Chương 5

BỘ NHỚ ẢO

(Virtual Memory)

Nội Dung Chương V

I. Khái niệm bộ nhớ ảo
II. Cơ chế thực hiện bộ nhớ ảo: phân trang theo yêu cầu
III. Thuật toán thay thế trang
IV. Cấp phát khung trang
V. Tình trạng trì trệ hệ thống

I. Dẫn Nhập

Ràng buộc về kích thước bộ nhớ các tiến trình:

Có 2 tiến trình A, B với kích thước bộ nhớ logic là 300 và 400.

Kích thước bộ nhớ vật lý: 600

→ Làm thế nào để A, B thực hiện cùng lúc?

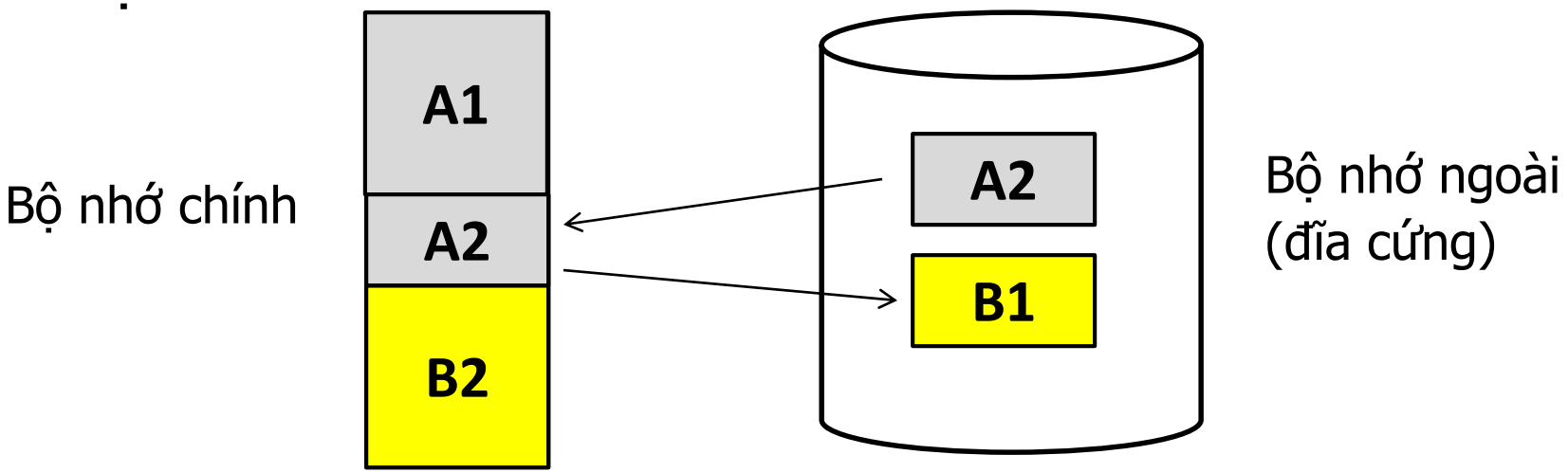
- Nhân xét: Trong bộ nhớ của tiến trình: có những đoạn được sử dụng thường xuyên, có những đoạn ít hoặc không được sử dụng

  Ví dụ:

  - Các đoạn lệnh xử lý lỗi.
  - Các biến chỉ sử dụng 1 lần.

1) Khái niệm bộ nhớ ảo:

Đối với 1 tiến trình, có thể swap những đoạn bộ nhớ ít được sử dụng ra đĩa cứng. Khi nào cần sử dụng sẽ nạp lại vào bộ nhớ.



Tại sao gọi là 'bộ nhớ ảo'?

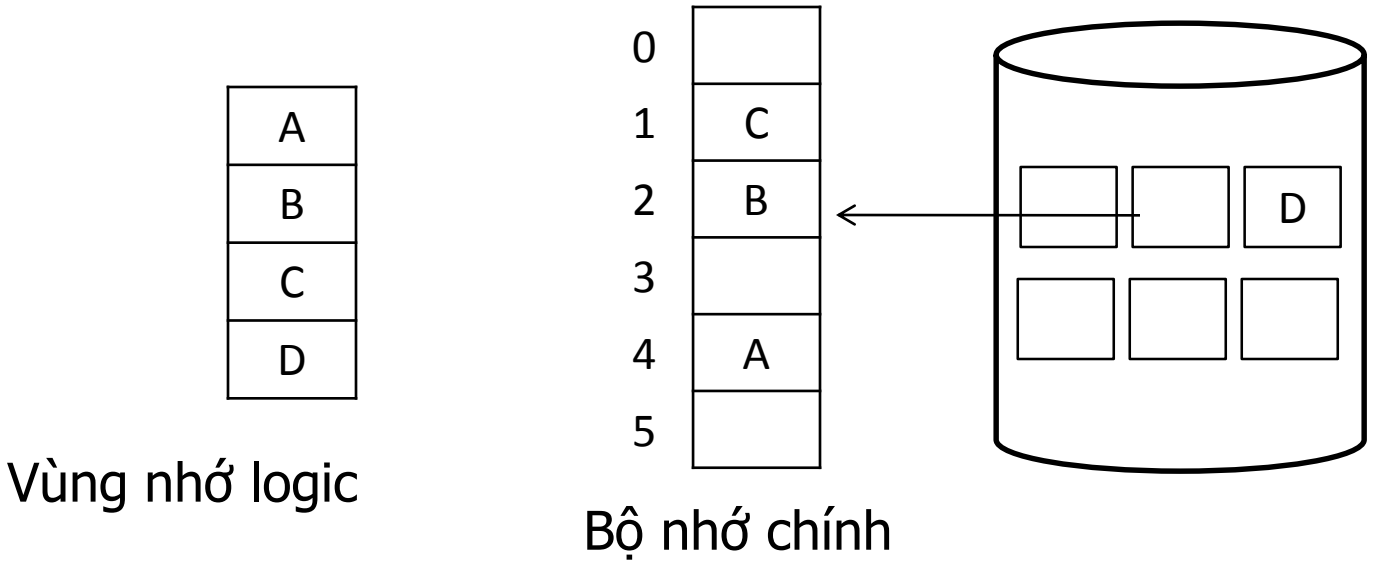
→ Vì đĩa cứng được sử dụng giả lập thành bộ nhớ chính.

4) Ích lợi của bộ nhớ ảo

- Kích thước logic một chương trình không bị giới hạn bởi kích thước vật lý của bộ nhớ.
- Nhiều tiến trình có thể thực hiện song song.

II. Phân trang theo yêu cầu

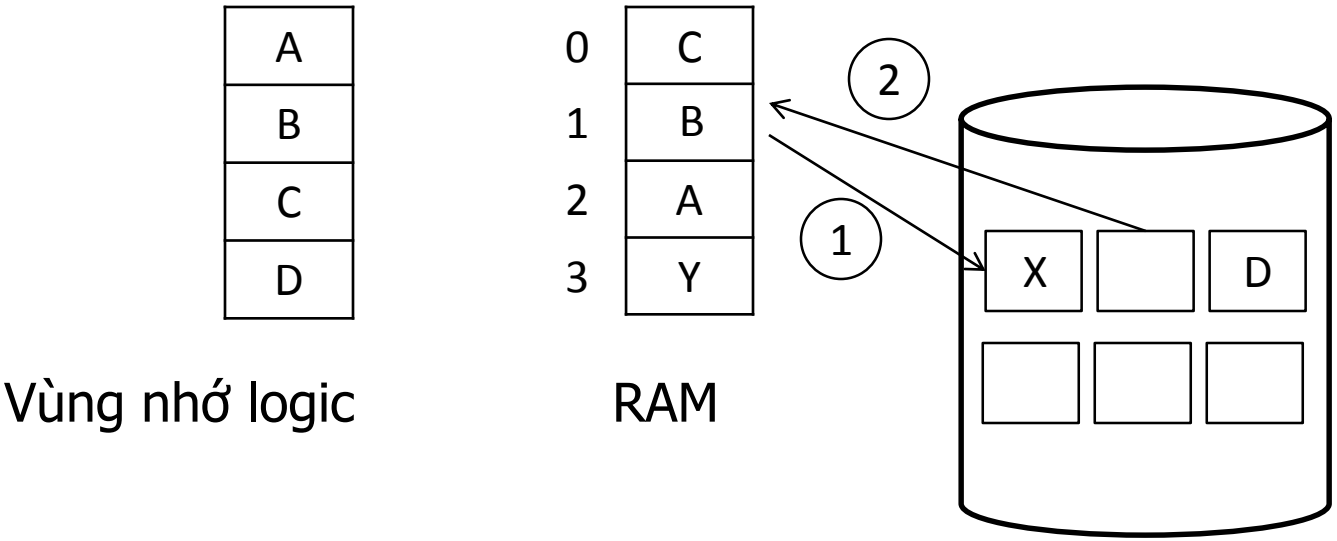
1) Thực hiện bộ nhớ ảo theo mô hình phân trang:
   - Những trang cần thiết của tiến trình được nạp vào bộ nhớ chính.
   - Các trang còn lại để trên bộ nhớ ngoài, chỉ nạp vào bộ nhớ khi có nhu cầu (demand paging)



2) Thay thế trang

Nếu trong bộ nhớ không còn khung trang trống để nạp trang mới, cần chọn một trang để swap ra bộ nhớ ảo để tạo trang trống.

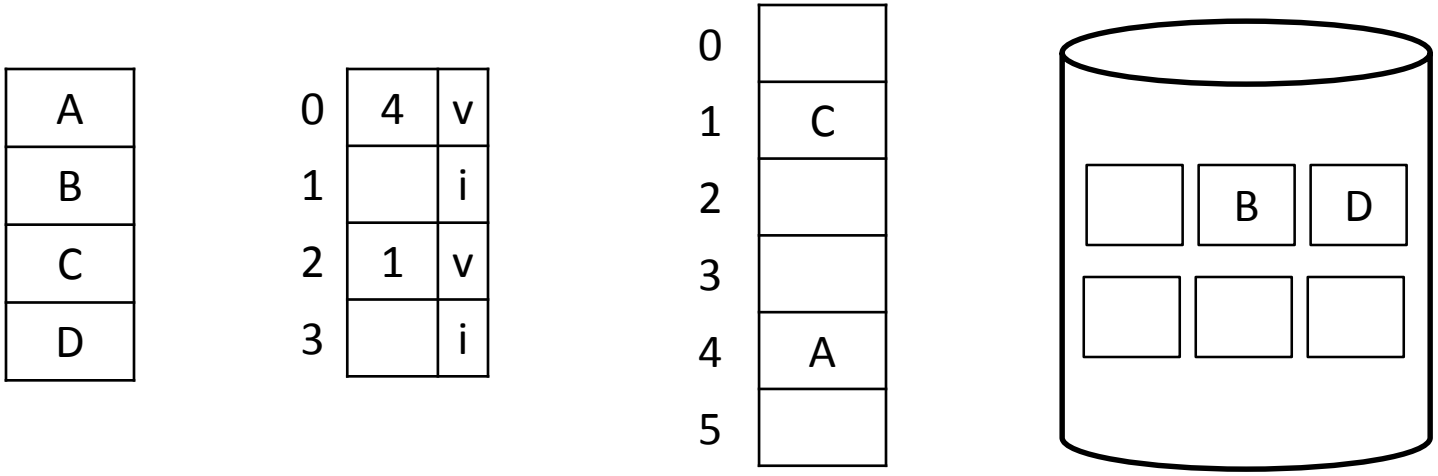
Ví dụ: cần trang B



Trang X được chuyển ra đĩa gọi là trang "nạn nhân"

3) Cơ chế MMU trong quản lý bộ nhớ ảo:

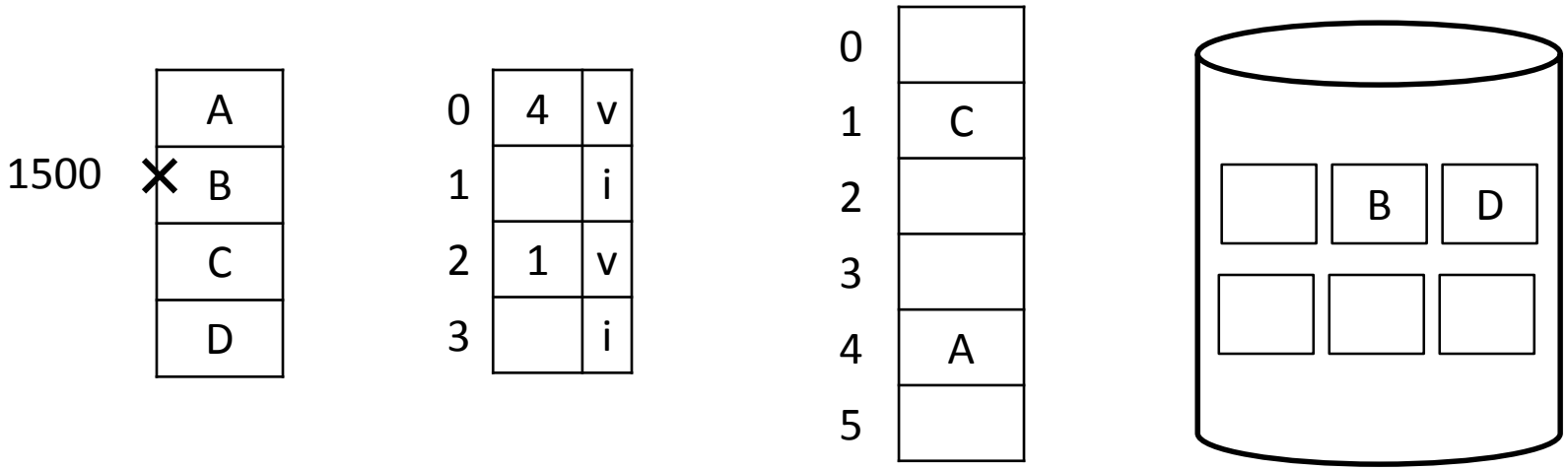
- Mỗi phần tử bảng trang có thêm một bít valid - invalid.
  - Nếu bít là valid: nội dung trang đang ở bộ nhớ chính
  - Nếu bít là invalid: nội dung trang ở bộ nhớ ngoài.



4) Lỗi trang:

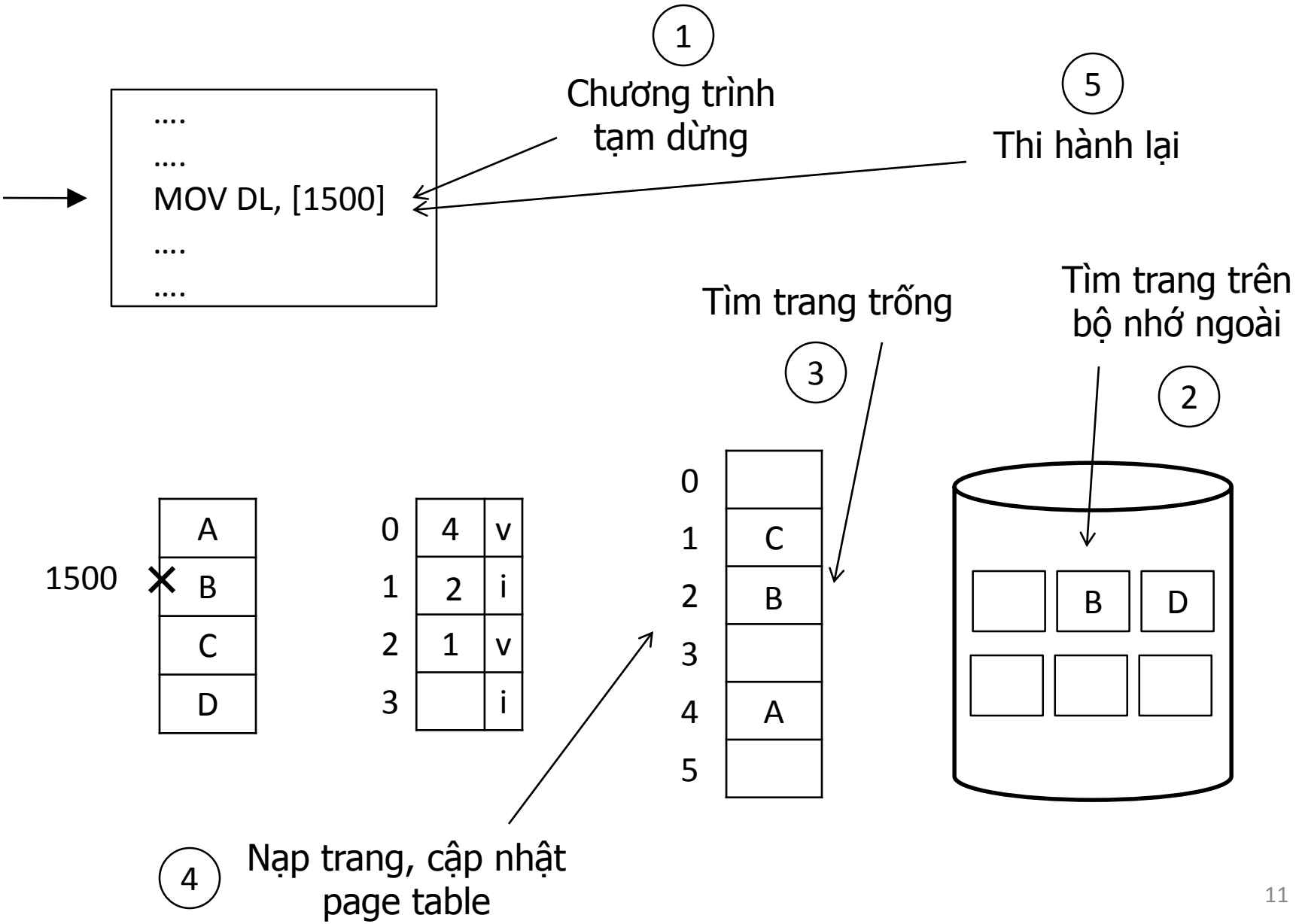
Xảy ra khi CPU cần sử dụng một trang, nhưng trang đang ở bộ nhớ ngoài.

ví dụ lệnh: MOV DL, [1500] kích thước trang 1024



Xử lý lỗi trang: cần nạp lại trang vào bộ nhớ chính

Các bước xử lý lỗi trang:



Các bước xử lý lỗi trang:

1) Chương trình tạm dừng
2) Hệ điều hành tìm trang cần thiết trên bộ nhớ ngoài
3) Hệ điều hành tìm trang trống trong bộ nhớ chính (Nếu không có, chọn một trang "nạn nhân" chuyển ra bộ nhớ ngoài để tạo trang trống)
4) Nạp trang từ bộ nhớ ngoài vào bộ nhớ chính, cập nhật page table
5) Thi hành lại chương trình

4) Thời gian truy xuất bộ nhớ khi có lỗi trang:

Ký hiệu:

- t_{in}: thời gian tìm và nạp trang từ đĩa vào trang trống, cập nhật bảng trang.
- t_{out}: thời gian tìm và chuyển trang nạn nhân ra đĩa
- tdp: tổng thời gian xử lý một lỗi trang

Nếu trong bộ nhớ có sẵn trang trống:

$tdp = t_{in}$

Nếu trong bộ nhớ không có trang trống:

$tdp = t_{out} + t_{in}$

4) Thời gian truy xuất bộ nhớ trung bình:

Ký hiệu:

- ma: thời gian truy xuất bộ nhớ chính của 1 lệnh.
- p: tỷ lệ xảy ra lỗi trang
- tdp: thời gian xử lý một lỗi trang trung bình

→ Thời gian truy xuất bộ nhớ trung bình TEA:

$$\mathrm{TEA} = (1-\mathrm{p}) \times \mathrm{ma} + \mathrm{p} \times (\mathrm{tdp}+\mathrm{ma})$$

Vì tdp >> ma, nên để giảm TEA cần giảm tỷ lệ xảy ra lỗi trang p.

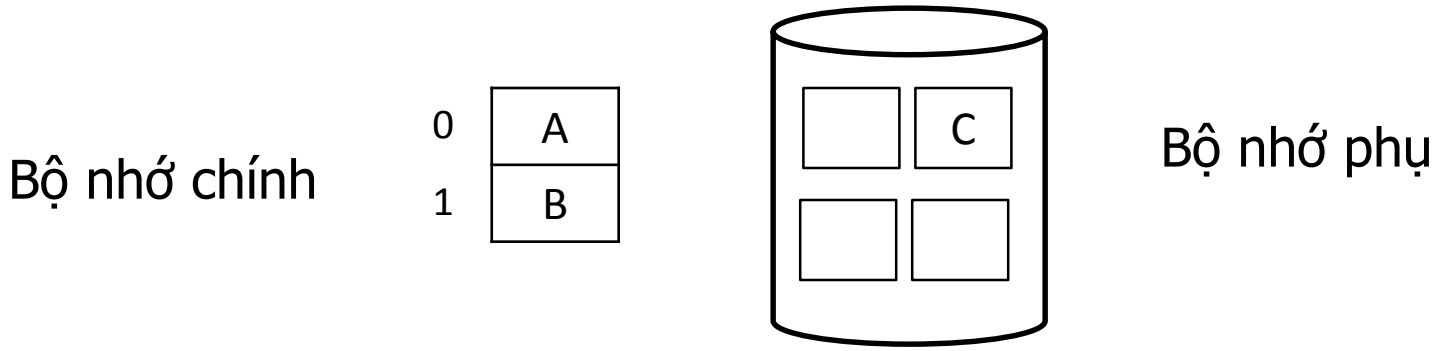
II. Chiến lược thay thế trang

1) Vấn đề chọn trang nạn nhân

Cần chọn trang nào trong bộ nhớ làm trang nạn nhân?

Chọn trang ngẫu nhiên bất kỳ? Chọn theo thuật toán?

Ví dụ: tiến trình có 3 trang logic A,B,C. Bộ nhớ chính chỉ có 2 khung trang



Giả sử thứ tự truy xuất trang là C, A. Cần chọn trang nào làm trang nạn nhân?

Nếu chọn A: xảy ra 2 lỗi trang.

Nếu chọn B: xảy ra 1 lỗi trang.

Tiêu chí để chọn trang nạn nhân: "các truy xuất bộ nhớ *tiếp theo* gây ít ra lỗi trang nhất"

2) Khái niệm chuỗi truy xuất trang

- Một tiến trình cần truy xuất các địa chỉ bộ nhớ theo thứ tự:

  0100, 0432, 0101, 0162, 0611, 0609, 0102, 0610

- Kích thước trang: 100
- Chuỗi truy xuất viết lại theo trang: 1 4 1 1 6 6 1 6

3) Các thuật toán thay thế trang

a) Thuật toán FIFO

Trong các trang đang ở trong bộ nhớ, chọn trang được nạp vào bộ nhớ trước nhất để thay thế.

Ví dụ: Chuỗi truy xuất trang sau với 3 frame bộ nhớ

	6	1	0	2	1	6	1	0	1	2
8 Lỗi trang →	*	*	*	*		*	*	*		*

(6)	6	6	(2)	2	2	2	(0)	0	0
	(1)	1	1	1	(6)	6	6	6	(2)
		(0)	0	0	0	(1)	1	1	1

Câu hỏi: Số lỗi trang của FIFO có ít nhất?

b) Thuật toán tối ưu OPT

Chọn trang sẽ lâu được sử dụng nhất trong tương lai để thay thế.

Ví dụ: Xét lại chuỗi truy xuất trên

	6	1	0	2	1	6	1	0	1	2
5 Lỗi trang →	*	*	*	*				*		

(6)	6	6	6	6	6	6	(0)	0	0
	(1)	1	1	1	1	1	1	1	1
		(0)	(2)	2	2	2	2	2	2

Tuy nhiên không khả thi vì không biết trước được chuỗi truy xuất.

c) Thuật toán "lâu nhất chưa sử dụng"

(Least Recently Used - LRU)

Chọn trang lâu nhất chưa được sử dụng để thay thế (với hy vọng trang này cũng sẽ chưa được sử dụng trong tương lai gần, còn những trang mới sử dụng thì tiếp tục được sử dụng → tính locality).

Ví dụ: Xét lại chuỗi truy xuất trên

	6	1	0	2	1	6	1	0	1	2
7 Lỗi trang →	*	*	*	*		*		*		*

(6)	6	6	(2)	2	2	2	(0)	0	0
	(1)	1	1	1	1	1	1	1	1
		(0)	0	0	(6)	6	6	6	(2)

LRU được sử dụng phổ biến trong thực tế.

Cơ chế thực hiện LRU: Rất tốn kém → Dùng các thuật toán xấp xỉ LRU.

d) Xấp xỉ LRU: Thuật toán cơ hội thứ 2 (second chance)

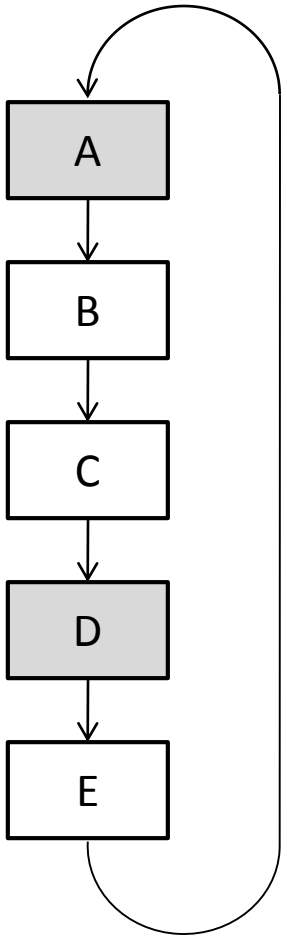
- Ý tưởng: Giống như thuật toán FIFO, tuy nhiên mỗi trang sẽ có thêm một "cơ hội" (tức vẫn được ở lại bộ nhớ chưa bị thay ra).
- Chi tiết: Các trang được chia làm hai nhóm:
  - Nhóm vừa được sử dụng (nhóm α)
  - Nhóm đã được cấp cơ hội thứ 2 (nhóm β)

Khi cần tìm trang thay thế, các trang được duyệt theo thứ tự FIFO. Nếu trang đang xét thuộc nhóm β → thay ra. Nếu trang đang xét thuộc nhóm α → chuyển xuống nhóm β (có thêm cơ hội thứ 2).

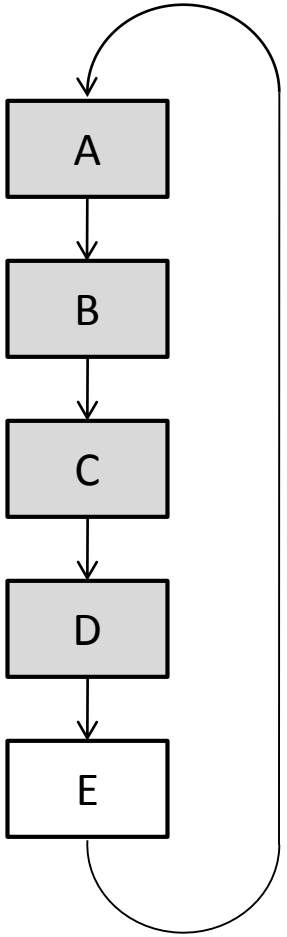
Một trang vừa sử dụng xong, nếu thuộc nhóm β thì được chuyển lại sang nhóm α

Nhóm α Nhóm β

Trang đang xét →



Trang nạn nhân →



Danh sách vòng các trang theo thứ tự vào bộ nhớ

III. Tình trạng trì trệ tiến trình (thrashing)

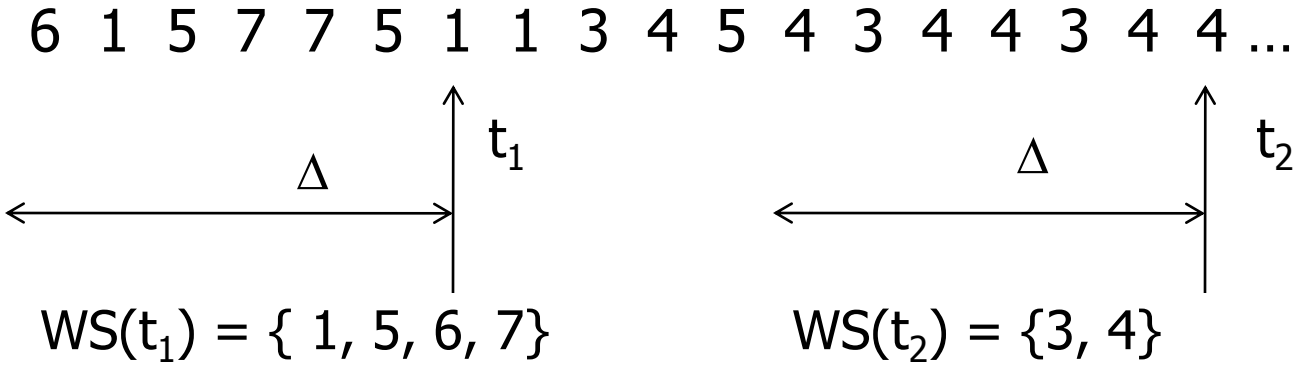
Vấn đề: bộ nhớ ảo cho phép thực hiện nhiều tiến trình cùng lúc không phụ thuộc vào kích thước bộ nhớ vật lý → nhiều là bao nhiêu?

1) Khái niệm trì trệ tiến trình
   - Gọi p là tỉ lệ lỗi trang của tiến trình P1.
   - Nếu p lớn (do nhiều tiến trình cùng thực hiện) → P1 tốn nhiều thời gian xử lý lỗi trang hơn là thời gian thi hành lệnh → P1 bị trì trệ.
   - Giải quyết: cung cấp số lượng frame bộ nhớ phù hợp cho P1 để số lỗi trang *p* là nhỏ.

2) Tính toán số lượng frame phù hợp

Dựa trên tính locality: trong một khoảng thời gian Δ , một tiến trình chỉ sử dụng một số trang xác định (WS).

Ví dụ: $\Delta = 6$



Các trang này được gọi là tập làm việc (working-set)

Nếu cấp cho tiến trình số frame ≈ số lượng trang trong working-set:

- Số lỗi trang sẽ nhỏ.
- Số lượng frame cấp phát cho tiến trình có thể thay đổi theo thời gian.

IV. Cấp phát khung trang

1) Khái niệm cấp phát khung trang:

Cấp phát khung trang là quyết định số khung trang bộ nhớ chính cần thiết cấp cho một tiến trình sử dụng

Ví dụ:

- Bộ nhớ có 100 frame
- Tiến trình P1 gồm 200 page, P2 gồm 300 page
- Cần cấp cho P1, P2 bao nhiêu frame?

2) Cấp trang với số lượng cố định

a) Cấp bằng nhau

Nếu có bộ nhớ có m khung trang và n tiến trình, mỗi tiến trình được cấp số trang là m/n.

Ví dụ: P1: 50 frame, P2: 50 frame

b) Cấp theo tỷ lệ kích thước tiến trình

Tiến trình có kích thước bộ nhớ logic lớn thì được cấp nhiều khung trang hơn.

Ví dụ: P1: 40 frame, P2: 60 frame

Khi P1 xảy ra lỗi trang, chỉ được chọn nạn nhân là trang thuộc P1, không được chọn trang của tiến trình khác → ***thay thế cục bộ***.

Việc chọn kích thước cố định → có thể xảy ra thrashing cho một tiến trình (trong khi tiến trình khác không sử dụng hết).

3) Cấp trang với số lượng thay đổi

- Mỗi tiến trình có thể được cấp các khung trang với số lượng thay đổi tùy theo thời gian.
- Khi tiến trình P1 xảy ra lỗi trang, có thể chọn trang của tiến trình khác làm trang nạn nhân → thay thế toàn cục.
- Số khung trang cấp cho tiến trình P_i tại một thời điểm là số trang trong working-set của P_i tại thời điểm đó

Tổng số khung trang cần thiết:

$$D = \sum WSS_i$$

WSS_i là kích thước working-set của P_i

Nếu kích thước bộ nhớ:

- < D: thiếu bộ nhớ → thrashing toàn bộ hệ thống → cần hủy bỏ một tiến trình để giảm D
- > D: dư bộ nhớ → có thể kích hoạt tiến trình khác.

Q & A

Câu hỏi ôn tập

1. Ý tưởng của bộ nhớ ảo là gì?
2. Với bộ nhớ ảo sử dụng mô hình phân trang, khi nào thì xảy ra lỗi trang? Nêu các bước thực hiện của hệ điều hành để xử lý lỗi trang.
3. Tiêu chí để chọn trang nạn nhân trong việc thay thế trang là gì?
4. Trong ba thuật toán thay thế trang FIFO, OPT, và LRU, thuật toán nào khả thi nhất? Vì sao?
5. Tình trạng trì trệ tiến trình là gì?
6. Nêu khái niệm working-set và tác dụng của working-set trong việc tránh tình trạng trì trệ tiến trình.

Bài tập

1) Giả sử một chuỗi truy xuất bộ nhớ có chiều dài là p với n số hiệu trang khác nhau xuất hiện trong chuỗi. Giả sử hệ thống sử dụng m khung trang (khởi động trống), n>m. Hãy cho biết:
   - Số lượng tối thiểu các lỗi trang xảy ra.
   - Số lượng tối đa các lỗi trang xảy ra.

2) Giả sử có một hệ thống sử dụng kỹ thuật phân trang theo yêu cầu. Bảng trang được lưu trữ trong các thanh ghi. Để xử lý một lỗi trang tốn 8 miligiây nếu có sẵn một khung trang trống, và tốn 20 miligiây nếu cần thay thế trang. Giả sử xác suất cần thay thế trang khi có lỗi trang là 70%. Mỗi truy xuất bộ nhớ tốn 100 nanogiây. Giả sử tỷ lệ phát sinh lỗi trang là 1% thì thời gian truy xuất bộ nhớ trung bình là bao nhiêu?

3) Bộ nhớ vật lý có 4 khung trang, thời điểm nạp và thời điểm truy cập cuối cùng được cho trong bảng sau:

Trang	*Thời điểm nạp*	*Thời điểm truy xuất*
0	126	279
1	230	260
2	120	272
3	160	280

Trang nào được chọn làm trang nạn nhân trong các thuật toán thay thế:

a) FIFO b) LRU

4) Xét chuỗi truy xuất bộ nhớ sau

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

Giả sử bộ nhớ vật lý có 4 khung trang. Minh họa quá trình thay thế trang với các thuật toán thay thế sau:

a) FIFO b) OPT c) LRU

5) Cho chương trình sau:

```
int A[100][100];
for (i=0; i<100; i++)
   for (j=0; j<100; j++) A[i][j]= 0;
```

Trong đó mảng A thuộc đoạn data và vòng lặp for thuộc đoạn code của chương trình. Đoạn code có kích thước là 200 byte, còn đoạn data có kích thước là 100x100x2 = 20.000 byte. Hai biến i, j là inline trong thanh ghi CPU.

Bộ nhớ máy tính được tổ chức theo mô hình phân trang với kích thước trang là 200 và có 3 khung trang. Đoạn code đã được nạp sẵn vào khung trang thứ nhất, 2 khung trang còn lại trống. Đoạn data đang ở trong bộ nhớ ảo.

Với thuật toán thay thế trang là LRU, hãy tính số lỗi trang phát sinh khi chạy chương trình trên.

6) Làm lại bài 5 nếu thay lệnh `a[i][j]=0` bằng lệnh `a[j][i]=0`