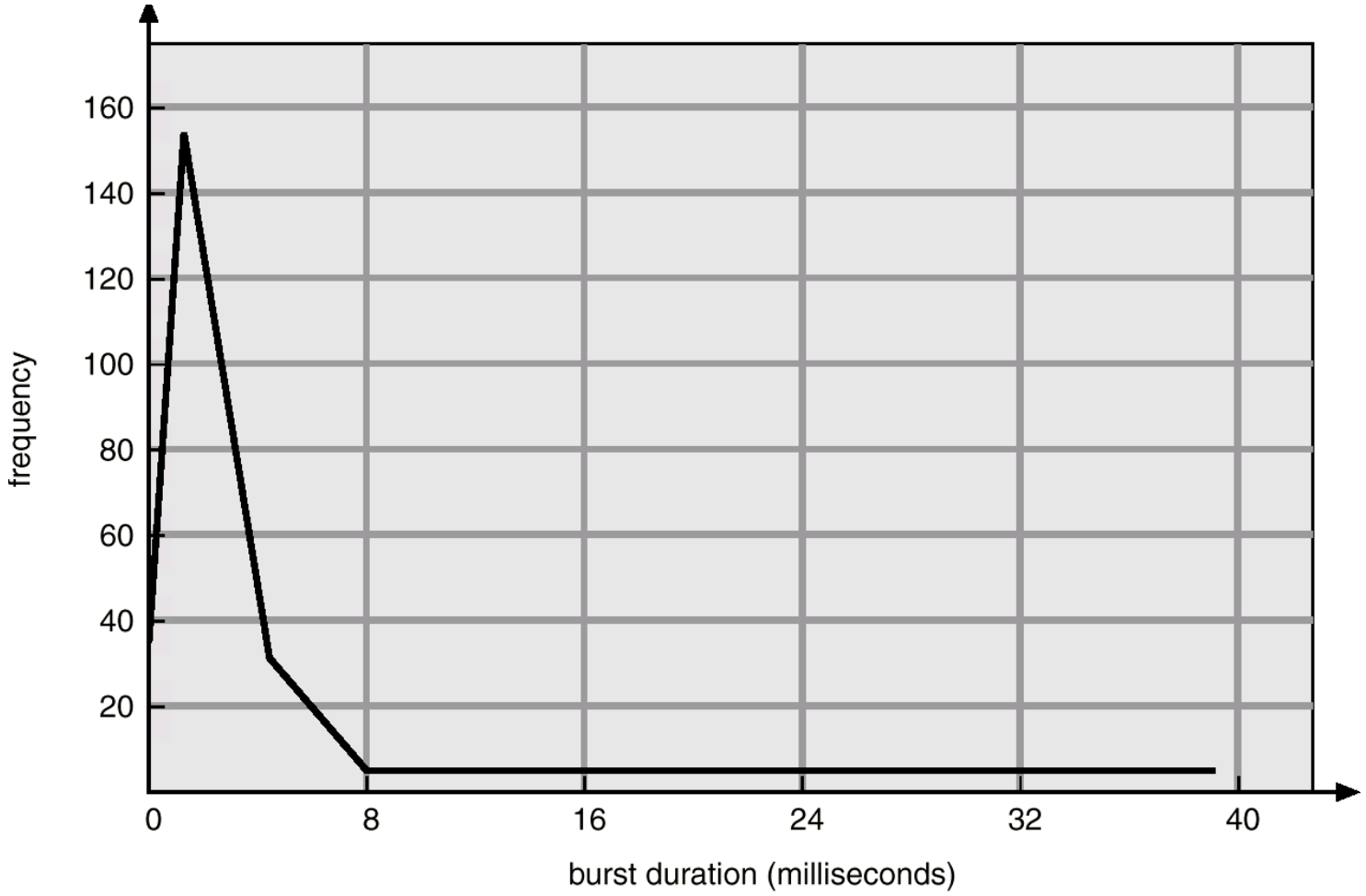


Değişen CPU ve G/Ç Çoğuşma (Burst) Sırası





CPU Çoğuşma Zamanı Histogramı



İş Planlayıcı- Scheduler

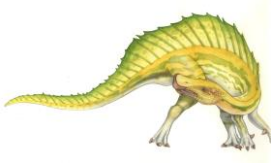
- Bellekte hazır konumdaki süreçler arasından işlemciyi kullanmak üzere seçer.
- İş planlama kararları bir sürecin :
  1. Çalışır durumdan bekler durumuna geçmesinde.
  2. Çalışır durumdan hazır duruma geçmesinde.
  3. Bekler durumdan hazır duruma geçmesinde.
  4. Çalışmayı bitirmesinde (termination) verilir.
- 1 ve 4 durumdaki görev yönetim algoritmaları *kesmeyen* (*nonpreemptive*).
- Diğer tüm algoritmalar *kesen* (*preemptive*) dir.

Görev Dağıtıcısı - Dispatcher

- Görev dağıtıcısı modülü kısa zamanlı görev yöneticisi tarafından seçilen sürece işlemcinin kontrolünü verir; bu işlem şunları içerir:
  - Bağlam anahtarlama (context switching)
  - Kullanıcı moduna anahtarlama
  - Kullanıcı programının uygun yerine sıçrayıp programın çalışmasına yeniden başlama
- *Dağıtım gecikmesi (dispatch latency)*– bir programdan bir başka programın çalıştırılmasına geçilmesi sırasında geçen zaman

İş Planlama Kriterleri

- İşlemci Kullanımı (CPU utilization) – işlemciyi mümkün olduğu kadar meşgul tut
- Çıkan iş oranı (Throughput) – Birim zamanda işini bitiren süreç sayısı
- Yanıt / Tamamlanma süresi (Turnaround time) – Belli bir sürecin işini bitirmesi için geçen süre. Sürecin belleğe alınması, hazır kuyruğunda bekleme süresi, CPU'da işlenmesi ve G/Ç yapma sürelerinin toplamıdır.
- Bekleme süresi (Waiting time) – Bir sürecin hazır kuyruğunda beklediği toplam süre
- Tepki süresi (Response time) – bir isteğin girilmesinden ilk tepkinin –çıktı değil- verilmesine kadar geçen süre (zaman-paylaşımlı ortam için)



Optimizasyon Kriterleri

- Maksimum CPU kullanımı
- Maksimum throughput
- Minimum yanıt süresi
- Minimum bekleme süresi
- Minimum tepki süresi

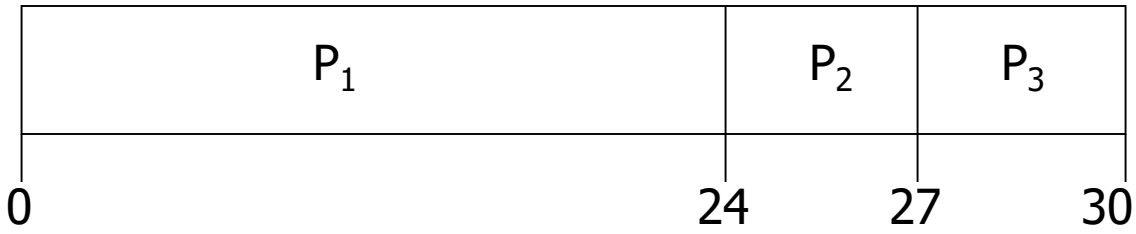
İş Planlama Algoritmaları

İlk Gelen, İlk Hizmet Alır/First Come First Served (FCFS) İş Planlama

- Örnek:

süreç	Çoğuşma Süresi
P_1	24
P_2	3
P_3	3

- süreçlerin t=0 da şu sırayla vardığı varsayılsın: P_1 , P_2 , P_3
  Yönetimin Gantt Chart'ı şöyle olur:



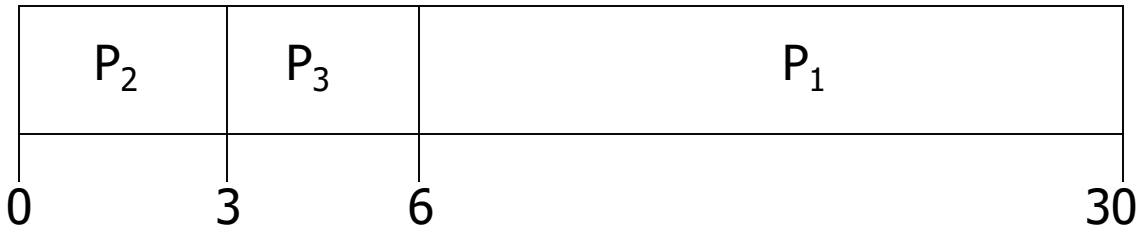
Bekleme süreleri P_1 = 0; P_2 = 24; P_3 = 27

- Ortalama bekleme süresi: (0 + 24 + 27)/3 = 17

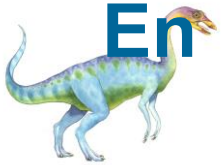
FCFS Yönetimi (Devam)

Süreçlerin P_2 , P_3 , P_1 sırasıyla vardığını varsayalım.

- Yönetimin Gantt Chart'ı şu şekilde olur:

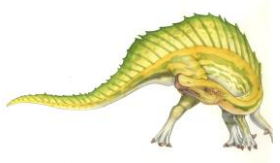


- Bekleme Süresi $P_1 = 6; P_2 = 0; P_3 = 3$
- Ortalama Bekleme Süresi: (6 + 0 + 3)/3 = 3
- Önceki durumdan çok daha iyi.
- *Konvoy etkisi* - uzun süreçlerin arkasında kısa süreçler



En Kısa Görev İlk / Shortest-Job-First (SJF) İş Planlama

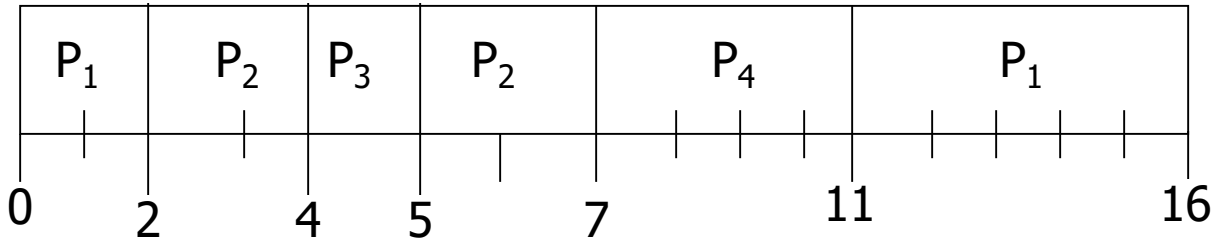
- Her süreci bir sonraki işlemci çoğuşma süresi ile ilişkilendir. Bu uzunlukları en kısa zamanlı süreci görevlendirmek için kullan.
- İki şema:
  - Kesmeyen – bir defa bir sürece işlemci kullanımı verilirse çoğuşma bitene kadar süreç kesilmez.
  - Kesen – eğer mevcut sürecin kalan zamanından daha kısa süreli yeni bir süreç gelirse mevcut işlemi kes. Bu şema En Kısa Kalan Zaman İlk/Shortest-Remaining-Time-First (SRTF) olarak da bilinir.
- SJF en iyidir – verilen bir dizi süreç için en kısa ortalama bekleme süresini verir.



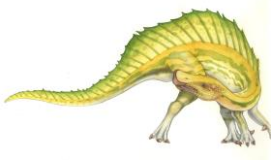
Kesen SJF Örneği

Süreç	Varış Zamanı	Çoğuşma Süresi
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (kesen)



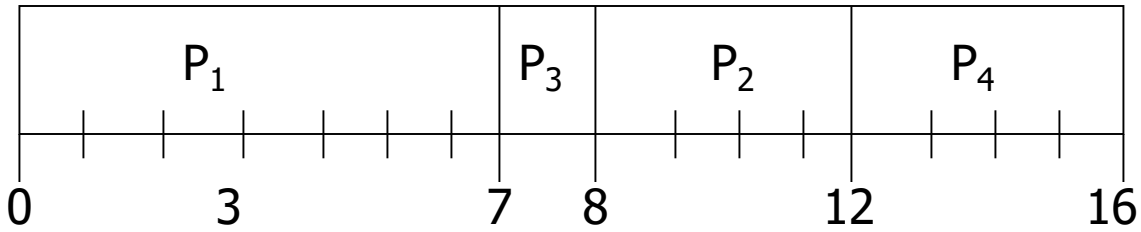
- Ortalama bekleme süresi = (9 + 1 + 0 +2)/4 = 3



Kesmeyen SJF Örneği

süreç	Varış Zamanı	Çoğuşma Süresi
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (kesmeyen)



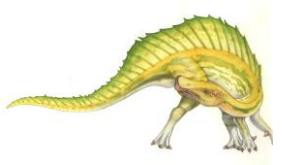
- Ortalama bekleme süresi= (0 + 6 + 3 + 7)/4 = 4

Bir sonraki CPU Çoğuşma Süresinin Belirlenmesi

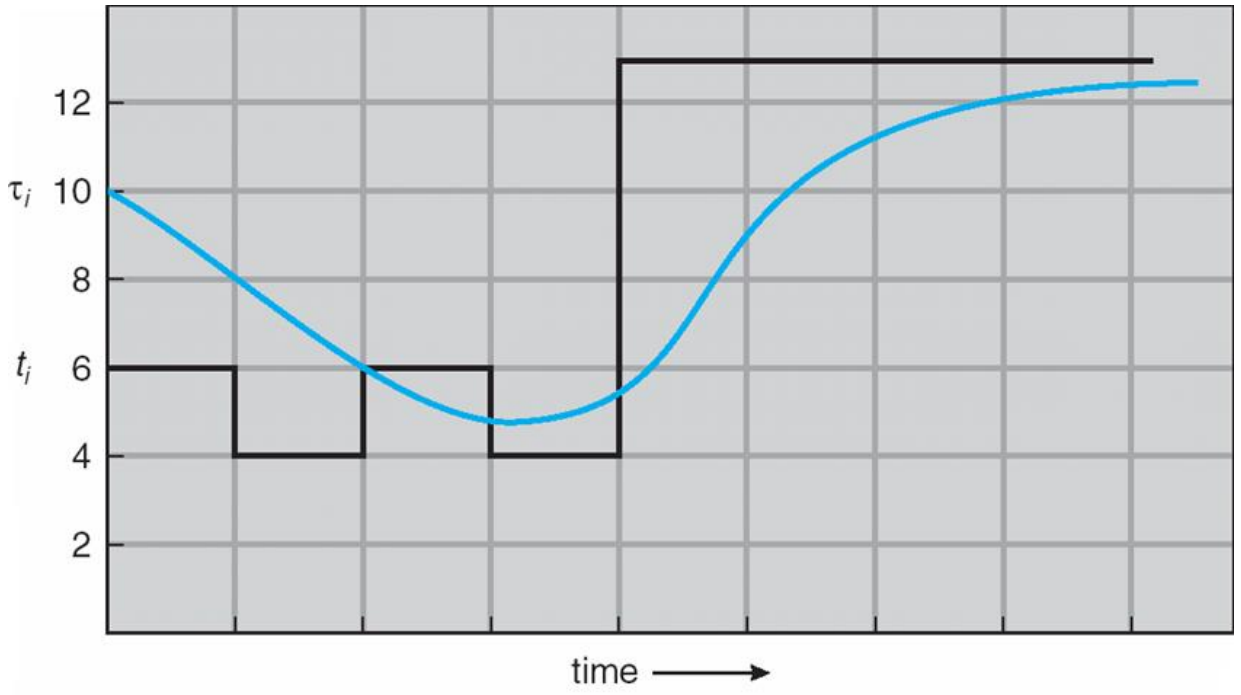
- Sadece uzunluğu tahmin eder
- Önceki CPU çoğuşmaları kullanılarak *üstel ortalama (exponential averaging)* yöntemi ile hesaplanır.

1. t_n = n. CPU çoğuşmasının gerçek uzunluğu
2. τ_{n+1} = bir sonraki CPU çoğuşmasının tahmini uzunluğu
3. $\alpha, 0 \le \alpha \le 1$

Formül: $$\tau_{n+1} = \alpha\, t_n + (1-\alpha)\tau_n.$$



Bir sonraki CPU Çoğuşma Süresinin Belirlenmesi



$\alpha = 1/2$

$\tau_{0=}10$

CPU burst (t_i)		6	4	6	4	13	13	13	...
"guess" (τ_i)	10	8	6	6	5	9	11	12	...

Üstel Ortalama Örnekleri

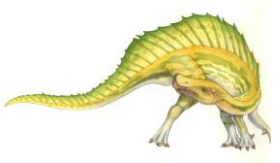
- α =0
  - $\tau_{n+1} = \tau_n$
  - Yakın geçmiş hesaba katılmaz.
- α =1
  - $\tau_{n+1} = t_n$
  - Sadece geçmiş gerçek CPU çoğuşmaları hesaba katılır.
- Formülü genişletirsek, şunu elde ederiz:

$$\tau_{n+1} = \alpha\, t_n + (1 - \alpha)\, \alpha\, t_{n-1} + \ldots$$
$$+ (1 - \alpha)^j\, \alpha\, t_{n-1} + \ldots$$
$$+ (1 - \alpha)^{n+1}\, t_n\, \tau_0$$

- Hem α hem de $(1 - \alpha)$ 1'den küçük eşit olduğu için, her ardışık terim öncekinden daha az ağırlığa sahiptir.

Öncelik Tabanlı İş Planlama

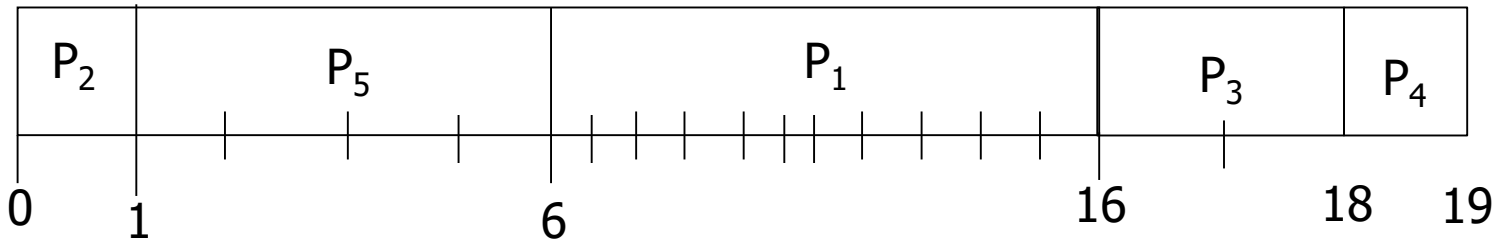
- Her süreç ile bir öncelik numarası (tamsayı) eşleştirilir
- CPU en yüksek önceliğe sahip sürece tahsis edilir (en küçük tamsayı ≡ en yüksek öncelik).
  - Kesen / Preemptive
  - Kesmeyen / Non-preemptive
- SJF bir sonraki tahmini CPU çoğuşma süresinin öncelik olarak kullanıldığı bir öncelik tabanlı algoritmadır.
- Problem ≡ Açlıktan ölüm/Starvation – düşük öncelikli süreçler hiçbir zaman işletilmeyebilir.
- Çözüm ≡ Yaşlandırma – zaman geçtikte sürecin önceliğini artır.



Öncelik Tabanlı İş Planlama Örneği

Süreç	Çoğuşma Süresi	Öncelik
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

- SJF (kesen)



- Ortalama bekleme süresi = (0 + 1 + 6 + 16 + 18)/5 = 8.2

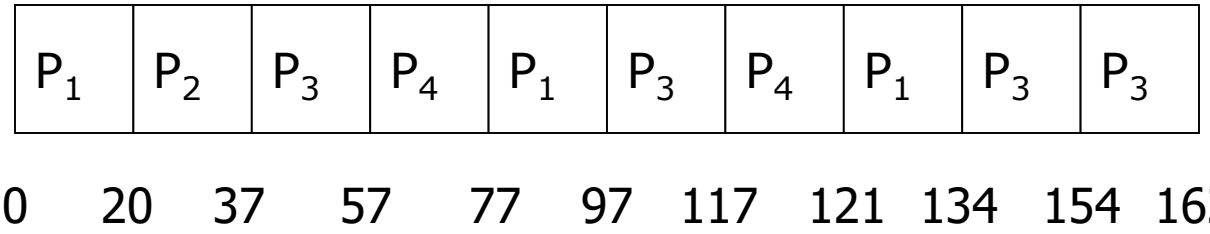
Zaman Dilimli / Round Robin (RR) İş Planlama

- Her süreç küçük bir CPU zaman dilimi (*time quantum*) alır, bu genelde 10-100 milisaniyedir. Bu süre geçtiğinde, süreç kesilir ve hazır kuyruğunun sonuna atılır.
- Eğer hazır kuyruğunda *n* süreç varsa ve time quantum değeri *q* ise, o zaman her süreç CPU zamanının 1/*n* ini bir defada en fazla *q* birimlik öbekler halinde alır. Hiçbir süreç (n-1)*q* zaman biriminden daha fazla beklemez.
- Performans
  - *q* büyük $\Rightarrow$ FCFS (İlk gelen ilk hizmet alır)
  - *q* küçük $\Rightarrow$ *q* bağlam anahtarlama süresine nazaran daha büyük olmalıdır, aksi halde çok fazla *zaman kaybı / overhead* olur.

Örnek: q= 20 ile RR

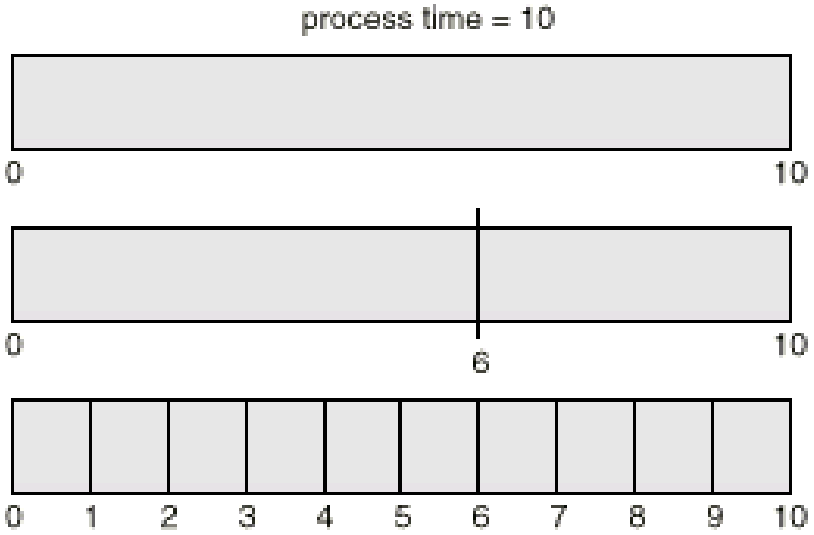
Süreç	Çoğuşma Süresi
P_1	53
P_2	17
P_3	68
P_4	24

- Gantt chart:



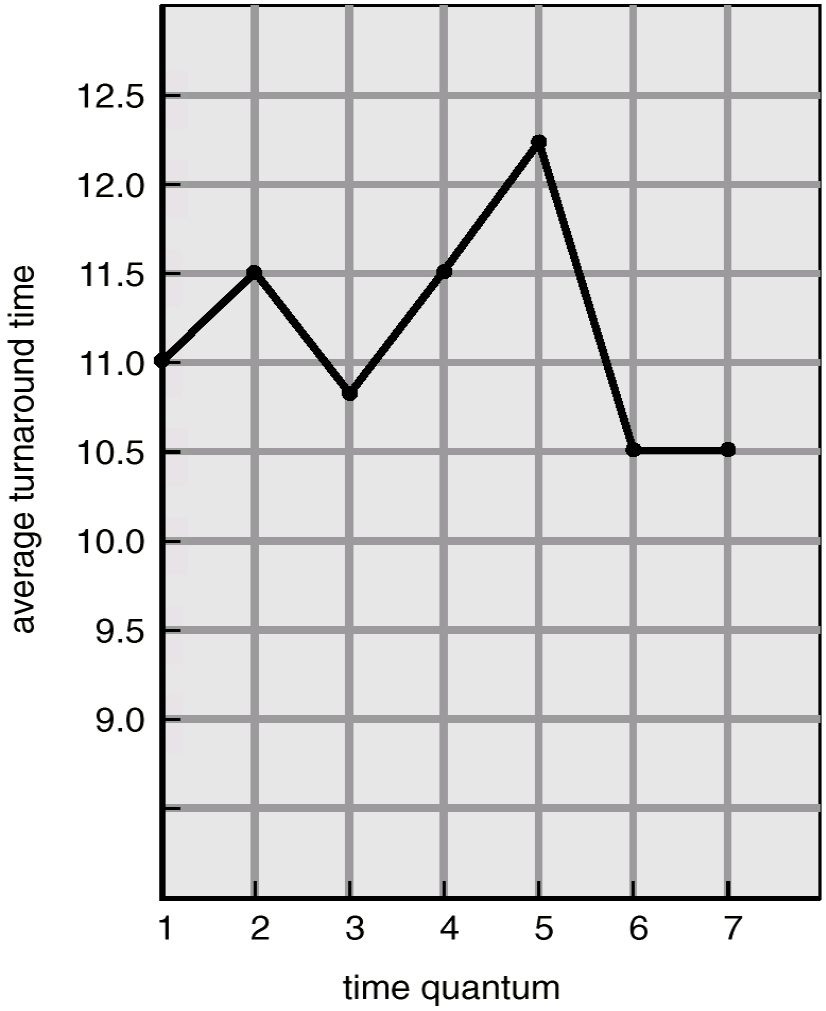
- Tipik olarak, SJF'den daha yüksek ortalama *yanıt süresine* ancak daha iyi *tepki süresine* sahiptir.

Daha Küçük Zaman Dilimi Kavram Değişikliğini Nasıl Artırır

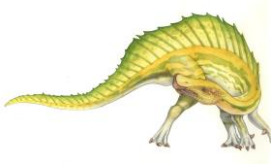


quantum	context switches
12	0
6	1
1	9

Zaman Dilimi ile Ortalama Yanıt Süresinin Değişimi



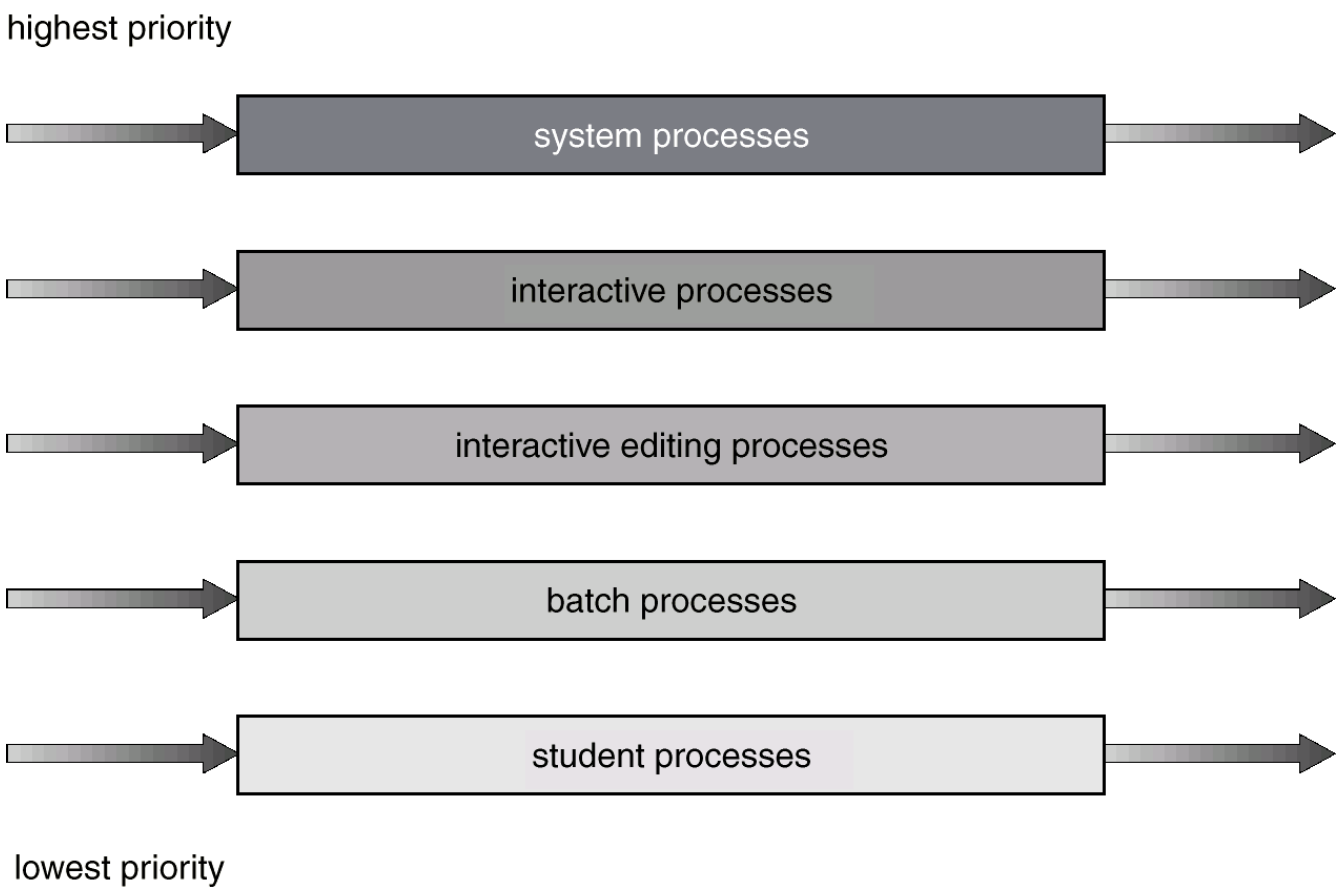
process	time
P_1	6
P_2	3
P_3	1
P_4	7

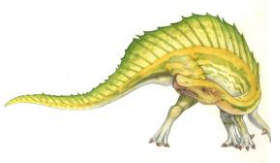


Çok-Katmanlı Kuyruk Multilevel Queue

- Hazır kuyruğu çeşitli kuyruklara ayrılmıştır:
  önplan - foreground (etkileşimli - interactive)
  arkaplan - background (yığın - batch)
- Her kuyruğun kendi görev yönetim algoritması vardır,
  önplan – RR
  arkaplan – FCFS
- Kuyruklar arasında görev yönetimi yapılmalıdır.
  - Sabit öncelikli görev planlama; mesela, önce bütün önplan sonra arkaplan süreçleri işlet. Starvation ihtimali.
  - Zaman dilimi – her kuyruk kendi içindeki süreçlere bölüştüreceği belli bir zaman dilimi alır; mesela,
    - **%80 RR ile yönetilen önplan süreçler için**
    - **%20 FCFS ile yönetilen arkaplan süreçler için**

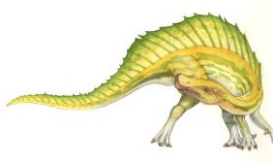
Çok-Katmanlı Kuyruk İş Planlama



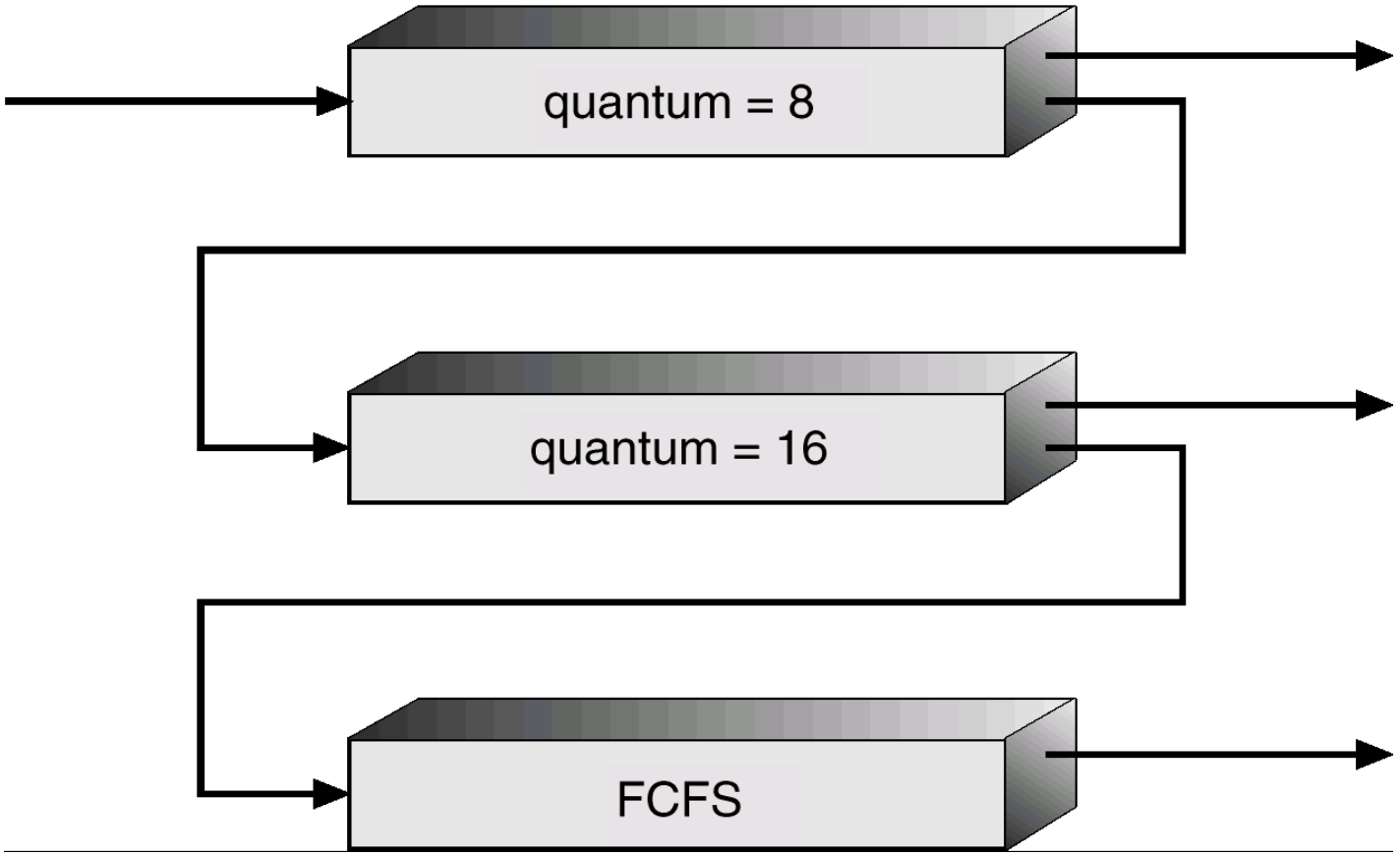


Çok-Katmanlı Geribildirim Kuyruğu
Multilevel Feedback Queue

- Bir süreç zaman içinde farklı kuyruklara girebilir; bu yolla yaşlandırma uygulanabilir.
- Çok-katmanlı kuyruk görev yöneticisi aşağıdaki parametreler ile tanımlanır:
  - Kuyruk sayısı
  - Her kuyruk için görev yönetim algoritmaları
  - Bir sürecin ne zaman terfi ettirileceği ile ilgili yöntem
  - Bir sürecin ne zaman tenzil ettirileceği ile ilgili yöntem
  - Bir sürecin ilk olarak hangi kuyruğa gireceği ile ilgili yöntem



Çok-Katmanlı Geribildirim Kuyrukları



Çok-Katmanlı Geribildirim Kuyruklarına Örnek

- Üç kuyruk
  - Q_0 – time quantum 8 milisaniye
  - Q_1 – time quantum 16 milisaniye
  - Q_2 – FCFS
- İş Planlama
  - Yeni bir görev FCFS ile çalışan Q_0 ya giriş yapar. İşlemciyi alınca 8 milisaniye çalışır. Eğer 8 milisaniyede işi bitmezseQ_1' e atılır.
  - Yine FCFS ile çalışan Q_1'de sürece ek 16 milisaniye süre verilir. Eğer bitmezde, süreç kesilir veQ_2 ye atılır.