



Chapter 3. Memory Management

Main Memory

Virtual Memory

Nội dung

Bộ nhớ chính

Bộ nhớ ảo

1.Bộ nhớ chính

- Các yêu cầu đối với việc quản lý bộ nhớ
- Các kiểu địa chỉ nhớ
- Các chiến lược quản lý bộ nhớ

- Bộ nhớ chính là thiết bị lưu trữ duy nhất thông qua đó CPU có thể trao đổi thông tin với môi trường ngoài.
- Nhu cầu tổ chức, quản lý bộ nhớ là một trong những nhiệm vụ trọng tâm hàng đầu của hệ điều hành.
- Bộ nhớ chính được tổ chức như một mảng một chiều các từ nhớ (word), mỗi từ nhớ có một địa chỉ.
- Việc trao đổi thông tin với môi trường ngoài được thực hiện thông qua các thao tác đọc hoặc ghi dữ liệu vào một địa chỉ cụ thể nào đó trong bộ nhớ.

Các yêu cầu đối với việc quản lý bộ nhớ

- Cấp phát bộ nhớ cho các tiến trình
- Bảo vệ: Phải kiểm tra truy xuất bộ nhớ có hợp lệ không
- Chia sẻ: Cho phép các tiến trình chia sẻ vùng nhớ chung
- Ánh xạ địa chỉ luận lý (logic) của người sử dụng vào địa chỉ thực (vật lý)
- Tái định vị (relocation): khi swapping,...

Các kiểu địa chỉ nhớ

- Địa chỉ vật lý
- Địa chỉ luận lý
- Chuyển đổi địa chỉ logic sang địa chỉ vật lý và ngược lại.

Địa chỉ vật lý & địa chỉ luận lý

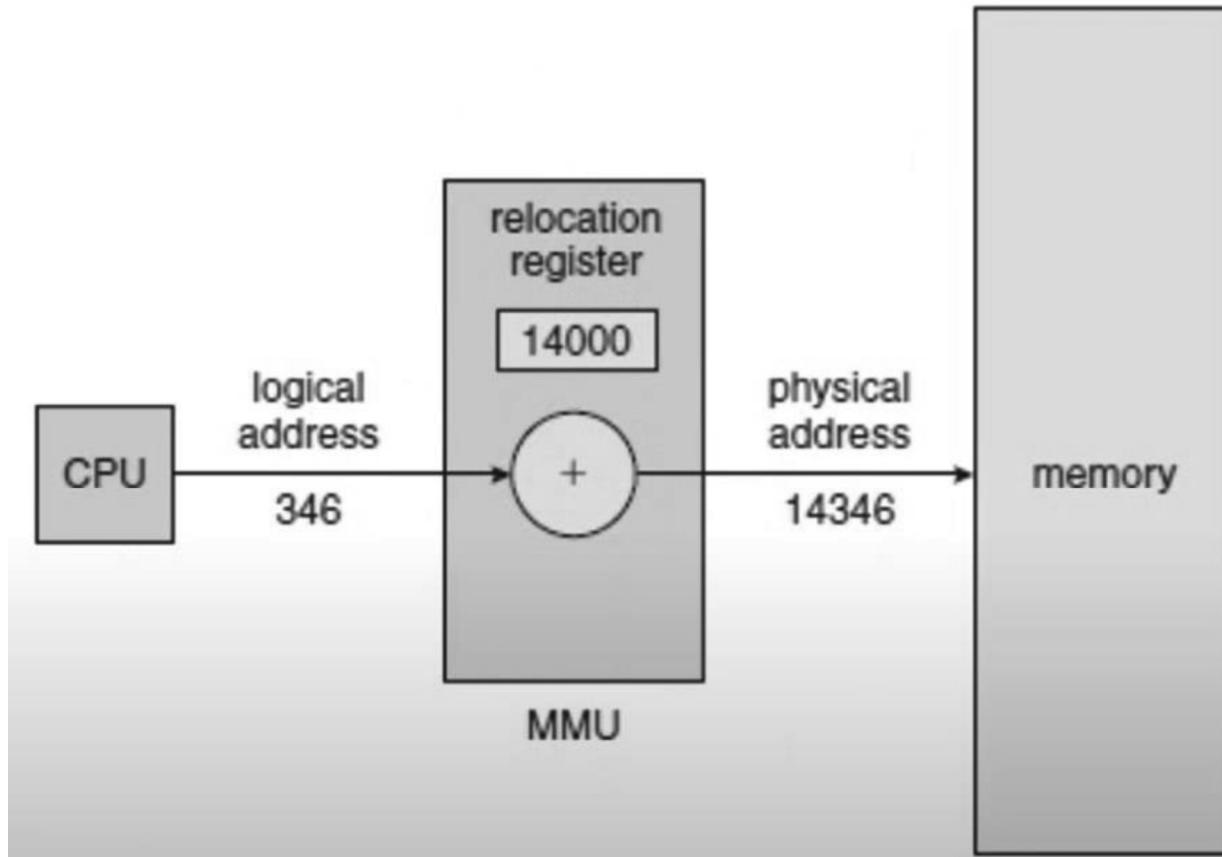
- Địa chỉ vật lý (physical address – địa chỉ thực, địa chỉ tuyệt đối): Là một vị trí thực trong bộ nhớ chính.
- Địa chỉ luận lý (logical address – địa chỉ logic): Địa chỉ được CPU tạo ra, là một vị trí nhớ được diễn tả trong một chương trình.
 - Các trình biên dịch tạo ra mã lệnh chương trình mà trong đó mọi tham chiếu bộ nhớ đều là địa chỉ luận lý.
 - Địa chỉ tương đối là một kiểu địa chỉ luận lý, trong đó các địa chỉ được biểu diễn tương đối so với một vị trí xác định nào đó trong chương trình.
 - Địa chỉ tuyệt đối là địa chỉ tương đương với địa chỉ thực.

Chuyển đổi địa chỉ logic sang địa chỉ vật lý

- Khi một lệnh được thực thi, các tham chiếu đến địa chỉ luận lý phải được chuyển đổi thành địa chỉ thực.
- Thao tác chuyển đổi này thường có sự hỗ trợ của phần cứng máy tính để đạt năng suất cao.

MMU (memory management unit)

- MMU là một cơ chế phần cứng được sử dụng để thực hiện chuyển đổi địa chỉ ảo thành địa chỉ vật lý vào thời điểm xử lý.
- Chương trình của người sử dụng chỉ thao tác trên các địa chỉ ảo.
- Địa chỉ thật sự ứng với vị trí của dữ liệu trong bộ nhớ chỉ được xác định khi thực hiện truy xuất đến dữ liệu.



Dynamic relocation using a relocation register.

Các chiến lược quản lý bộ nhớ

Chiến lược phân chương cố định

Chiến lược phân chương động

Chiến lược phân đoạn

Chiến lược phân trang

Chiến lược kết hợp phân đoạn - phân trang

Chiến lược phân chương cố định

Bộ nhớ được chia thành n phần:

- Mỗi phần gọi là một chương (partition).
- Mỗi chương ở một vị trí cố định.
- Chương được sử dụng như một vùng nhớ độc lập: Mỗi chương chứa đúng một tiến trình. Tiến trình khi được tải vào sẽ được cấp phát một chương; sau khi tiến trình kết thúc, hệ điều hành giải phóng chương và chương có thể được cấp phát cho tiến trình mới.
- Các chương không nhất thiết phải có kích thước bằng nhau.

Chiến lược phân chương động

- Kích thước, số lượng và vị trí chương có thể thay đổi.
- Các vùng bộ nhớ trống cũng có thể được liên kết thành một danh sách kết nối.
- Khi một tiến trình yêu cầu bộ nhớ:
 - ✓ Hệ điều hành cấp cho tiến trình một chương có kích thước đúng bằng kích thước tiến trình đó đang cần.
 - ✓ Nếu tìm thấy vùng nhớ tương ứng thì vùng trống được chia thành 2 phần: một phần cung cấp theo yêu cầu, một phần trả lại danh sách vùng trống tự do.
 - ✓ Nếu không tìm thấy thì phải chờ tới khi có được một vùng trống thỏa mãn.
 - ✓ Cho phép tiến trình khác trong hang đợi thực hiện (nếu độ ưu tiên đảm bảo).
- Khi tiến trình kết thúc sẽ tạo vùng trống trong bộ nhớ và các vùng trống nằm cạnh nhau được nhập lại thành vùng lớn hơn.

Các chiến lược lựa chọn vùng trống tự do

❖ First-fit:

Chọn khối nhớ trống đầu tiên có kích thước đủ lớn để nạp tiến trình.

❖ Best-fit:

Chọn khối nhớ có kích thước nhỏ nhất vừa đúng bằng kích thước của tiến trình cần được nạp vào bộ nhớ.

❖ Worst-fit:

Chọn khối nhớ có kích thước lớn nhất để nạp tiến trình.

❖ **Next-fit:**

Tương tự như First-fit, nhưng hệ điều hành bắt đầu quét từ khối nhớ trống kế sau khối nhớ vừa được cấp phát và chọn khối nhớ trống kế tiếp đủ lớn để nạp tiến trình.

❖ **Last-fit:**

Chọn khối nhớ trống cuối cùng có kích thước đủ lớn để nạp tiến trình.

Chiến lược phân đoạn

- Chương trình thường gồm các modul
 - Một chương trình chính (main program)
 - Tập các chương trình con
 - Các biến, các cấu trúc dữ liệu,...
- Khi đưa chương trình vào bộ nhớ để thực hiện: các chương trình gồm nhiều đoạn khác nhau; mỗi đoạn là một khối logic ứng với một modul. Một đoạn chiếm một vùng nhớ liên tục: có vị trí bắt đầu và kích thước, có thể nằm tại bất cứ đâu trong bộ nhớ; đối tượng trong đoạn được xác định bởi vị trí tương đối so với đầu đoạn.

Chiến lược phân trang

- Bộ nhớ vật lý được chia thành từng khối có kích thước bằng nhau: khung trang vật lý (frames). Khung trang vật lý được đánh số 0,1,2,....
- Bộ nhớ logic (chương trình) được chia thành những khối gọi là trang (page) có kích thước bằng khung trang vật lý: trang logic (pages).
- Tiến trình được cấp các khung để chứa các trang của mình.
- Trang có thể chứa trong các khung nằm rải rác trong bộ nhớ.
- Bảng trang: HĐH quản lý việc cấp phát khung cho mỗi tiến trình bằng bảng trang (page table): mỗi ô tương ứng với một trang và chứa số khung cấp cho trang đó. Mỗi tiến trình có bảng trang riêng.

Vấn đề nạp trang và thay thế trang

- Số trang vật lý dành cho chương trình lớn: Thực hiện nhanh nhưng hệ số song song giảm. Số trang vật lý dành cho chương trình bé: Hệ số song song cao nhưng thực hiện chậm do tình huống thiếu trang.
- Hiệu quả phụ thuộc các chiến lược nạp trang và thay thế trang.
- Các chiến lược nạp trang:
 - ✓ Nạp tất cả: Nạp toàn bộ chương trình
 - ✓ Nạp trước: dự báo trang cần thiết tiếp theo
 - ✓ Nạp theo yêu cầu: chỉ nạp khi cần thiết
- Các chiến lược thay thế trang (sẽ đề cập trong phần bộ nhớ ảo).
 - ✓ FIFO: First In First Out
 - ✓ LRU: Least Recently Used
 - ✓ LFU: Least Frequently Used

Ví dụ 1a. Chuyển đổi địa chỉ logic sang địa chỉ vật lý

Cho bảng phân trang như sau:

Page	Frame
0	3
1	2
2	6
3	4

Với kích thước mỗi trang là 1KB, hãy chuyển các địa chỉ sau sang địa chỉ vật lý.

+ 1240 (*)

+ 3580

+ 5540

Gợi ý

B1: 1Kb = 1024byte

B2: (*) div 1024 : lấy phần nguyên của nó (Page) → Frame nào?

B3: Frame x 1024 + ((*) % 1024) → thì nó sẽ ra địa chỉ vật lý

B1: 1Kb = 1024byte

B2: (*) div 1024 : lấy phần nguyên của nó (Page) → Frame nào???

B3: Frame x 1024 + ((*) % 1024) → thì nó sẽ ra địa chỉ vật lý

+ 1240: $1240 \text{ div } 1024 = 1$ → Frame 2

$2 \times 1024 + (1240 \% 1024) = 2048 + 216 = 2264$ (đây là địa chỉ vật lý của 1240)

+ 3580: $3580 \text{ div } 1024 = 3$ → frame 4

$4 \times 1024 + (3580 \% 1024) = 4096 + 508 = 4604$ (đây là địa chỉ vật lý của 3580)

+ 5540: $5540 \text{ div } 1024 = 5$ dư 420 ==> Địa chỉ này không hợp lệ.

Ví dụ 1b. Chuyển đổi địa chỉ logic sang địa chỉ vật lý

- Cho địa chỉ logic 3654 sẽ được chuyển thành địa chỉ vật lý là bao nhiêu? Biết rằng kích thước mỗi frame là 2K bytes và bảng ánh xạ địa chỉ logic như hình sau:

0	6
1	4
2	5
3	7
4	1
5	9

Page Table

Tóm tắt: Số chi logic: 3654
đơn byte

$$\text{đơn byte} = 2048 \text{ byte}$$

$$3654 \text{ div } 2048 = 1$$

$$\Rightarrow \text{Frame : } 4$$

$$\begin{aligned} \text{Số chi rã ly'} &= 4 \times 2048 + (3654 \% 2048) \\ &= 9798 \end{aligned}$$

Số chi rã ly' là: 9798

Ví dụ 1c. Chuyển đổi địa chỉ vật lý sang địa chỉ logic

Cho địa chỉ vật lý 6578 sẽ được chuyển thành địa chỉ logic là bao nhiêu? Biết rằng kích thước mỗi frame là 1K bytes và bảng ánh xạ địa chỉ logic như hình sau:

0	6
1	4
2	5
3	7
4	1
5	9

Page Table

Địa chỉ vật lý: 6578

1KB

1KB = 1024 byte

$$\text{G}: 6578 \text{ div } 1024 = 6$$

\Rightarrow page : 0

$$\text{Địa chỉ logic: } 0 \times 1024 + (6578 - 6 \cdot 1024)$$

$$= 434$$

Vậy địa chỉ logic là 434

Ví dụ 2. Địa chỉ vật lý & địa chỉ luận lý trong phân trang

Xét một không gian địa chỉ luận lý có 8 trang, mỗi trang có kích thước 1KB. Ánh xạ vào bộ nhớ vật lý có 32 khung trang.

- a) Địa chỉ logic gồm bao nhiêu bit ?
- b) Địa chỉ physic gồm bao nhiêu bit ?
- c) Bảng trang có bao nhiêu mục? Mỗi mục trong bảng trang cần bao nhiêu bit?

Lũy thừa của 2

$$1KB = 2^{10}.$$

- a Số bit của địa chỉ logic: $8 \times 2^{10} = 2^3 \times 2^{10} = 2^{13}$
⇒ 13 bit địa chỉ luận lý.
- b Số bit của địa chỉ vật lý: $32 \times 2^{10} = 2^5 \times 2^{10} = 2^{15}$
⇒ 15 bit cho địa chỉ vật lý
- c Bảng trang có 8 mục.

Bộ nhớ vật lý 32 byte . Lưu trữ từ 0 – 31

2 Lũy thừa của 5 để lưu trữ 31 nên mỗi mục chỉ cần 5 bit.

*Ghi chú:

Trong bảng trang chỉ có chứa khung trang, vì trang được đánh mặc định
tăng dần 0,1,2,3,..

Ví dụ 3.

- Giả sử khối nhớ chính được phân thành các phân vùng có thứ tự và kích thước là 580KB, 500KB, 200KB, 300KB.
- Các tiến trình có thứ tự và kích thước là 212KB, 417KB, 112KB, 426KB sẽ được cấp phát như thế nào theo các chiến lược sau:
 - First-fit:
 - Best-fit:
 - Worst-fit:
 - Next-fit:
 - Last-fit:
- Chiến lược cấp phát bộ nhớ nào trong các chiến lược kể trên là hiệu quả nhất trong trường hợp này (trong ví dụ này).

Bài giải

	580K	500K	200K	300K
First-fit	212K,112K	417K		
Best-fit	426K	417K	112K	212K
Worst-fit	212K, 112K	417K		
Next-fit	212K	417K	112K	
Last-fit	426K	417K	112K	212K

Chiến lược cấp phát bộ nhớ Best-fit và Last-fit là hiệu quả nhất đối với bài tập cụ thể này.

2. Bộ nhớ ảo

Dẫn nhập

Định nghĩa

Cài đặt bộ nhớ ảo

Lỗi trang

Các thuật toán thay thế trang

Dẫn nhập

- Nếu đặt toàn thể không gian địa chỉ vào bộ nhớ vật lý, kích thước của chương trình bị giới hạn bởi kích thước bộ nhớ vật lý.
- Giải pháp được đề xuất là cho phép thực hiện một chương trình chỉ được nạp từng phần vào bộ nhớ vật lý. Ý tưởng chính của giải pháp này là tại mỗi thời điểm chỉ lưu trữ trong bộ nhớ vật lý các chỉ thị và dữ liệu của chương trình cần thiết cho việc thi hành tại thời điểm đó. Khi cần đến các chỉ thị khác, những chỉ thị mới sẽ được nạp vào bộ nhớ, tại vị trí trước đó bị chiếm giữ bởi các chỉ thị này không còn cần đến nữa. Với giải pháp này, một chương trình có thể lớn hơn kích thước của vùng nhớ cấp phát cho nó.

- Một cách để thực hiện ý tưởng của giải pháp trên là sử dụng kỹ thuật overlay. Kỹ thuật overlay không đòi hỏi bất kỳ sự trợ giúp đặc biệt nào của hệ điều hành, nhưng trái lại, lập trình viên phải biết cách lập trình theo cấu trúc overlay, và điều này đòi hỏi khá nhiều công sức.
- Để giải phóng lập trình viên khỏi các suy tư về giới hạn của bộ nhớ, mà cũng không tăng thêm khó khăn cho công việc lập trình của họ, người ta nghĩ đến các kỹ thuật tự động, cho phép xử lý một chương trình có kích thước lớn chỉ với một vùng nhớ có kích thước nhỏ hơn. Giải pháp được tìm thấy với khái niệm bộ nhớ ảo (virtual memory).



Định nghĩa

- Bộ nhớ ảo là một kỹ thuật cho phép xử lý một tiến trình không được nạp toàn bộ vào bộ nhớ vật lý. Bộ nhớ ảo mô hình hóa bộ nhớ như một bảng lưu trữ rất lớn và đồng nhất, tách biệt hẳn khái niệm không gian địa chỉ và không gian vật lý. Người sử dụng chỉ nhìn thấy và làm việc trong không gian địa chỉ ảo, việc chuyển đổi sang không gian vật lý do hệ điều hành thực hiện với sự trợ giúp của các cơ chế phần cứng cụ thể.

Cài đặt bộ nhớ ảo

Bộ nhớ ảo thường được thực hiện với kỹ thuật phân trang theo yêu cầu (demand paging). Cũng có thể sử dụng kỹ thuật phân đoạn theo yêu cầu (demand segmentation) để cài đặt bộ nhớ ảo, tuy nhiên việc cấp phát và thay thế các phân đoạn phức tạp hơn thao tác trên trang, vì kích thước không bằng nhau của các đoạn.

- Phân trang theo yêu cầu (demand paging)

Một hệ thống phân trang theo yêu cầu là hệ thống sử dụng kỹ thuật phân trang kết hợp với kỹ thuật swapping. Một tiến trình được xem như một tập các trang, thường trú trên bộ nhớ phụ (thường là đĩa từ). Khi cần xử lý, tiến trình sẽ được nạp vào bộ nhớ chính. Nhưng thay vì nạp toàn bộ chương trình, chỉ những trang cần thiết trong thời điểm hiện tại mới được nạp vào bộ nhớ. Như vậy một trang chỉ được nạp vào bộ nhớ chính khi có yêu cầu.

- Với mô hình này, cần cung cấp một cơ chế phần cứng giúp phân biệt các trang đang ở trong bộ nhớ chính và các trang trên đĩa. Có thể sử dụng lại bit valid-invalid nhưng ngữ nghĩa mới:
 - Valid: trang tương ứng là hợp lệ và đang ở trong bộ nhớ chính.
 - Invalid: hoặc trang bất hợp lệ hoặc trang hợp lệ nhưng đang được lưu trên bộ nhớ phụ.
- Một phần tử trong bảng trang mô tả cho một trang không nằm trong bộ nhớ chính, sẽ được đánh dấu invalid và chứa địa chỉ của trang trên bộ nhớ phụ.

- Cơ chế phần cứng:

Cơ chế phần cứng hỗ trợ kỹ thuật phân trang theo yêu cầu là sự kết hợp của cơ chế hỗ trợ kỹ thuật phân trang và kỹ thuật swapping:

- Bảng trang: Cấu trúc bảng trang phải cho phép phản ánh tình trạng của một trang là đang nằm trong bộ nhớ chính hay bộ nhớ phụ.
- Bộ nhớ phụ: Bộ nhớ phụ lưu trữ những trang không được nạp vào bộ nhớ chính. Bộ nhớ phụ thường được sử dụng là đĩa, và vùng không gian đĩa dùng để lưu trữ tạm các trang trong kỹ thuật swapping được gọi là không gian swapping.

Lỗi trang

- Truy xuất đến một trang được đánh dấu bất hợp lệ sẽ làm phát sinh một lỗi trang (page fault). Khi dò tìm trong bảng trang để lấy các thông tin cần thiết cho việc chuyển đổi địa chỉ, nếu nhận thấy trang đang được yêu cầu truy xuất là bất hợp lệ, cơ chế phần cứng sẽ phát sinh một ngắt để báo cho hệ điều hành.
- Hệ điều hành sẽ xử lý lỗi trang như sau:

Bước 1: Kiểm tra truy xuất đến bộ nhớ có hợp lệ ?

Bước 2: Nếu truy xuất bất hợp lệ: kết thúc tiến trình; ngược lại: đến bước 3.

Bước 3: Tìm vị trí chứa trang muốn truy xuất trên đĩa.

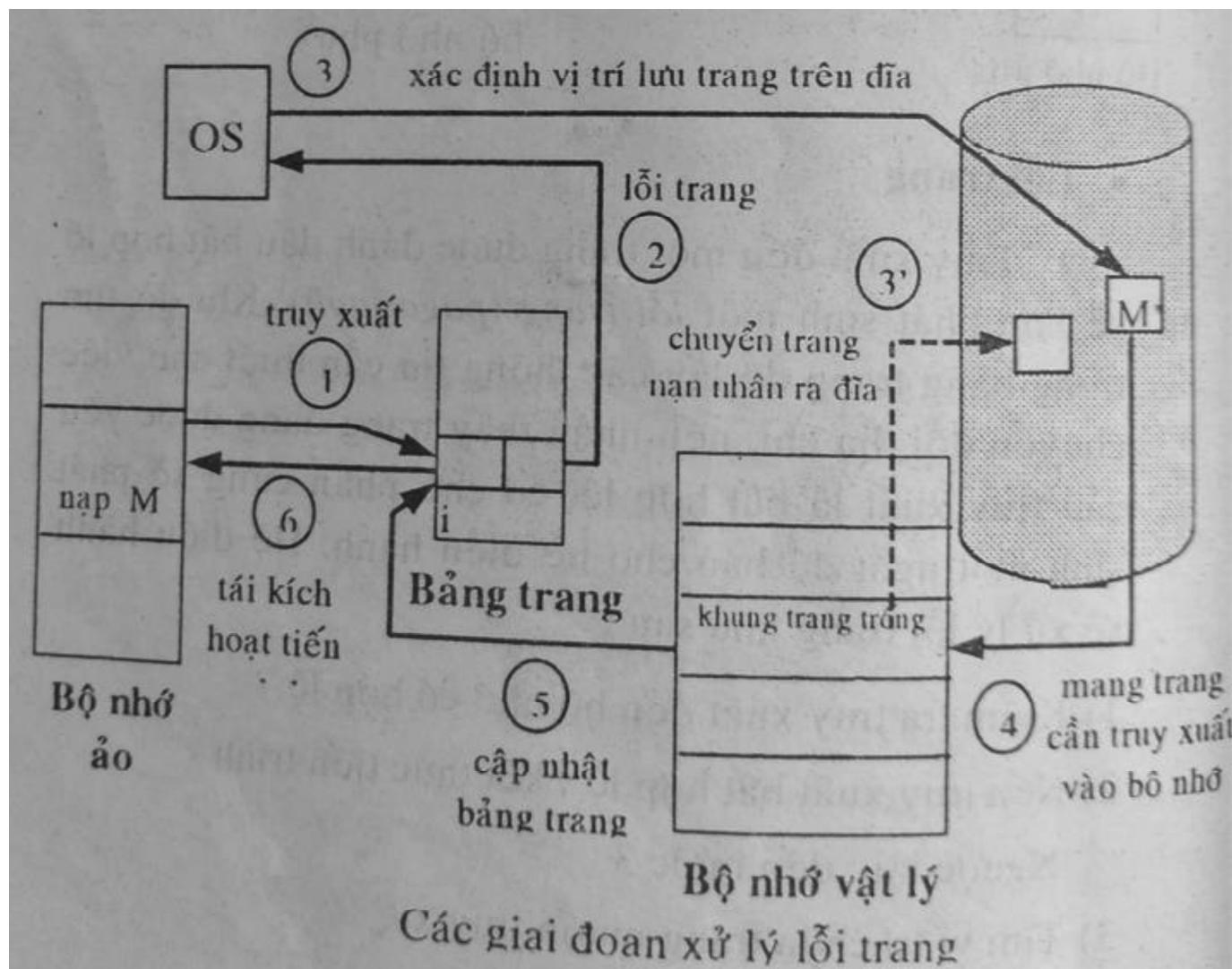
Bước 4: Tìm một khung trang trống trong bộ nhớ chính:

a.Nếu tìm thấy: đến bước 5

b.Nếu không còn khung trang trống, chọn một khung trang “nạn nhân” và chuyển trang “nạn nhân” ra bộ nhớ phụ (lưu nội dung của trang đang chiếm giữ khung trang này lên đĩa), cập nhật bảng trang tương ứng rồi đến bước 5.

Bước 5: Chuyển trang muốn truy xuất từ bộ nhớ phụ vào bộ nhớ chính: nạp trang cần truy xuất vào khung trang trống đã chọn (hay vừa mới làm trống); cập nhật nội dung bảng trang, bảng khung trang tương ứng.

Bước 6: Tái kích hoạt tiến trình người sử dụng.



Các thuật toán thay thế trang

Thay thế trang

- Khi xảy ra một lỗi trang, cần phải mang trang vắng mặt vào bộ nhớ. Nếu không có một khung trang nào trống, hệ điều hành cần thực hiện công việc thay thế trang – chọn một trang đang nằm trong bộ nhớ mà không được sử dụng tại thời điểm hiện tại và chuyển nó ra không gian swapping trên đĩa để giải phóng một khung trang dành chỗ nạp trang cần truy xuất vào bộ nhớ.
- Sự thay thế trang là cần thiết cho kỹ thuật phân trang theo yêu cầu. Nhờ cơ chế này, hệ thống có thể hoàn toàn tách rời bộ nhớ ảo và bộ nhớ vật lý, cung cấp cho lập trình viên một bộ nhớ ảo rất lớn trên một bộ nhớ vật lý có thể bé hơn rất nhiều lần.

Để duy trì ở một mức độ chấp nhận được sự chậm trễ trong hoạt động của hệ thống do phân trang, cần phải duy trì tỉ lệ phát sinh lỗi trang thấp.

Hơn nữa, để cài đặt kỹ thuật phân trang theo yêu cầu, cần phải giải quyết hai vấn đề chính yếu: xây dựng một thuật toán cấp phát khung trang, và thuật toán thay thế trang.

- Vấn đề chính khi thay thế trang là chọn lựa một trang “nạn nhân” để chuyển ra bộ nhớ phụ.
- Có nhiều thuật toán thay thế trang khác nhau, nhưng tất cả cùng chung một mục tiêu: chọn trang “nạn nhân” là trang mà sau khi thay thế sẽ gây ra ít lỗi trang nhất.
- Có thể đánh giá hiệu quả của một thuật toán bằng cách xử lý trên một chuỗi các địa chỉ cần truy xuất và tính toán số lượng lỗi trang phát sinh.
- Để minh họa các thuật toán thay thế trang sẽ trình bày, chuỗi truy xuất được sử dụng là:

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

Thuật toán FIFO

Nguyên tắc:

Ghi nhận thời điểm một trang vào bộ nhớ chính.
Khi cần thay thế trang, trang ở trong bộ nhớ lâu nhất sẽ được chọn.

Ví dụ 4a.

- Sử dụng 3 khung trang, ban đầu cả 3 khung trang đều trống.

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2	2	2	2	4	4	4	0	0	0	0	0	0	0	7	7	7
0	0	0	0	3	3	3	2	2	2	2	2	1	1	1	1	1	0	0	
1	1	1	1	0	0	0	3	3	3	3	3	2	2	2	2	2	1		
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	

Ghi chú: *: có lỗi trang

Thảo luận

- Để áp dụng thuật toán FIFO, thực tế không nhất thiết phải ghi nhận thời điểm mỗi trang được nạp vào bộ nhớ, mà chỉ cần tổ chức quản lý các trang trong bộ nhớ trong một danh sách FIFO, khi đó trang đầu danh sách sẽ được chọn để thay thế.
- Thuật toán thay thế trang FIFO dễ cài đặt. Tuy nhiên khi thực hiện không phải lúc nào cũng có kết quả tốt: trang được chọn để thay thế có thể là trang chứa nhiều dữ liệu cần thiết, thường xuyên được sử dụng nên được nạp sớm, do vậy khi được chuyển ra bộ nhớ phụ sẽ nhanh chóng gây ra lỗi trang.
- Số lượng lỗi trang xảy ra sẽ tăng lên khi số lượng khung trang sử dụng tăng. Hiện tượng này gọi là nghịch lý Belady.

Ví dụ 4b.

Chuỗi truy xuất: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5 khi sử dụng 3 khung trang

1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	4	4	4	5	5	5	5	5	5
2	2	2	1	1	1	1	1	1	3	3	3
3	3	3	2	2	2	2	2	2	4	4	
*	*	*	*	*	*	*		*	*		

có 9 lỗi phát sinh

Ví dụ 4c.

Chuỗi truy xuất: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5 khi sử dụng 4 khung trang

1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	1	1	1	5	5	5	5	4	4
2	2	2	2	2	2	1	1	1	1	1	5
3	3	3	3	3	3	3	2	2	2	2	2
	4	4	4	4	4	4	4	3	3	3	3
*	*	*	*		*	*	*	*	*	*	*

có 10 lỗi phát sinh

Thuật toán tối ưu

Nguyên tắc:

Thay thế trang sẽ lâu được sử dụng nhất trong tương lai.

Ví dụ 5.

Sử dụng 3 khung trang, khởi đầu đều trống

Thảo luận

- Thuật toán này bảo đảm số lượng lỗi trang phát sinh là thấp nhất, nó cũng không gánh chịu nghịch lý Belady.
- Đây là một thuật toán không khả thi trong thực tế, vì không thể biết trước chuỗi truy xuất của tiến trình.

Thuật toán Least recently used - LRU

Nguyên tắc:

Với mỗi trang, ghi nhận thời điểm cuối cùng trang được truy cập, trang được chọn để thay thế sẽ là trang lâu nhất chưa được truy xuất.

Ví dụ 6.

- Sử dụng 3 khung trang, khởi đầu đều trống:

Thảo luận

- Thuật toán FIFO sử dụng thời điểm nạp để chọn trang thay thế, thuật toán tối ưu lại dùng thời điểm trang sẽ sử dụng, vì thời điểm này không thể xác định trước nên thuật toán LRU phải dùng thời điểm cuối cùng trang được truy xuất; dùng quá khứ gần để dự đoán tương lai.
- Thuật toán này đòi hỏi phải được cơ chế phần cứng hỗ trợ để xác định một thứ tự cho các trang theo thời điểm truy xuất cuối cùng. Có thể cài đặt theo một trong hai cách:
 - Sử dụng bộ đếm
 - Sử dụng stack

Bài tập chương 3

Bài tập 1

Giả sử bộ nhớ chính được phân thành các phân vùng có kích thước là 600K, 500K, 200K, 300K (theo thứ tự), cho biết các tiến trình có kích thước 212K, 417K, 112K và 426K (theo thứ tự) sẽ được cấp phát bộ nhớ như thế nào, nếu sử dụng :

- a) Thuật toán First fit
- b) Thuật toán Best fit
- c) Thuật toán Worst fit

Thuật toán nào cho phép sử dụng bộ nhớ hiệu quả nhất trong trường hợp trên ?

Bài tập 2

a.

Hệ thống phân trang, kích thước trang là 1024B. Bảng phân trang hiện thời như sau (page, frame):

3 0

2 4

1 x

0 1

Hãy tính địa chỉ vật lý của các địa chỉ logic sau:

- 1020
- 2060

b.

Cho bảng phân trang page 0,1,2,3; frame tương ứng là 3,2,6,4. Với kích thước mỗi trang là 1KB thì địa chỉ logic 2021 ứng với địa chỉ vật lý nào ?

Tóm tắt:

lage

0

1

2

3

Frame

3

2

6

9

Địa chỉ logic: 1021 1KB

1KB = 1024 byte

1021 div 1024 = 1 \Rightarrow Frame: 2

Địa chỉ vật lý: $2 \times 1024 + (1021 \% 1024)$

= 3045

(Day địa chỉ vật lý là 3045)

Bài tập 3

a.

Cho địa chỉ vật lý là 4100 sẽ được chuyển thành địa chỉ ảo bao nhiêu? Biết rằng kích thước mỗi frame là 1K bytes, và bảng ánh xạ địa chỉ ảo như hình.

0	6
1	4
2	5
3	7
4	1
5	9

Page Table

Hướng dẫn tham khảo:

- Địa chỉ vật lý 4100 nằm ở khung trang: $4100/1024$ (kích thước của khung trang)=1.
- Tra vào bảng trang ta có: trang địa chỉ logic nằm ở trang 1 (1024).
- Trường hợp ta đang xét là địa chỉ vật lý nằm ngay ở đầu frame số 4, nghĩa là cách frame 4: $4100 - 4096 = 4$. Vậy trong địa chỉ logic nó cũng cách địa chỉ của page số 1 là 4.
- Vậy ta có thể kết luận được địa chỉ logic ở đây là $1*1024 + 4=1028$.

b.

Cho địa chỉ vật lý là 4604 sẽ được chuyển thành địa chỉ logic bao nhiêu? Biết rằng kích thước mỗi frame là 1K bytes và bảng ánh xạ địa chỉ logic như hình sau:

0	6
1	4
2	5
3	7
4	1
5	9

Page Table

Bài tập 4 (thuật toán thay thế trang)

Xem chuỗi truy xuất bộ nhớ sau:

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

Có bao nhiêu lỗi trang xảy ra khi sử dụng các thuật toán thay thế sau đây, giả sử có 4 khung trang ?

- a) LRU
- b) FIFO
- c) Chiến lược tối ưu

Bài tập 5 (thuật toán thay thế trang)

Cho chuỗi truy xuất 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0 khi sử dụng thuật toán thay thế trang Optimal với 3 khung trang thì sẽ có bao nhiêu lỗi trang phát sinh ?

6a. Bài tập phân đoạn, tính địa chỉ vật lý cho địa chỉ logic

Giả sử trong quá trình quản lý bộ nhớ ảo dạng phân đoạn, hệ điều hành duy trì segment table

Segment	base	limit
0	300	700
1	1200	500
2	2000	600

Hãy tính địa chỉ vật lý cho mỗi địa chỉ logic sau:

(1,200); (1,0); (0,700); (2,0); (2,600)

6b. Bài tập phân đoạn, tính địa chỉ vật lý cho địa chỉ logic có trường hợp không hợp lệ

Giả sử có bảng đoạn như sau:

Segment	base	limit
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

Hãy tính địa chỉ vật lý cho mỗi địa chỉ lô-gic sau:

0430; 1010; 2500; 3400; 4112

Bài tập 7

Cài đặt các thuật toán đã được trình bày trong chương này.