Chương 3 - Quản lý bộ nhớ

Kiến trúc máy tính và Hệ điều hành (Đại học Kinh tế Quốc dân)

Chương 3 – Quản lý bộ nhớ

3.1, Địa chỉ vầ các vấn đề liên quan

- Đơn vị đánh địa chỉ có thể là từ máy (words) nhưng thường là **byte**

3.1.2, Địa chỉ logic và địa chỉ vật lý

Địa chỉ lôgic là địa chỉ được gán cho các lệnh và dữ liệu không phụ thuộc vào vị trí cụ thể của tiến trình trong bộ nhớ. => một dạng địa chỉ logic điển hình là địa chỉ tương đối. Toàn bộ địa chỉ được gán trong chương trình tạo thành không gian nhớ lôgic của chương trình.

Để truy cập bộ nhớ, địa chỉ logic cần được biến đổi thành địa chỉ vật lý. Địa chỉ vật lý là địa chỉ chính xác trong bộ nhớ của máy tính ... => địa chỉ tuyệt đối. Thông thường, không gian nhớ vật lý khác với không gian nhớ lôgic của chương trình.

Trong thời gian thực hiện tiến trình, địa chỉ lôgic được ánh xạ sang địa vật lý nhờ một cơ chế phần cứng gọi là khối ánh xạ bộ nhớ (MMU=Memory Mapping Unit)

3.2. Một số các tổ chức chương trình

Liên kết tĩnh

Liên kết động và các thư viện dùng chung:

3.3. Phân chương bộ nhớ

Thường là hệ thống đa chương trình trong đó hệ điều hành cho phép tải và giữ trong bộ nhớ nhiều tiến trình cùng một lúc. Để có thể chứa nhiều tiến trình cùng một lúc trong bộ nhớ, hệ điều hành tiến hành chia sẻ bộ nhớ giữa các tiến trình. Kỹ thuật đơn giản nhất là chia bộ nhớ thành các phần liên tục gọi là chương (partition), mỗi tiến trình sẽ được cung cấp một chương để chứa lệnh và dữ liệu của mình. Quá trình phân chia bộ nhớ thành chương như vậy gọi là phân chương bộ nhớ (partitioning) hay còn gọi là cấp phát vùng nhớ liên tục.

Phân chương được coi là lỗi thời, xem xét kỹ thuật này là cơ sở để tìm hiểu nhiều vấn đề khác trong quản lý không gian nhớ

2 loại phân chương: cố dịnh và động

3.3.1. Phân chương cố định => phân mảnh nội

Bộ nhớ được phân thành những chương có **kích thước cố định ở những vị trí cố định**. Mỗi chương chứa được đúng **một** tiến trình do đó số tiến trình tối đa có thể chứa đồng thời trong bộ nhớ sẽ bị giới hạn bởi số lượng chương

Lựa chọn kích thước chương:

+ Kích thước các chương có thể chọn bằng nhau hoặc không bằng nhau. Việc chọn các chương kích thước bằng nhau mặc dù đơn giản hơn một chút song rất không mềm dẻo. Thực tế, hdh chỉ sử dụng phuong pháp phân chương với kích thước không bằng nhau

+ Muốn cho chương chứa được các tiến trình lớn, ta phải tăng kích thước của chương bằng kích thước của tiến trình lớn nhất

+ Do mỗi process chiếm cả 1 chương, các process nhỏ được cung cấp và chiếm cả chương như 1 tiến trình lớn. Phần bộ nhớ rất đáng kể còn lại của chương sẽ bị bỏ trống gây lãng phí bộ nhớ. Hiện tượng này gọi là phân mảnh trong (internal fragmentation)

Lựa chọn chương nhớ để cấp cho tiến trình đang đợi

- Mỗi chương có 1 hàng đợi
  + Mỗi chương khi đó có một hàng đợi riêng. Tiến trình có kích thước phù hợp với chương nào sẽ nằm trong hàng đợi của chương đó
  + Ưu điểm của cách cấp chương này là cho phép giảm tối thiểu phân mảnh trong.
  + Nhược: Do mỗi chương có một hàng đợi riêng nên sẽ có thời điểm hàng đợi của chương lớn hơn thì rỗng và chương cũng không chứa tiến trình nào, trong khi hàng đợi của chương nhỏ hơn thì có các tiến trình. Các tiến trình nhỏ này buộc phải đợi được cấp chương nhỏ trong khi có thể tải vào chương lớn hơn và chạy.
- Có hàng đợi chung cho tất cả các chương: ....

3.3.2. Phân chương động => phân mảnh ngoại

Kích thước và số lượng chương đều không cố định và có thể thay đổi. Mỗi khi tiến trình được tải vào bộ nhớ, tiến trình được cấp một lượng bộ nhớ đúng bằng lượng bộ nhớ mà tiến trình cần, sau khi kết thúc, tiến trình giải phóng bộ nhớ.

Vùng bộ nhớ do tiến trình chiếm trước đó trở thành một “lỗ” (vùng trống) trong bộ nhớ nếu các vùng nhớ trước và sau thuộc về các tiến trình khác.

Ở mỗi thời điểm, trong bộ nhớ **tồn tại một tập hợp các vùng trống có kích thước khác nhau**. Hệ điều hành sử dụng một bảng để biết được phần bộ nhớ nào đã được dùng, phần nào đang còn trống. Các vùng bộ nhớ cũng có thể được liên kết thành một danh sách kết nối.

+ Tiến trình cần bộ nhớ được xếp trong hàng đợi để chờ tới lượt mình. Mỗi khi đến lượt một tiến trình, hệ điều hành sẽ cố gắng cung cấp bộ nhớ cho tiến trình đó bằng cách tìm một lỗ (vùng bộ nhớ) trống có kích thước lớn hơn hoặc bằng kích thước tiến trình.

+ Trong trường hợp kích thước vùng trống tìm được lớn hơn kích thước tiến trình, vùng trống được chia thành hai phần. Một phần cấp cho tiến trình, **phần còn lại trở thành một vùng trống** có kích thước nhỏ hơn vùng trống ban đầu và được bổ sung vào danh sách các vùng trống mà hệ điều hành quản lý.

+ Mỗi tiến trình sau khi kết thúc tạo ra một vùng trống mới. Nếu vùng trống này nằm kề cận với một vùng trống khác, chúng sẽ được nối với nhau để tạo ra vùng trống mới có kích thước lớn hơn

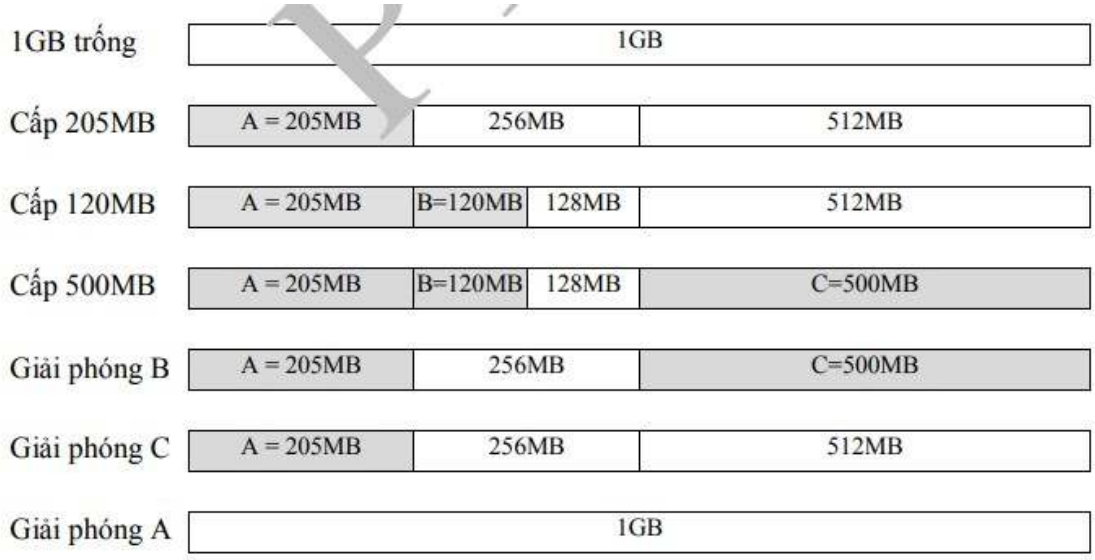
Phân mảnh ngoài. Cùng với thời gian, việc phân chương động có thể sinh ra trong bộ nhớ các vùng trống kích thước quá nhỏ và do vậy không thể cấp phát tiếp cho bất kỳ tiến trình nào. Không gian mà các vùng trống này chiếm do vậy bị bỏ phí. Hiện tượng này gọi là phân mảnh ngoài. Sở dĩ gọi như vậy là do không gian bên ngoài các chương bị chia nhỏ, trái với phân mảnh trong như ta đã nhắc tới ở trên.

Giải pháp: dồn bộ nhớ

Lựa chọn vùng trống để cấp phát: first-ft, best-fit, worst-fit

3.3.3. Phương pháp kề cận

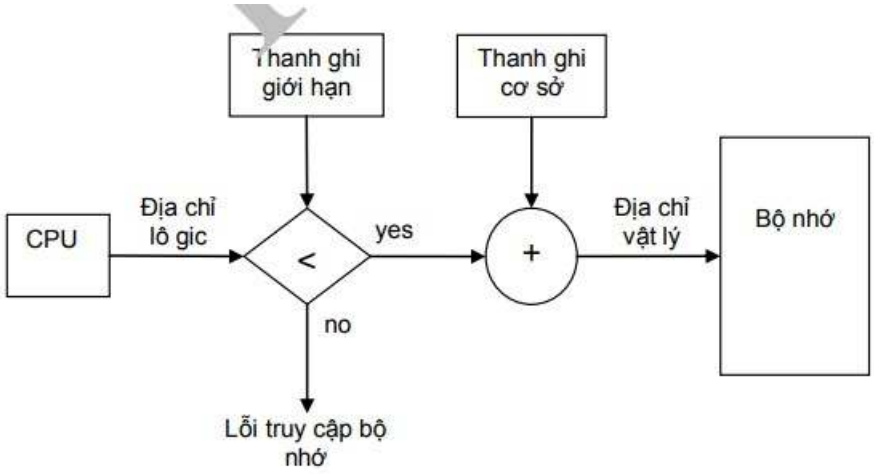
Dung hòa 2 nhược điểm phương pháp trên: pp kề cận (buddy systems)



Hình 3.5. Ví dụ cấp phát và giải phóng chương nhớ sử dụng kỹ thuật kề cận

3.3.4. Ánh xạ địa chỉ và chống truy cập trái phép

Với nhiều tiến trình chứa trong bộ nhớ, các tiến trình có thể vô tình (do lỗi) hoặc cố ý truy cập tới vùng bộ nhớ thuộc tiến trình khác. Việc truy cập trái phép có thể phá vỡ an toàn bảo mật thông tin.



Hình 3.6. Cơ chế ánh xạ địa chỉ và chống truy cập trái phép

3.3.5. Swapping -> không thi

3.4. Phân trang bộ nhớ

Nhược điểm của việc phân chương bộ nhớ là tạo ra phân mảnh: phân mảnh trong đối với phân chương cố định, phân mảnh ngoài đối với phân chương động, hậu quả đều là việc không tận dụng hết bộ nhớ.

3.4.1. Khái niệm

Trong các hệ thống phân trang:

+ Bộ nhớ vật lý được chia thành các khối nhỏ kích thước cố định và bằng nhau gọi là khung trang (page frame) hoặc đơn giản là khung => kí hiệu là **f**

+ Không gian địa chỉ lôgic của tiến trình cũng được chia thành những khối gọi là trang (page) có kích thước = kích thước khung

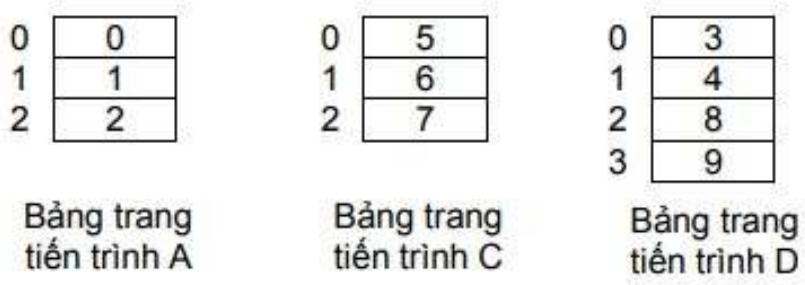
+ Mỗi tiến trình sẽ được cấp các khung để chứa các trang nhớ của mình. Các khung thuộc về một tiến trình có thể nằm ở các vị trí khác nhau trong bộ nhớ chứ không nhất thiết nằm liền nhau theo thứ tự các trang

+ Kỹ thuật phân trang có điểm tương tự với phân chương cố định, trong đó mỗi khung tương tự một chương, có kích thước và vị trí không thay đổi. Tuy nhiên, khác với phân chương, mỗi tiến trình được cấp phát nhiều khung kích thước nhỏ, nhờ vậy giảm đáng kể phân mảnh trong. Còn trang là kích thước cố định cmnr

+ Khi phân trang, khoảng không gian bỏ phí do phân mảnh trong bằng phần không gian không sử dụng trong trang cuối của tiến trình

3.4.2. Ánh xạ địa chỉ

Bảng trang: Mỗi ô của bảng tương ứng với một trang và chứa số thứ tự hay địa chỉ cơ sở của khung chứa trang đó trong bộ nhớ. Mỗi tiến trình có bảng trang riêng của mình



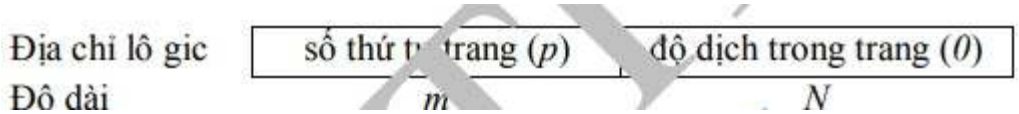
Hình 3.8. Ví dụ bảng trang của các tiến trình

Địa chỉ:

- **LOGIC:** gồm hai phần: số thứ tự trang (p) và độ dịch (o) của địa chỉ tính từ đầu trang đó.
  + số thứ tự trang (p): được dùng để tìm ra ô tương ứng trong bảng trang, từ bảng ta tìm được địa chỉ cơ sở của khung tương ứng
  + độ dịch (o) của địa chỉ tính từ đầu trang đó

  Đỉa chỉ vật lý = địa chỉ cơ sở của khung tương ứng + độ dịch trang

- **Cấu trúc địa chỉ logic:**



Độ dài địa chỉ = m + n **bit. Tương ứng với không gian nhớ logic của tiến trình: 2^(m+n)**

Phần m bit cao chứa STT của trang, và n bit thấp chứa độ dịch trong nhớ. Kích thước trang nhớ = 2^n. Không gian địa chỉ lôgic của tiến trình cũng được chia thành những khối gọi là trang (page) có kích thước = kích thước khung

Ví dụ: cho trang nhớ kích thước 1024B, độ dài địa chỉ lô gic 16 bit. Địa chỉ lô gic 1502, hay 0000010111011110 ở dạng nhị phân, sẽ có:

Phép chia:

Phần p = 1502/1024 = 1. Tương ứng với 6 bit cao: 000001

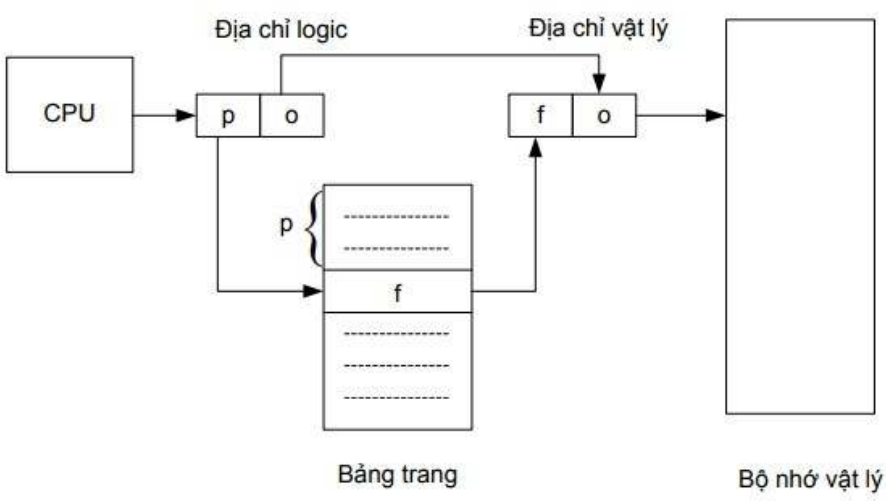
Phần o = 1502 mod 1024 = 478 ứng với 10 bit thấp: 0111011110

Có thể suy ra bằng cách ra như sau

TH: kích thước trang luôn được chọn bằng lũy thừa của 2

Trang nhớ = 1024B = 2^10B => o = 10, p = độ dài địa chỉ - o = 16 – 10 = 6

Việc phân trang phải dựa vào sự hỗ trợ của phần cứng khi biến đổi địa chỉ, kích thước trang và khung do phần cứng quyết định. **Kích thước trang luôn được chọn bằng lũy thừa của 2**, và nằm trong khoảng từ 512B đến 1GB. Lưu ý rằng với kích thước trang bằng lũy thừa của 2, việc tách phần p và o trong địa chỉ lô gic được thực hiện dễ dàng bằng phép dịch bit thay vì thực hiện phép chia và chia mô đun thông thường

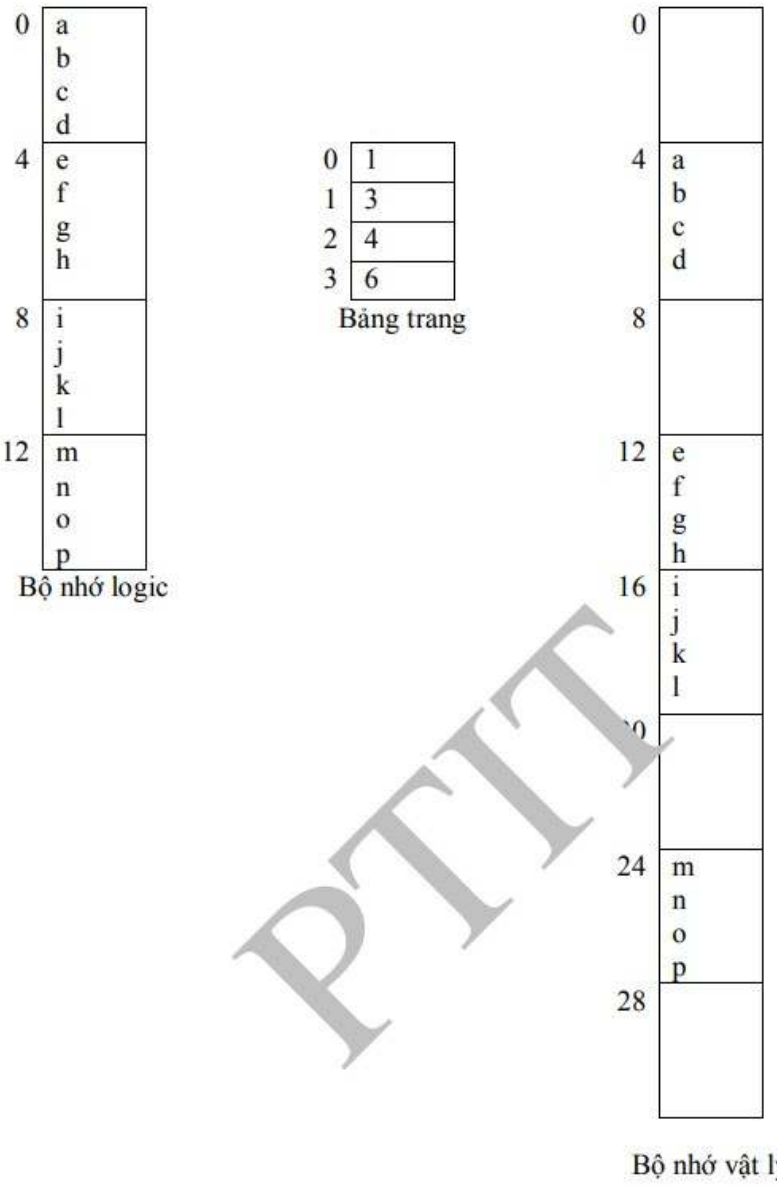


Không gian nhớ logic có kích thước 16B = 2^4 , gồm 4 trang, bộ nhớ vật lý có 8 trang, kích thước trang nhớ bằng 4B = 2^2

- Không gian nhớ ta suy ra được độ dài logic: 2^4 => **độ dài địa chỉ logic: 4 bit = p + o**
- Bộ nhớ vật lý gồm 8 trang => 2^3 => **f : 3bit**
- Kích thước trang nhớ: 2^2 => **o = 2 => p = 2**
- Không gian nhớ vật lý là 32B = 2^5 => **độ dài địa chỉ vật lý: 5 bit = f + o**

Kích thước trang: Giảm phân mạch nội, làm cho kích thước trang nhỏ. Tuy nhiên, kích thước trang nhỏ làm số lượng trang tăng lên, dẫn tới bảng trang to hơn, khó quản lý.

Quản lý khung: Mỗi khoản mục của bảng khung ứng với một khung của bộ nhớ vật lý và chứa các thông tin về khung này



Hình 3.10. Ví dụ ánh xạ địa chỉ trong phân trang bộ nhớ

Nhìn hình. Ta cũng có thể suy ra 1 số thông tin thú vị sau

- Không gian nhớ logic: 16 = 2^4 => địa chỉ logic dài 4 bit
- Không gian nhớ vật lý: 32 = 2^5 => đỉa chỉ vật lý dài 5 bit
- Bảng trang gồm 4 số thứ tự: Ứng với các trường hợp xảy ra của p (p trong địa chỉ logic để xác định STT trang) = 2^2 => p gồm 2 bit
- Với dự kiện hình ảnh trên tiếp tục tìm được o = 2 bit (địa chỉ logic – p) => kích thước trang nhớ = 4B
- f = 3 bit (địa chỉ vật lý – o) => Bộ nhớ vật lý có 2^3 trang = 8 trang.

Tôi sẽ ví dụ TH:

Địa chỉ logic 1 nằm tại trang 0 và độ dịch từ đầu trang là 1 (1/4 = 0; 1 module 4 = 1. **4 ở đây là kích thước trang nhớ : 2^(số bit o)**). Theo bảng trang, trang 0 nằm trong khung 1, do vậy địa chỉ vật lý tương ứng là 1*4 + 1 = 5. (Đỉa chỉ vật lý = địa chỉ cơ sở của khung tương ứng + độ dịch trang)

Trong đó địa chỉ cơ sở của khung tương ứng = khung tương ứng * kích thước trang nhớ do trang (page) có kích thước = kích thước khung tôi đã nhắc ở trên

Biểu diễn dưới dạng nhị phân, ta có: địa chỉ logic = 0001, phần p = 00, phần o = 01, địa chỉ khung f = 001, địa chỉ vật lý (f,o) = 00101 = 5.

Cái này có cái khá là hay, khi bạn chuyển từ nhị phân sang thập phân

Địa chỉ vật lý = 00101 = (0*2^4 + 0*2^3 + 1*2^2)+ (0*2^1 + 1*2^0)

= 2^2*(0*2^2 + 0*2^1 + 1) + (0*2^1 + 1*2^0)

(Lí do có 2^2. Do đằng sau nó còn 2 số ứng với 2 bit của offset. Mà kích thước trang nhớ luông bằng = 2^(độ dài bit o). Nếu o chiếm 3 bit, thì cái kia là 2^3.)

= Kích thước trang nhớ * frame tương ứng + độ dịch so với đầu trang.

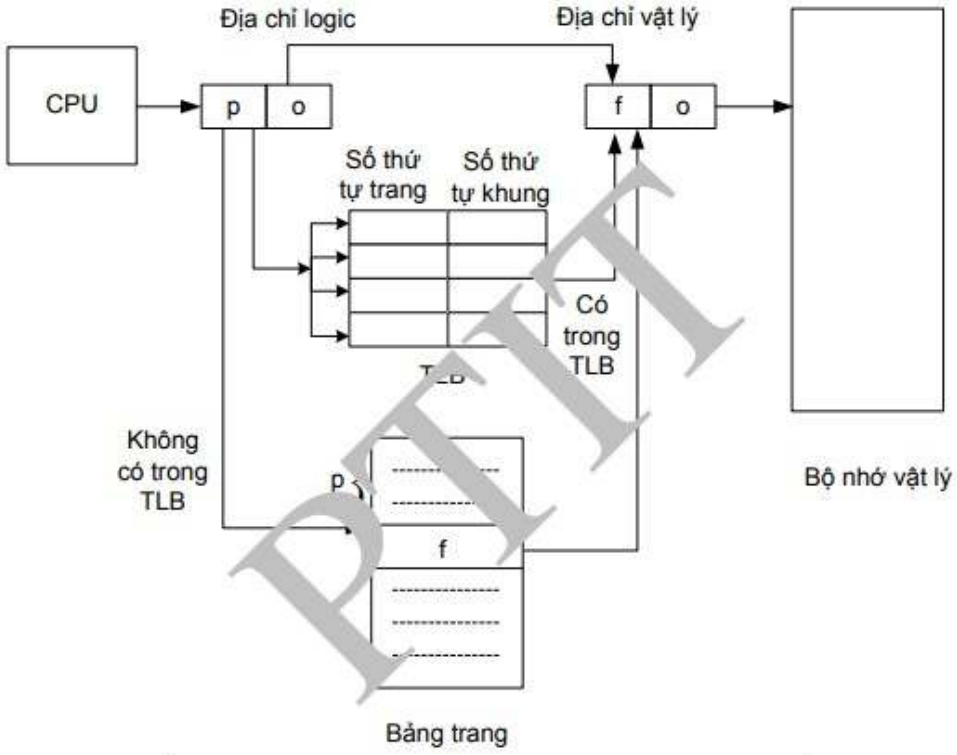
Ví dụ nhé: cho độ dài vật lý : 0101|101. Với f chiếm 4 bit, o chiếm 3 bit

⇨ Độ dài vật lý = 2^3 * (0101 => 5) + (101 => 5) = 8*5+5 = 45

Bạn có thể làm tùy cách bạn muốn, tôi chỉ đưa ra 1 góc nhìn thêm về bài thôi.

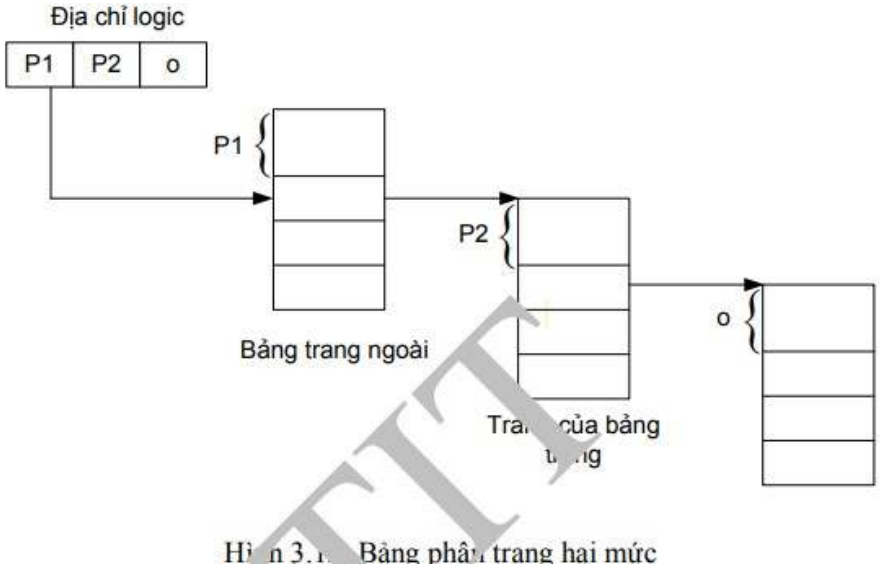
Tương tự, địa chỉ logic 13 nằm tại trang 3 (13/4 = 3), độ dịch từ đầu trang là 1 (13 module 4 = 1). Theo bảng trang, trang 3 nằm ở khung 6, do vậy địa chỉ vật lý = 4*6 + 1 = 25. Biểu diễn dưới dạng nhị phân, ta có: địa chỉ logic = 1101, phần p = 11, phần o = 01, địa chỉ khung f = 110, địa chỉ vật lý (f,o) = 11001 = 25

BẢNG TRANG TRONG CACHE (sử dụng TBL – Translation Look-aside Buffer): đệm dịch địa chỉ



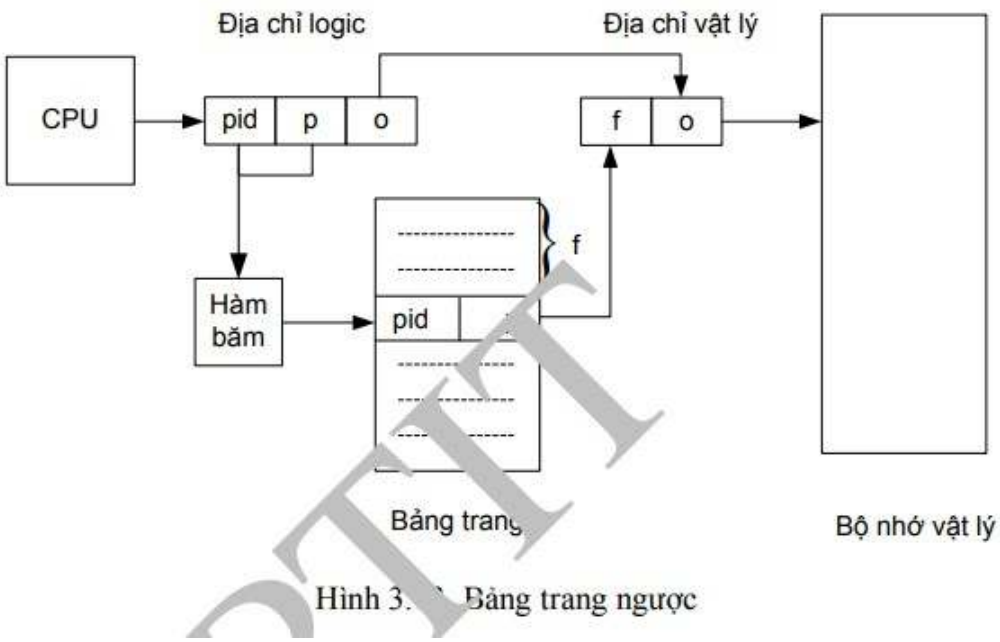
Hình 3.11. Kết hợp đệm dịch địa chỉ (TLB) với bảng trang nằm trong bộ nhớ trong

BẢNG TRANG NHIỀU MỨC



Hình 3.12. Bảng phân trang hai mức

BẢNG TRANG NGƯỢC



Hình 3. Bảng trang ngược

3.5. Phân đoạn bộ nhớ

3.5.1. Khái niệm

Khi phân trang, chương trình được chia thành các trang kích thước đều nhau, không quan tâm đến tổ chức lôgic và ý nghĩa các thành phần.

Một cách tổ chức khác cho phép tính tới tổ chức lôgic của chương trình là phân đoạn (segmentation). Chương trình được chia thành những phần kích thước khác nhau gọi là đoạn (segment) tuỳ theo ý nghĩa của chúng

Nếu như phân trang giống phân chương cố định **thì phân đoạn bộ nhớ giống với phân chương động** ở chỗ bộ nhớ được cấp phát theo từng vùng kích thước thay đổi. Tuy nhiên, khác với phân chương động, mỗi chương trình có thể chiếm những đoạn bộ nhớ không nằm liền kề nhau. Mỗi khi có yêu cầu cấp phát bộ nhớ cho các đoạn, thuật toán cấp phát first-fit hoặc best-fit như phân chương động sẽ được sử dụng.

Phân đoạn tạo phân mảnh ngoài

3.5.2. Ánh xạ địa chỉ và trống truy cập trái phép

Địa chỉ: Khi sử dụng phân đoạn, không gian nhớ lô gic sẽ bao gồm không gian nhớ của các đoạn. Do vậy, địa chỉ lô gic bao gồm hai thành phần, chỉ rõ hai thông tin: **địa chỉ thuộc đoạn nào, và độ dịch từ đầu đoạn là bao nhiêu**

Đối với đoạn được đánh số, địa chỉ lô gic sẽ có hai thành phần, được cấu trúc như sau:

< số thứ tự đoạn (s - segment), độ dịch trong đoạn (o - offset) >

Ánh xạ địa chỉ: Tương tự như trong trường hợp phân trang, việc ánh xạ được thực hiện dựa trên bảng đoạn (segment table). Mỗi ô của bảng đoạn chứa địa chỉ cơ sở và giới hạn của đoạn.

Địa chỉ cơ sở là vị trí bắt đầu của đoạn trong bộ nhớ máy tính, còn giới hạn đoạn chính là độ dài đoạn và sẽ được sử dụng để chống truy cập trái phép ra ngoài đoạn. Mỗi tiến trình có một bảng như vậy.

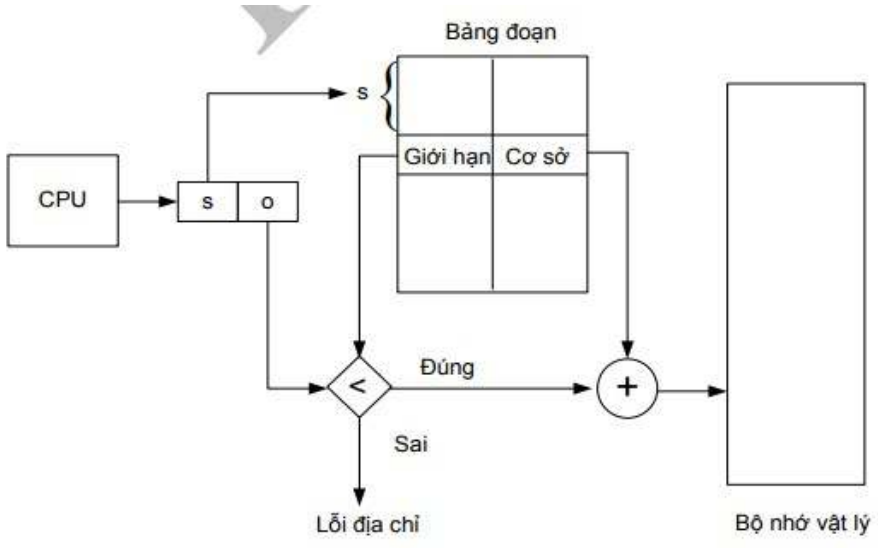
Bảng đoạn được sử dụng kết hợp với một cơ chế phần cứng cho phép biến đổi từ địa chỉ lô gic sang địa chỉ vật lý.

+ phần s trong địa chỉ được sử dụng để tìm tới ô thứ s trong bảng đoạn và truy cập hai giá trị giới hạn và cơ sở chứa trong ô này

+ phần độ dịch o của địa chỉ được so sánh với phần giới hạn chứa trong ô. Nếu o nhỏ hơn giới hạn thì đây là truy cập hợp lệ, ngược lại nếu o lớn hơn hoặc bằng giới hạn thì địa chỉ này vượt ra ngoài phạm vi của đoạn và do vậy sẽ bị báo lỗi truy cập.

⇨ Địa chỉ vật lý = phần độ dịch o + địa chỉ cơ sở

Nhận xét thì phần này nó khá dễ, dạng bài không có gì ngoài hỏi cấp phát có hợp lệ hay không.



Hình 3.14. Cơ chế ánh xạ địa chỉ khi phân đoạn

PHẦN NẠP TRANG THÌ KHÁ DỄ. X: là số lỗi trang, bạn tự tính ra theo yêu cầu

FIFO: thằng nào dùng lâu thì ra

1	3	1	4	2	1	5	6	2	1	2	3	6	7	3	3	2	1	2	3	6
1	1		1	1		5	5		5		5		7			7				7
	3		3	3		3	6		6		6		6			2				2
			4	4		4	4		1		1		1			1				6
				2		2	2		2		3		3			3				3

x	x		x	x		x	x		x		x		x		x		x			x

LRU: xem quá khứ, ít nhắc tới thì ra

1	3	1	4	2	1	5	6	2	1	2	3	6	7	3	3	2	1	2	3	6
1	1		1	1		1	1				1		7				7			6
	3		3	3		5	5				3		3				3			3
			4	4		4	6				6		6				1			1
				2		2	2				2		2				2			2
x	x		x	x		x	x				x		x				x			x

OPT: nhìn tương lai, tương lai xuất hiện càng muộn hoặc không xuất hiện thì ra

1	3	1	4	2	1	5	6	2	1	2	3	6	7	3	3	2	1	2	3	6
1	1		1	1		1	1						1							6
	3		3	3		3	3						3							3
			4	4		5	6						7							7
				2		2	2						2							2
x	x		x	x		x	x						x							x

cột cuối, thằng nào ở lâu thì đẩy ra (FIFO)

TỔNG KẾT CHƯƠNG NÀY: BẠN NÊN QUAN TÂM. CÁI KHÁC BẠN ĐỌC CHO BIẾT

+ PHÂN CHƯƠNG CỐ ĐỊNH, ĐỘNG.

- **Cố định nó sẽ hỏi phân mảnh nội**
- **Động thì các dạng best-fit, first-fit, worst-fit**

+ PHÂN TRANG

- **Do cái này giống với phân chương động nên là họ sẽ hỏi phân mảnh nội khi phân trang. Khá dễ thôi**
- **Tính các thông số như độ dài logic, vật lý, kích thước trang nhớ ,...**
- **Ánh xạ địa chỉ logic sang vật lý**

+ PHÂN ĐOẠN

- **Ánh xạ địa chỉ chống truy cập trái phép -> dễ nhất**

+ NẠP TRANG

Mọi thứ tôi ghi đều ở trong giáo trình PTIT, kia chỉ là tóm gọn lại lí thuyết cho dễ hiểu thôi.