Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

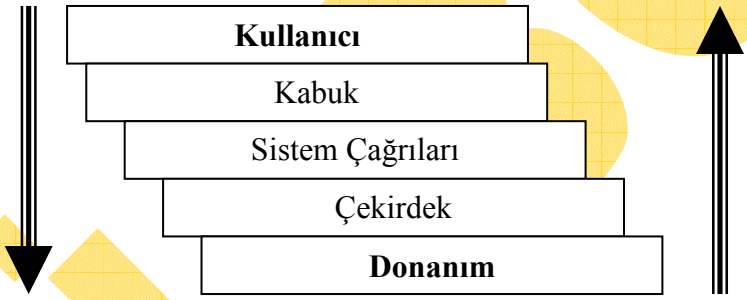
İşletim Sistemleri

Modern bilgisayar sistemleri Von Neumann ilke ve bileşenlerine sahiptir. Von Neumann bileşenlerine sahip bilgisayar sistemlerinde merkezi işlem birimi, ana bellek ve giriş/çıkış birimlerinden oluşmaktadır. İşletim sistemlerinin temel amacı bu kaynakların en verimli ve etkin biçimde kullanılmalarını sağlamaktır.

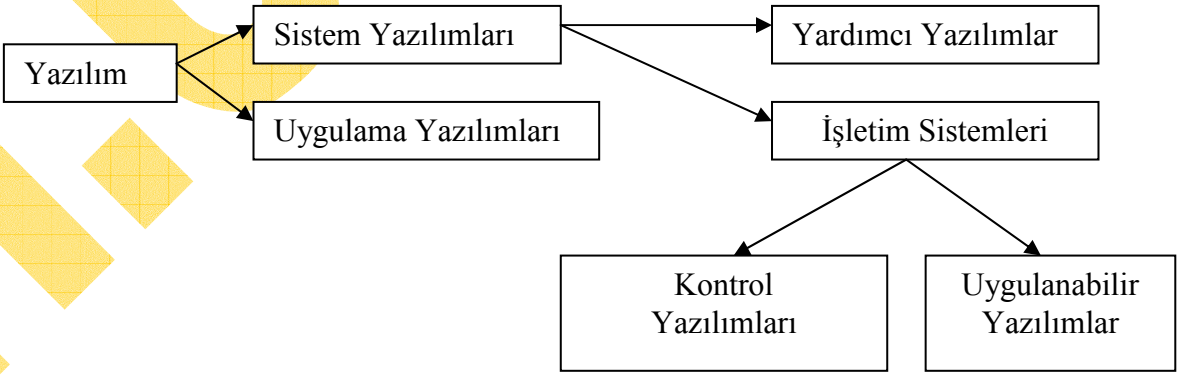
İşletim sistemlerini de daha önce; tek kullanıcılı (single user), çok kullanıcılı (multi user), tek işlemli (single tasking) ve çok işlemli (multi asking) olarak sınıflamıştık. Tek kullanıcılı ve tek işlemli yapılarda işletim sisteminin görevleri daha basit bir düzenekte gerçekleşmektedir. Ancak çok işlemli ya da çok kullanıcılı yapılarda işletim sistemi daha karmaşık bir organizasyona sahiptir.

Bu yaklaşımla; işletim sistemlerinin 3 katmanı ve işletim sistemlerinin temel olarak 4 işlevi vardır.
İşletim sistemlerinin katmanları sırasıyla;

1. Kabuk (Shell)
2. Sistem Çağrıları (System Calls)
3. Çekirdek (Kernel)



Bazı kaynaklar ise farklı bir sınıflama yapmaktadırlar:



Kabuk: İşletim sisteminin kullanıcıya gözüken yüzüdür. Bir çok kaynakta kabuk, komut yorumlayıcısı olarakta adlandırılmaktadır. Bir bakıma kullanıcı ile çekirdek arasındaki arabirim olarak da tanımlamak olanaklıdır.

Bilgisayar açıldıktan bir süre sonra komut satırı (prompt) görüntülenir. Kullanıcı tarafından komut satırına girilen komutlar bilgisayar tarafından işleme konulur. İşte

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

bu nokta da kabuk (shell) olarak adlandırılan yazılım devreye girer. Öncelikle komutun geçerliliğini inceler, kullanıcının ne yapmak istediğini çözümler ve bu iş için gerekli olan programları belleğe yükler.

MS-DOS işletim sisteminde kabuk olarak command.com kullanılır. Dos işletim sisteminin aksine Unix'in kaynak kodları açık olduğundan dolayı ve Unix'in C gibi yüksek seviyeli bir programlama dili ile yazılmış olması nedeniyle Unix'te geliştirilmiş bir çok kabuk programı vardır.

Shell	**Açılımı**	**Açıklama**
sh	Bourne Shell	Steve Bourne tarafından geliştirilen orijinal Unix Shell
csh	C-Shell	Berkeley Ünivesitesinde C dili kullanılarak yazılmıştır.
ksh	Korn Shell	David Korn tarafından geliştirilmiştir. sh ve tcsh'nin gelişmiş bütün özeliklerine sahiptir. En etkili kabuk olarak bilinir.
bash	Bourne Again Shell	Free Software Foundation tarafından geliştirilmiştir. Bourne benzeri bir script dili ile yazılmıştır ve tcsh ve ksh'ın bütün özelliklerine sahiptir.
tcsh	T-Shell	Geliştirilmiş C-Shell olarak bilinir.
zsh	Z-Shell	Bash, ksh ve tcsh ile benzerlik gösterir.

Sistem Çağrıları:

Çekirdek:

MS-DOS'ta sistem dosyaları olarak bilinen msdos.sys ve io.sys dosyaları, bu işletim sisteminin çekirdek dosyalarını oluşturmaktadır. Unix'te ise vmunix, unix gibi isimler almaktadır.

Çekirdek, bilgisayar açıldığı zaman belleğe yüklenir. Kabuk, kullanıcının girdiği komutu yorumladıktan sonra çekirdeğe bildirir. Dolayısı ile kabuk ile çekirdek birlikte çalışır. Örneğin; *rm silbeni* komutunu kullanıcı girmiş olsun. Komut ile istenen, silbeni dosyasının sistemden silinmesidir. Bu durumda kabuk *rm* komutunu içeren ilgili dosyayı araştırır ve bulunca da bu programı silbeni dosyası üzerinde çalıştırmak için çekirdeğe gereksinim duyar. *rm* komutunu içeren dosyanın çalışması bitince kabuk'a dönülür ve kabuk yeni bir komut beklemek için komut satırını görüntüler.

Çekirdek, bilgisayarın donanımıyla doğrudan etkileşen işletim sisteminin bir parçasıdır. Programlar tarafından kullanılan bir hizmet sağlar. En önemli fonksiyonları:

- Belleği yönetmek
- Bilgisayara ulaşımları kontrol etmek
- Dosya sistemini oluşturup, korumak
- İnterruptları kullanmak
- Hataları kontrol etmek
- Girdi-çıktı birimlerini çalıştırmak

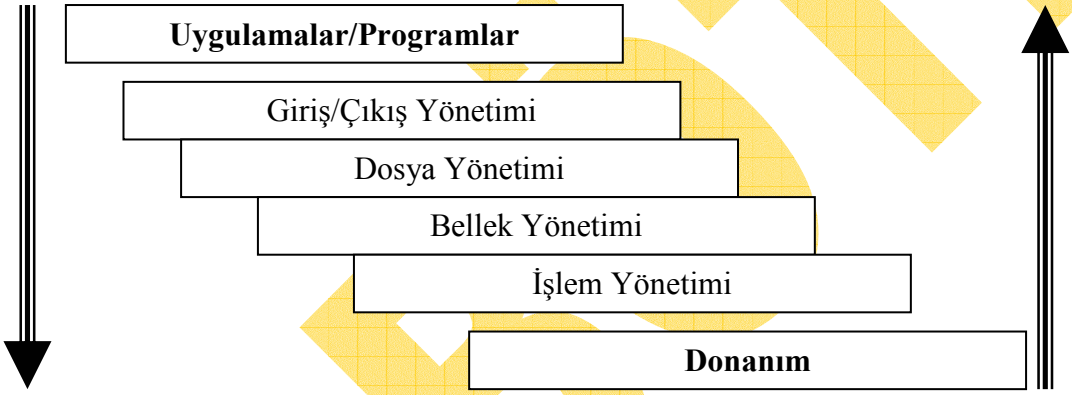
Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

- Bilgisayarın kaynaklarını (işlemci, girdi-çıktı birimleri gibi...) kullanıcılar arasında dağıtmak.

Bir çok kaynakta işletim sisteminin çekirdek olduğu ileri sürülmektedir. Bunun nedeni, işlevselliği bakımından en önemli işletim sistemi öğesi olmasıdır.

İşletim Sisteminin İşlevleri:

1) İşlem Yönetimi (Process Management)
2) Bellek Yönetimi (Memory Management)
3) Dosya Yönetimi (File Management)
4) Giriş/Çıkış Yönetimi (I/O Management)



Burada konu ile ilgili kavramlarından bahsetmek gerekmektedir:
Program: Verileri işlemek üzere oluşturulan komutlar/komut kümeleridir.
İşlem: Bir programın bilgisayar sistemlerince işletimi esnasında aldığı isimdir.

I-) İşlem Yönetimi:

Veriler ve verileri işleyecek olan komutlar ikili sistemde bellekte bulunmaktadır. Bunlar sırasıyla bellekten merkezi işlem birimine alınarak işlemler (operation), veriler (operand) üzerine uygulanır ve işlem bittikten sonra anabellek, ikincil bellek ya da giriş çıkış yazmaçlarına gönderilirler.

Tek işlemli ve tek kullanıcılı sistemlerde bu işlemler bir çizelge ile basit bir şekilde gerçekleştirilebilmektedir. Ancak, çok işlemli (multi tasking) ve/veya çok kullanıcılı durumlarda işlem yönetimi daha karmaşık düzenekler ile sağlanmaktadır. Bunun nedeni, merkezi işlem birimi bir diğer ifade ile işlemci, belirli bir anda yalnızca bir "işlem" gerçekleştirilebilmektedir. Birden fazla işlem talebi olduğunda (özellikle çok işlemli işletim sistemlerinde) ise olası işlemler belli konumlara sokularak işlem sürecine katılırlar. Bunu işletim sisteminin işlem yöneticisi gerçekleştirir.

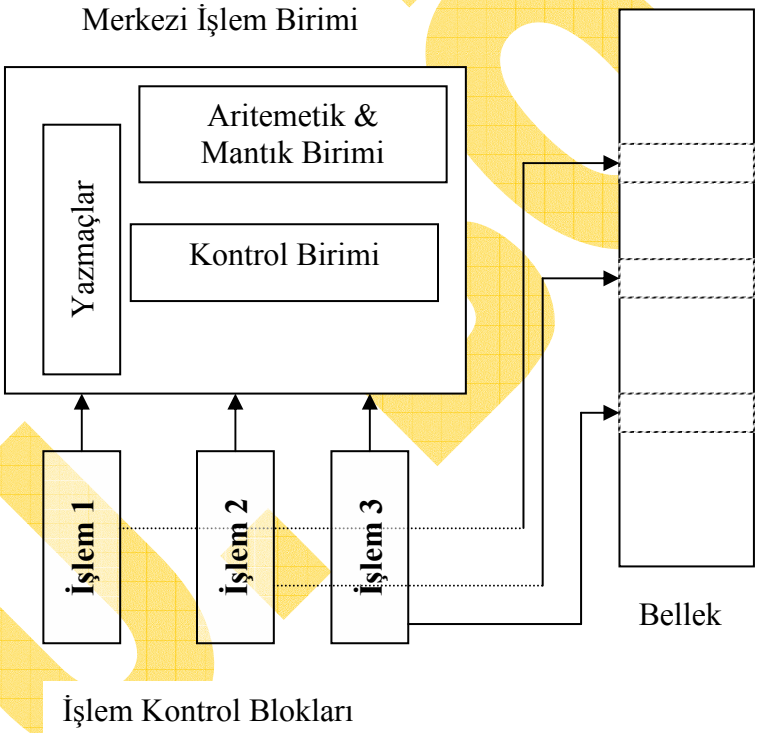
Merkezi işlem birimi, belirli bir anda yalnızca bir işlem gerçekleştirdiğini daha önce belirtmiştik. Çok işlemli (multi tasking) durumlarda birden fazla işlem olduğu için, merkezi işlem birimi bir işlemi gerçekleştirirken talep edilen diğer işlemler beklemeye alınır. Merkezi işlem birimi her işleme belli bir zaman dilimi ayırır ve sürekli olarak çalıştırdığı işlemi değiştirir. Bu süreç çok hızlı çalıştığı için sanki belirli bir an da çok işlem yapılıyormuş gibi görünmektedir.

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

İşletime alınan işlemlerin konumları aşağıdaki gibidir:

a) Hazır (ready)
b) Çalışıyor (running)
c) Bekliyor (suspend)

a) Hazır: Kullanıcı tarafından talep edilen işlem ana belleğe alınarak işleme alınması için hazır duruma getirilir. Aynı zamanda merkezi işlem birimini kullanabilir duruma gelen işlemler için kullanılır.
b) Çalışıyor: İşlemin merkezi işlem biriminde bulunma durumunu belirtir. Bir diğer ifade ile işletime alınan işlemler için kullanılır.
c) Bekliyor: İşlem, giriş/çıkış birimlerince veri transferi için bekliyor olması durumudur.



İşlem 1 bellekten alınarak merkezi işlem birimine sokulur. Ancak işlem2 ve/veya işlem 3 sürece sokulduğunda merkezi işlem birimini etkin kullanımı söz konusu olacaktır. İşlemlerin işletime alınması esnasındaki sürecin düzenlenmesi "işlem yönetimi" tarafından gerçekleştirilmektedir.

Bilindiği gibi, ana bellekten alınan ve işlemciye sokulan işleme ilişkin bilgiler öncelikle merkezi işlem biriminin yazmaçlarına yüklenir (örneğin şekildeki işlem 1). Ancak bir şekilde işlem 1, bekleme durumuna alınıp işlem iki çalışma durumuna getirilirken merkezi işlem birimi yazmaçlarında en son işlem1'e ilişkin bilgiler vardır. İşlem 1, merkezi işlem birimini boşaltırken, yazmaçlardaki bu bilgiler bir veri yapısı şeklinde tutulur. Bu yapıya işlem kontrol bloğu adı verilir. İşlem kontrol bloklarında aynı zamanda işlemin hangi komuttan devam edeceği, varsa kullanılan/açık dosya bilgileri vb. bilgileri tutar. Aynı zamanda her işleme bir kimlik numarası-İşlem Kimlik Numarası- verir (Process ID-PID).

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

İşlem Kontrol Bloğu	
İşlem Yönetim Bilgileri	İşlem Kimlik No
	Program Sayacı
	Yazmaç Bilgileri
	: : :
	Diğer Bilgiler
	Dosya Bilgileri
İşlenen Dosya Bilgileri	Açık Dosyalar
	: : :
	Diğer Bilgiler

İşlem-Durum Algoritmaları

İşlem durumları algoritmasından önce özellikle "çalışıyor" durumdaki (işletimdeki) işlemler için iki özellikten bahsetmek gerekir. Bunlar sırasıyla; müdahale edilebilir-müdahale edilemez özelliklerdir.

Müdahale Edilebilir: Merkezi işlem biriminde çalışır durumdaki işleme, işletim sistemi tarafından müdahale edilerek merkezi işlem birimini bırakmasıdır (bekleme durumuna geçmesidir).

Müdahale Edilemez: İşletim sistemi, merkezi işlem biriminde çalışıyor durumdaki işleme müdahale edemez. İşlem kendiliğinden sonlanıp merkezi işlem birimini bırakması gerekmektedir.

İşletim sistemi, hangi işlemin beklemeye alınacağını, hangisinin işletime alınacağına karar verirken bazı algoritmaları kullanır. Bu algoritmalar aşağıda verilmiştir.

1) FIFO/FCFS (First In First Out / First Come First Served):
"Hazır" durumda olan işlemler, hazır durumuna geliş sürelerine göre kendi aralarında bir sıra oluştururlar. İlk sıradaki işlem merkezi işlem birimine alınır. Bu algoritmada işletim sistemi merkezi işlem biriminde çalışıyor durumundaki işleme müdahale edemediği için kullanışlı değildir.

2) Round Robin:
"Hazır" durumunda bekleyen her işleme "çalışıyor" durumda geçirecekleri belirli bir zaman dilimi verilir. Bu zaman dilimi yaklaşık 10-100 milisaniyedir.

3) Öncelikle Kısa İş (Shortest Job First):
"Hazır" durumdaki işlemler arasında öncelik en kısa işleme verilir ve buna göre bir sıra oluşturulur. İşletim sistemi tarafından müdahale edilemez olarak tanımlanır.

4) Öncelikle Artan Zamanlı Kısa İş (Shortest Remaining Time First):

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

Bir işlem çalışıyor durumda iken, daha kısa süreli bir iş gelirse "çalışıyor" olan işlem önce "bekliyor", sonra "hazır" duruma getirilir. Merkezi işlem birimine ise kısa süreli iş girer. Müdahale edilebilir bir algoritmadır.

5) Öncelikli Çizelgeleme (Priority Scheduling):
"Hazır" durumdaki işlemler önceliklerine göre gruplanır. Yüksek öncelikli işlemlerden düşük öncelikli işlemlere doğru sıralanır. Bu sıralama esnasında Round Robin ilkesine göre çizelgelenir.

Bunlardan başka algoritmalar da vardır. Ancak çoğu, bu algoritmaların karışımı biçimindedir.

II-) Bellek Yönetimi:

Ana belleğin işlemler arasında paylaştırılmasına ana bellek yönetimi ya da bellek yönetimi, işletim sisteminin bu amaçla oluşturulan kesimine de bellek yöneticisi adı verilir.

Bellekte tutulan bilgiler; komutlar/operatörler (operation code) ve komutların uygulanacağı veriler (operand) olmak üzere ikiye ayrılır. Ana bellekte ayrıca işletim sistemi bulunmakta ve bilgisayar kapanana kadar sürekli bellekte bulunmaktadır.

Bellekte bilgiler ikili düzende saklanmaktadır:

Bit:	0-1	
Nibble:	Fh (15)	(4 Bit=1 Nibbles)
Byte:	FFh (255)	(8 Bit= 2 Nibbles=1 Byte)
Word	FFFFh (65536)	(16 Bit=4 Nibbles=2 Byte=1 Word)

(h, sisteminde sonuna geldiği ifadenin hexadecimal bir ifade olduğunu gösterir)

Bellek yöneticisinin görevleri:
1) Belleğin hangi kısımlarının kullanılıp hangilerinin kullanılmayacağını izlemek
2) İşlemlere gerektiğinde bellek ayırmak
3) İşlem sonlandığında ise işleme ayrılan bellek bölgesini boşaltmak
4) Ana bellekte yer kalmadığında, ikincil belleği kullanmak
5) Ana bellek alanındaki durum bilgisini tutmak

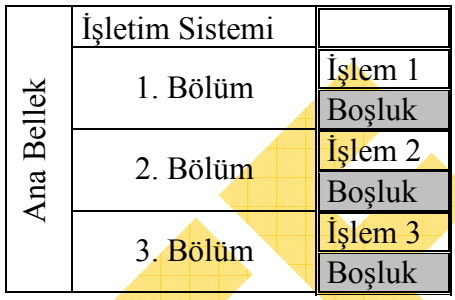
Bellek Yönetim Yöntemleri

1) Tek Programlama Yöntemi
- Tek ve Bitişken Bellek Yönetimi:
Genellikle tek işlemli (single tasking) işletim sistemleri tarafından kullanılır. İşlem ana bellekte yok ise ikincil bellekten ana belleğe yüklenir ve işlem bitimine kadar orada kalır.
Ana belekte çalışma esnasında yalnızca bir tek işlem/program bulunur.

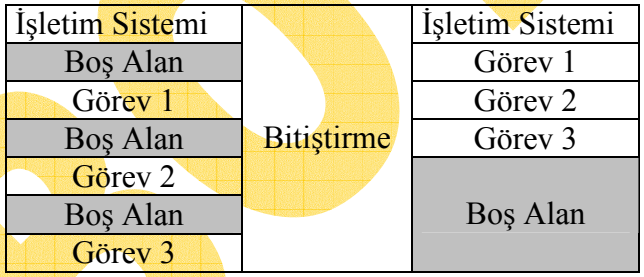
Ana Bellek	İşletim Sistemi
	İşlem
	Boş

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

2) Çoklu Programlama:
a) Değişmez Bölümlü Bellek Yönetimi:
Bu yöntemde bellek, işletim sistemi alanı ve kullanıcı alanı olarak ikiye bölünür. Daha sonra kullanıcı alanı da kendi içerisinde farklı boyutlarda bölümlere ayrılır. İşletime alınacak işlem kendine en uygun büyüklüğe sahip bölüme alınır ve sonlanana kadar orada kalır. Bu yerleşme de işlem boyutu ile bölüm boyutu arasında fark olduğu için boşluklar oluşmaktadır.



b) Değişken Bölümlü Bellek Yönetimi:
Bir önceki yönteme benzer ama ana bellekteki bölümler önceden sabit olarak belirlenmez, işlemler aktifleştiğinde işlemin boyutuna göre belirlenir. Burada kullanılan alanların ve boş alanların yerleri ve boyutları çizelgeler yardımı ile takip edilir.
Örneğin; 300 KB'lik bir bölüme 250 KB'lık bir işlem aktarıldığında 50 KB'lık bir boşluk oluşmaktadır. Buna iç parçalanma adı verilir. Bölümler arasında oluşan boşluklara ise dış parçalanma adı verilir. Boş alanlar sürekli olarak çizelgeler ile takip edilir ve şekilde görüldüğü gibi birleştirilir, bir başka ifade ile dağınık boşluklar bitiştirilerek ana belleğin alt bölgesine yerleştirilir. Boşlukların takip edilmesi, parçalanması ve bitiştirilmesi merkezi işlem birimini yavaşlatır.



c) YeriDeğişir Bölümlü Bellek Yönetimi:
d) Sayfalı Bellek Yönetimi
e) Kesimli Bellek Yönetimi
f) Sayfalı Sanal Bellek Yönetimi
g) Kesimli Sanal Bellek Yönetimi
h) Kesimli-Sayfalı Sanal Bellek Yönetimi

Yukarıdaki yöntemlerin daha iyi anlaşılabilmesi için sanal bellek ve sayfalama (paging) işlemlerinden bahsetmek gerekir.

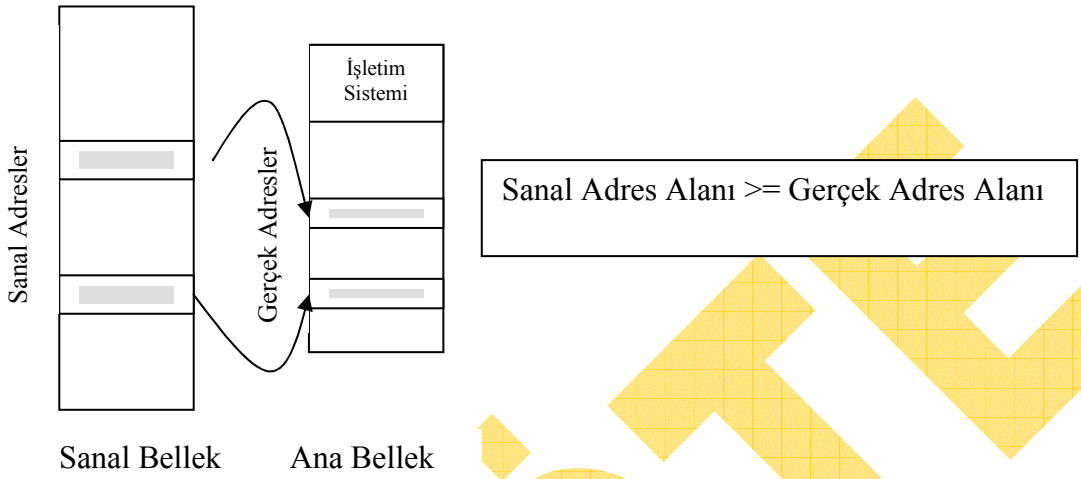
Sanal Bellek (Virtual Memory):

Bilindiği gibi "işlem", öncelikle belleğe yüklenir. Ancak işlemin hacmi, belleğin fizikel hacminden fazla ise, bu durumda sanal bellek tekniği kullanılır.

Sanal bellek tekniğinde, işlemin tamamı değil, onun yerine yalnızca o anda çalıştırılacak bölüm belleğe alınır. Böylelikle ikincil bellek yardımıyla ana belleğin kapasitesi artırılır.

Sanal bellek, ana belleğin bir benzeri olduğu için sanal bellekte de adresler vardır. Ancak bunlar sanal adreslerdir.

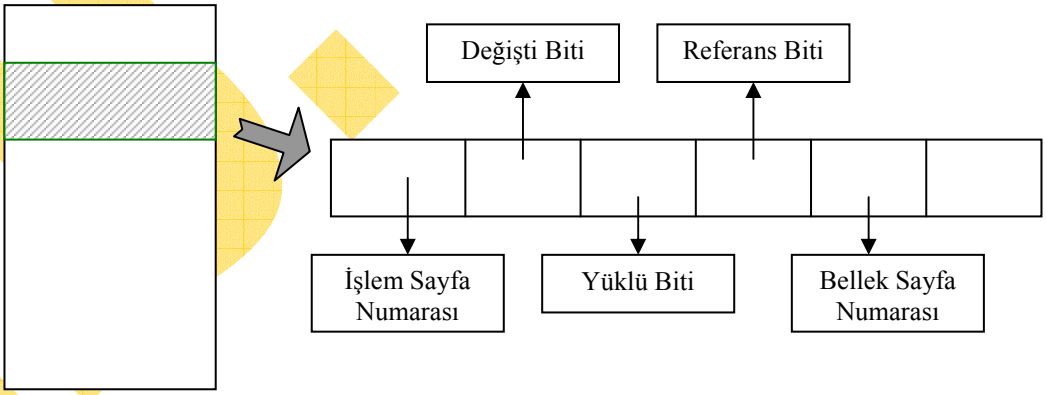
Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr



Sanal bellek tekniği ile sayfalama yöntemi birlikte kullanılır. "İşlem", eşit uzunlukta sayfa adı verilen kısımlara bölünür. İşlemcinin işleyeceği işlemin ilgili sayfası ikincil bellekten ana belleğe yüklenir.

Sayfalama (Paging)

Ana bellek ve işlem sırası ile sayfalara bölünür. Bir diğer ifade ile eşit bölümlere bölünürler. İşlemlere ilişkin sayfalar, sanal bellekte yer alır. İşlemcinin o esnada ihtiyaç duyduğu işlem sayfası, sanal bellekten ana bellek sayfalarına aktarılır. İşletim sistemi, gerçek ve sanal sayfa bilgilerini bir çizelge de tutar. Çizelgenin her bir elemanı, bir sayfaya ilişkin bilgileri tutar.



İşlem Sayfa Numarası: İşlemin sayfasını belirtmek amaçlı kullanılan numaradır.
Değişti Biti: Sayfada bir değişiklik yapılmış ise 1, yapılmamış ise 0'dır.
Yüklü Biti: Sayfa ana ana bellekte ise 1, sanal bellekte ise 0 değerini alır.
Referans Biti: Sayfa daha önce kullanıldı ise 1, kullanılmadı ise 0 değeri alır.
Bellek Sayfa Numarası: Eğer sayfa bellekte ise ana bellekteki adresi içerecek türden bellekteki sayfa numarasını verir.

Bellek Yönetim Birimi; sanal bellek, sayfalama ve sayfa bilgilerinin tutulmasının yanı sıra ana bellek ile sanal bellek arasındaki sayfaların değişimi ile de yükümlüdür. Sanal bellekteki sayfaların ana belleğe yüklenmesi, ana bellekteki sayfaların ise sanal

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

belleğe yüklenmesi her iki bellek arasında sürekli bir sayfa değişimini gerekitrmektedir. Bu değişimler ise bir takım algoritmalara dayalı olarak yapılmaktadır. Bu algoritmalardan bazıları aşağıdadır:

1) FIFO *(First In First Out-İlk Giren İlk Çıkar):* Ana belleğe ilk giren sayfanın işlemcideki işlem hacminin azaldığı varsayımı altında; ana belleğe yüklenecek bir sayfa olduğunda öncelikle ana bellekte en uzun kalan sayfa sanal belleğe, sanal bellekteki sayfa ise ana belleğe aktarılır.

2) LIFO *(Last In First Out-Son Giren İlk Çıkar):* Sanal bellekteki bir sayfa, işlemci tarafından ihtiyaç duyulup çağrılıp ana belleğe yüklendiği için bir sonraki çağrılacak sayfa ana belleğe yüklenirken en son girmiş olan sayfaya ilişkin uygulama "eskimiş" kabul edilir.

3) LFU *(Least Frequently Used-En Az Kullanılan):* Ana bellekteki en az kullanılan sayfa ile değiştirme ilkesine dayanır.

4) NRU *(Not Recently Used-Son Zamanlarda Kullanılmayan):* İşleme alınacak ve diskte bulunan sayfalar, ana bellekte bulunan ve son zamanlarda kullanılmayan sayfalar ile değiştirilme ilkesiyle hareket eden bir algoritmadır.

Yukarıda anlatılan algoritmalar işletim sisteminin bir parçası olan bellek yönetici tarafından "Hata Oluştuğunda" (Page Fault) devreye girer. Buradaki hata ifadesini açıklayalım; Merkezi işlem birimi tarafından çağırılan operant ya da operatörler bellek adresi ile çağrılır. Sayfalama yöntemiyle çalışan bellek yönetim birimlerinde ana bellekte sayfaya ilişkin adreste sayfa olmadığı zaman hata oluşur. Hatanın nedeni, ilgili sayfanın ana bellekte olmaması, sanal bellekte olmasıdır.

Burada bir konuya değinmekte yarar vardır: Ana belleğin büyük hacimli olması, işlemin hepsinin ana belleğe yüklenmesi demek olacağından, düşük hacimli ana belleğe göre yüksek hacimli ana bellekte işletim daha hızlı olacaktır.

Araştırma-İnceleme:

1) a) Critical Statement..?
   b) Semaphor Nedir..?

2) DeadLock Nedir?