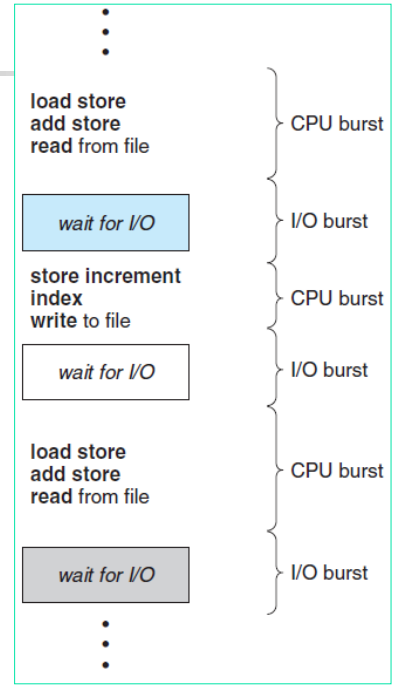
Temel kavramlar

- **CPU scheduling (planlama), multiprogramming çalışan işletim sistemlerinin temelini oluşturur.**
- CPU, process'ler arasında geçiş yaparak bilgisayarı daha verimli hale getirir.
- **Her zaman aralığında bir process'in çalıştırılması amaçlanır.**
- Tek işlemcili sistemlerde, bir anda sadece bir process çalıştırılabilir.
- CPU, process'lerde ortaya çıkacak bekleme durumlarında başka process'leri çalıştırır.
- Hafızada çok sayıda process bulundurulur.
- Bir process herhangi bir şekilde beklemeye geçtiğinde CPU başka bir process'e geçiş yapar.
- Bilgisayardaki **tüm kaynaklar** kullanılmadan önce zamana göre **planlanır**.

Temel kavramlar

CPU Burst Cycle ve I/O Burst Cycle

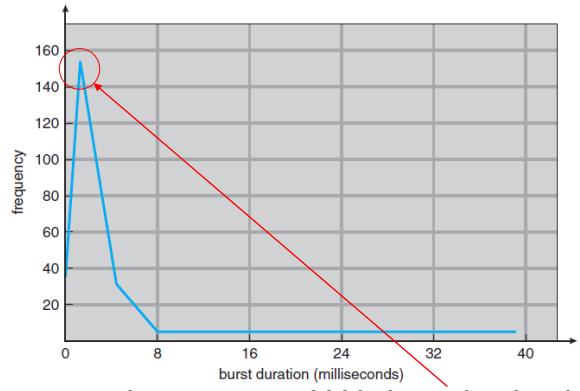
- Process çalıştırma, **CPU execution** ve **I/O wait döngüsünü içermektedir.**
- Process'ler bu iki durum arasında geçiş yaparlar.
- Process'ler **çalışmaya CPU burst ile başlarlar** ve I/O burst ile devam ederler.



Temel kavramlar

CPU Burst Cycle ve I/O Burst Cycle

- CPU burst süresi, process'ten process'e ve bilgisayardan bilgisayara çok farklı olabilmektedir.



- **Process'ler için CPU burst süresi sıklıkla kısa olmaktadır.**
- Process'ler kısa aralıklarla durumunu değiştirmektedir.

Temel kavramlar

CPU Scheduler

- **CPU bekleme durumuna geçtiğinde**, işletim sistemi hazır kuyruğundan (ready queue) bir process'i çalıştırılmak üzere seçmek zorundadır.
- Bu seçme işlemi kısa dönem planlayıcı (short-term scheduler veya CPU scheduler) tarafından gerçekleştirilir.
- Hazır kuyruğu, ilk gelen ilk çıkar (firs-in-first-out, FIFO) olmak zorunda değildir.
- Hazır kuyruğu, **FIFO**, **priority queue**, **ağaç**, **sırasız bağlı liste** şeklinde oluşturulabilir.
- Hazır kuyruğunda bekleyen tüm process'lerin CPU tarafından çalıştırılmak üzere seçilme olasılıkları vardır.
- Kuyruk içindeki kayıtlarda, **process control block (PCB) tutulur.**

Temel kavramlar

Preemptive Scheduling

- CPU-scheduling kararı 4 durum altında gerçekleştirilir:
  1. Bir process **çalışma** durumundan **bekleme** durumuna geçtiğinde (I/O isteği),
  2. Bir process **çalışma** durumundan **hazır** durumuna geçtiğinde (interrupt),
  3. Bir process **bekleme** durumundan **hazır** durumuna geçtiğinde (I/O tamamlanması),
  4. Bir process'in **sonlandırıldığında**.
- Eğer scheduling işlemi **1. ve 4. durumlarda gerçekleşmişse**, buna nonpreemptive veya cooperative scheduling denir.
- **2. ve 3. durumlarda gerçekleşmişse** preemptive scheduling denir.

Temel kavramlar

Preemptive Scheduling

- **Nonpreemptive scheduling'te**, CPU bir process'e tahsis edilmişse, bu process **sonlandırılıncaya kadar**, **CPU'yu serbest bırakıncaya kadar** veya **bekler durumuna geçinceye kadar tutar.**
- Windows 3.1, nonpreemptive scheduling kullanmıştır.
- Diğer tüm Windows versiyonları preemptive scheduling kullanmıştır.
- Mac OS X işletim sistemi de preemptive scheduling kullanmaktadır.
- **Preemptive scheduling veri paylaşımı yaptığında** race condition gerçekleşir.
- **Bir process kernel verisi üzerinde değişiklik yaparken** yarıda kesilerek **başka bir process'e geçilmesi** ve aynı veriye erişim yapılması **halinde çakışma** meydana gelir.

Temel kavramlar

Dispatcher

- CPU scheduling işlevini gerçekleştiren bileşen dispatcher olarak adlandırılır.
- **Dispatcher, short-term scheduler tarafından CPU'ya atanacak process'i seçer.**
- Dispatcher aşağıdaki işlevleri içermektedir:
  - Context geçişi
  - Kullanıcı moduna geçiş
  - Programı yeniden başlatmak için kullanıcı programında uygun konuma atlama
- Dispatcher'ın çok hızlı bir şekilde geçiş yapması zorunludur.
- Process'ler arasında **geçiş süresine** dispatch latency denilmektedir.

Konular

- Temel kavramlar
- Scheduling kriterleri
- Scheduling algoritmaları
- Çoklu process veya scheduling
- Gerçek zamanlı CPU scheduling

Scheduling kriterleri

- CPU scheduling algoritmaları çok sayıda farklı kritere göre karşılaştırılır:
  - CPU utilization: CPU'nun olabildiği kadar kullanımda olması istenir. CPU kullanım oranı %0 - %100 arasındadır. Gerçek sistemlerde bu oran %40 ile %90 arasındadır.
  - Throughput: Her zaman aralığında tamamlanan process sayısıdır.
  - Turnaround time: Bir process'in hafızaya alınmak için bekleme süresi, hazır kuyruğunda bekleme süresi, CPU'da çalıştırılması ve I/O işlemi yapması için geçen sürelerin toplamıdır.
  - Waiting time: Bir process'in hazır kuyruğunda beklediği süredir.
  - Response time: Bir process'e gönderilen isteğe cevap dönünceye kadar geçen süredir.
- **CPU utilization'ı** ve **thoroughput'u maksimum**, **turnaround time**, **waiting time** ve **response time'ı minimum** yapmak amaçlanır.
- Genellikle ortalama değerler optimize edilmeye çalışılır.

Konular

- Temel kavramlar
- Scheduling kriterleri
- Scheduling algoritmaları
- Çoklu process veya scheduling
- Gerçek zamanlı CPU scheduling

Scheduling algoritmaları

- CPU scheduling algoritmaları, **hazır kuyruğunda bekleyen process'lerden hangisinin CPU'ya atanacağını belirlerler.**
  - First-Come, First-Served Scheduling
  - Shortest-Job-First Scheduling
  - Priority Scheduling
  - Round-Robin Scheduling

Scheduling algoritmaları

First-Come, First-Served Scheduling

- En basit CPU scheduling algoritmasıdır ve first-come first served (FCFS) şeklinde çalışır.
- CPU'ya ilk istek yapan process, CPU'ya ilk atanan process'tir.
- **FIFO kuyruk yapısıyla yönetilebilir.**
- FCFS algoritmasıyla **ortalama bekleme süresi** genellikle **yüksektir**.
- **Bekleme süreleri** process'lerin **kuyruğa geliş sırasına göre çok değişmektedir.**

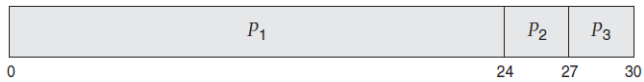
Scheduling algoritmaları

First-Come, First-Served Scheduling

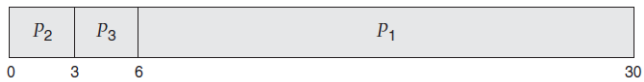
- Aşağıdaki 3 process için CPU'da çalışma süreleri ms olarak verilmiştir.

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Process'ler **P1, P2, P3** sırasıyla gelirse Gantt şeması aşağıdaki gibidir.



- Ortalama bekleme süresi **(0+ 24 + 27) / 3 = 17 ms** olur.
- **P2, P1, P3** sırasıyla gelirse Gantt şeması aşağıdaki gibidir.



- Ortalama bekleme süresi **(0+ 3 + 6) / 3 = 3 ms** olur.

Scheduling algoritmaları

First-Come, First-Served Scheduling

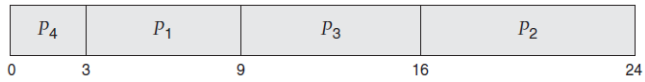
- FCFS algoritmasında, **process'lerin çalışma süreleri çok farklıysa** ortalama **bekleme süreleri çok değişken olur.**
- **Bir CPU-bound process ile çok sayıda I/O bound process varsa**, CPU-bound process CPU'da çalışırken **tüm I/O bound process'ler hazır kuyruğunda bekler**, **I/O cihazları boş kalır.**
- Çok sayıda küçük process'in **büyük bir process'in CPU'yu terketmesini beklemesine** convoy effect denilmektedir.
- **Bir process'e CPU tahsis edildiğinde sonlanana veya I/O isteği yapana kadar CPU'yu elinde tutar.**
- FCFS algoritması belirli zaman aralıklarıyla **CPU'yu paylaşan time-sharing sistemler için uygun değildir.**

Scheduling algoritmaları

Shortest-Job-First Scheduling

- Shortest-Job-First Scheduling (SJF) algoritmasında, CPU'ya bir sonraki **işlem süresi en kısa olan** (shortest-next-CPU-burst) process **atanır.**

Process	Burst Time
P_1	6
P_2	8
P_3	7
P_4	3



- Ortalama bekleme süresi, **(0 + 3 + 9 + 16) / 4 = 7 ms'**dir. **FCFS kullanılsaydı 10,25 ms olurdu** ((0 + 6 + 14 + 21) / 4)**.**
- **SJF algoritması minimum ortalama bekleme süresini elde eder.**

Scheduling algoritmaları

Shortest-Job-First Scheduling

- SJF algoritmasındaki **en büyük zorluk, sonraki çalışma süresini tahmin etmektir.**
- Long-term (job) scheduling için kullanıcının belirlediği süre alınabilir.
- **SJF algoritması genellikle long-term scheduling için kullanılır.**
- SJF algoritması **short-term scheduling seviyesinde kullanılamaz.**
- Short-term scheduling'te **CPU'da sonraki çalışma süresini bilmek mümkün değildir.**
- Short-term scheduling'te **sonraki çalışma süresi tahmin edilmeye çalışılır.**
- **Sonraki çalışma süresinin önceki çalışma süresine benzer olacağı beklenir.**

Scheduling algoritmaları

Priority Scheduling

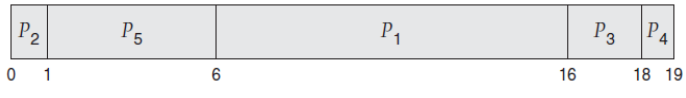
- Shortest-job-first (SJF) algoritması, priority scheduling algoritmalarının özel bir durumudur.
- **CPU en yüksek önceliğe sahip process'e atanır.**
- Eşit önceliğe sahip olanlar ise FCFS sırasıyla atanır.
- **SJF algoritması tahmin edilen CPU-burst süresine göre önceliklendirme yapar.**
- SJF algoritmasında, CPU burst süresi azaldıkça öncelik artar, CPU burst süresi arttıkça öncelik azalır.

Scheduling algoritmaları

Priority Scheduling

- Aşağıda 5 process için öncelik değerine göre gantt şeması verilmiştir.

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2



- Ortalama bekleme süresi **(1+ 6 + 16 + 18) / 4 = 8,2 ms** olur.

Scheduling algoritmaları

Priority Scheduling

- Önceliklendirme kriterleri aşağıdakilerden bir veya birkaç tanesi olabilir:
  - Zaman sınırı
  - Hafıza gereksinimi
  - Açılan dosya sayısı
  - I/O burst ve CPU burst oranı
  - Process'in önemi
- Priority scheduling preemptive veya nonpreemptive olabilir.
- **Preemptive yönteminde**, bir process hazır kuyruğuna geldiğinde, çalışmakta olan process'ten daha öncelikli ise, **çalışmakta olan kesilir.**
- **Nonpreemptive yönteminde**, bir process hazır kuyruğuna geldiğinde, çalışmakta olan process'ten daha öncelikli bile olsa, **çalışmakta olan durum değiştirene kadar devam eder.**

Scheduling algoritmaları

Priority Scheduling

- Priority scheduling algoritmasında, **CPU sürekli yüksek öncelikli process'leri çalıştırabilir** ve bazı processler sürekli hazır kuyruğunda bekleyebilir (indefinite blocking, starvation).
- **Sınırsız beklemeyi engellemek için düşük öncelikli process'ler kuyrukta beklerken öncelik seviyesi artırılır** (Örn. her 15 dakikada 1 artırılır).
- Öncelik değeri artırılarak **en düşük önceliğe sahip process'in** bile belirli **bir süre sonunda çalışması sağlanır.**

Scheduling algoritmaları

Round-Robin Scheduling

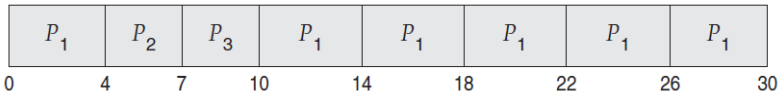
- Round-robin (RR) scheduling, genellikle time-sharing sistemlerde kullanılır.
- Hazır kuyruğundaki **process'ler belirli bir zaman aralığında** (time slice) **CPU'ya sıralı atanır.**
- Zaman aralığı genellikle 10 ms ile 100 ms aralığında seçilir.
- Time slice aralığından daha kısa sürede sonlanan process CPU'yu serbest bırakır.
- Round-robin scheduling ile **ortalama bekleme süresi genellikle uzundur.**

Scheduling algoritmaları

Round-Robin Scheduling

- Aşağıda 3 process için CPU-burst time ve gantt şeması verilmiştir.
- Örnekte **time slice = 4 ms** olarak alınmıştır.

Process	Burst Time
P_1	24
P_2	3
P_3	3



- P_1 için = 10-4 = **6**, P_2 için **4**, P_3 için **7 ms bekleme süresi vardır.**
- **Ortalama bekleme süresi ise** 17 / 3 = **5,66 ms'dir.**
- **q time slice** süresiyle **n process** çalışan sistemde, bir process için **en fazla bekleme süresi** $(n - 1) * q$ olur.

Scheduling algoritmaları

Round-Robin Scheduling

- **Time slice** süresi **çok büyük olursa** çalışma **FCFS yöntemine benzer.**
- Time slice süresi **çok küçük olursa context switch işlemi çok fazla yapılır.**
- Context switch süresi overhead olur ve çok fazla context switch yapılması istenmez.
- **Time slice süresi, context switch süresinin genellikle 10 katı alınır.**
- CPU'nun %10 süresi context switch için harcanır.