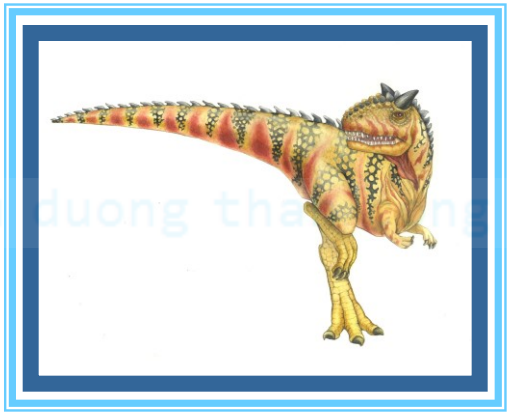
Chương 8: Bộ nhớ ảo



UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Câu hỏi ôn tập chương 7

- Bộ nhớ luận lý là gì? Bảng phân trang dùng để làm gì?
- Bảng trang được lưu trữ ở đâu? Các thanh ghi cần sử dụng trong cơ chế phân trang?
- TLB là gì? Dùng để làm gì?
- Thế nào là phân trang đa cấp? Cho ví dụ?
- Tại sao phải phân đoạn? Các đoạn được phân chia do cái gì?
- Các thanh ghi được sử dụng trong phân đoạn?

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Câu hỏi ôn tập chương 7

Xét một không gian địa chỉ có 14 trang, mỗi trang có kích thước 1MB. ánh xạ vào bộ nhớ vật lý có 38 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?

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Câu hỏi ôn tập chương 7

Xét một hệ thống sử dụng kỹ thuật phân trang, với bảng trang được lưu trữ trong bộ nhớ chính.

a) Nếu thời gian cho một lần truy xuất bộ nhớ bình thường là 124 nanoseconds, thì mất bao nhiêu thời gian cho một thao tác truy xuất bộ nhớ trong hệ thống này ?

b) Nếu sử dụng TLBs với hit-ratio (tỉ lệ tìm thấy) là 95%, thời gian để tìm trong TLBs bằng 34, tính thời gian cho một thao tác truy xuất bộ nhớ trong hệ thống (effective memory reference time)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Câu hỏi ôn tập chương 7

- Địa chỉ vật lý 6568 sẽ được chuyển thành địa chỉ ảo bao nhiêu? Biết rằng kích thước mỗi frame là 1K bytes
- Địa chỉ ảo 3254 sẽ được chuyển thành địa chỉ vật lý bao nhiêu? Biết rằng kích thước mỗi frame là 2K bytes

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

Bảng trang của P1

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Câu hỏi ôn tập chương 7

Xét một hệ thống sử dụng kỹ thuật phân trang, với bảng trang được lưu trữ trong bộ nhớ chính. Nếu sử dụng TLBs với hit-ratio (tỉ lệ tìm thấy) là 87%, thời gian để tìm trong TLBs là 24 nanosecond. Thời gian truy xuất bộ nhớ trong hệ thống (effective memory reference time) là 175. Tính thời gian cho một lần truy xuất bộ nhớ bình thường?

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Câu hỏi ôn tập chương 7

Biết thời gian truy xuất trong bộ nhớ thường không sử dụng TLB là 250ns. Thời gian tìm kiếm trong bảng TLB là 26ns. Hỏi sác xuất bằng bao nhiêu nếu thời gian truy xuất trong bộ nhớ chính là 182ns.

Câu hỏi ôn tập chương 7

Xét bảng phân đoạn sau đây :

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

Cho biết địa chỉ vật lý tương ứng với các địa chỉ logic sau đây :

a. 0,430 b. 1,100 c. 2,500 d. 3,400 e. 4,112

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Mục tiêu

- Hiểu được các khái niệm tổng quan về bộ nhớ ảo
- Hiểu và vận dụng các kỹ thuật cài đặt được bộ nhớ ảo:
  - Demand Paging
  - Page Replacement
  - Demand Segmentation
- Hiểu được một số vấn đề trong bộ nhở ảo
  - Frames
  - Thrashing

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

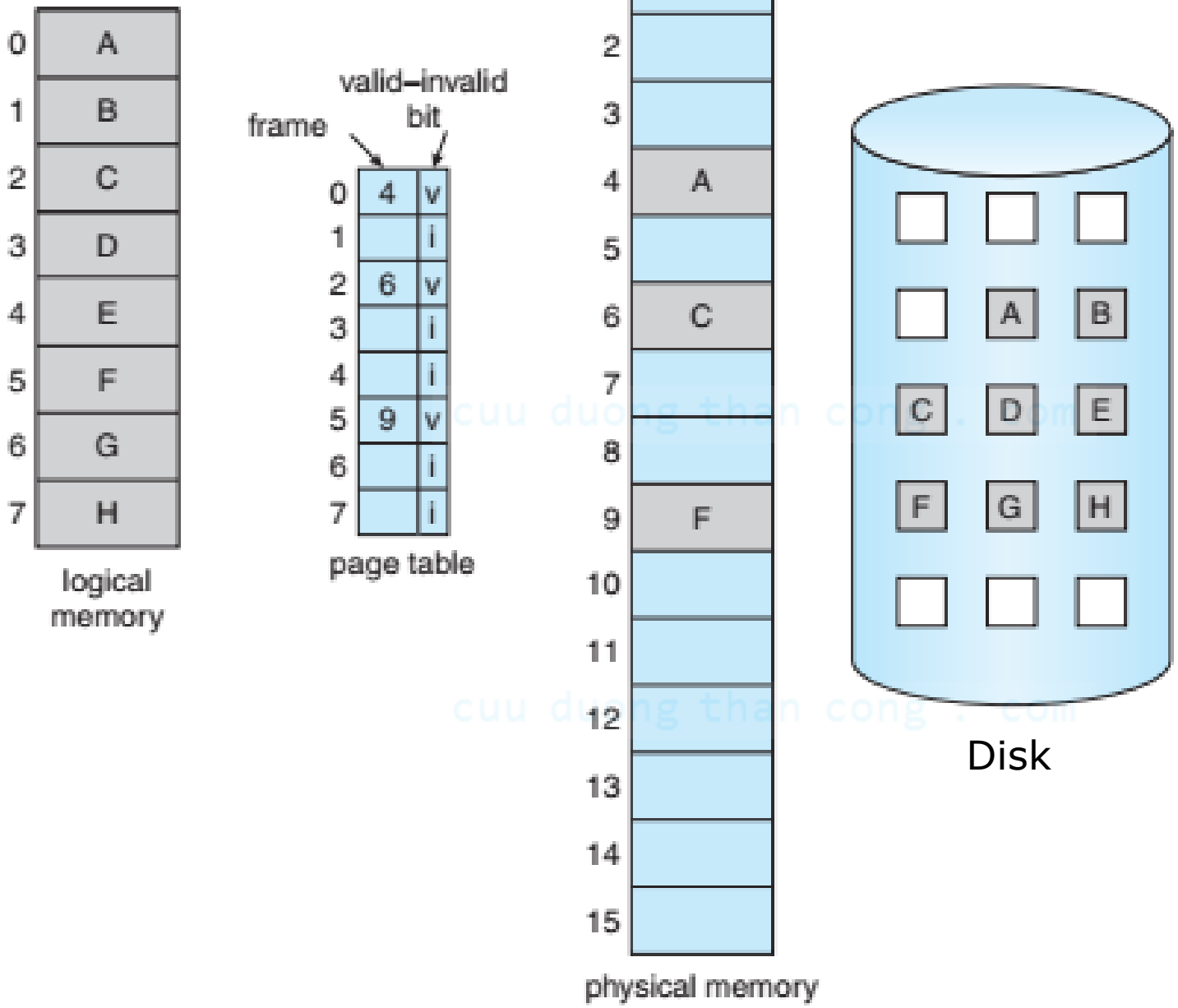
Nội dung

- Tổng quan về bộ nhớ ảo
- Cài đặt bộ nhớ ảo: Demand Paging
- Cài đặt bộ nhớ ảo: Page Replacement
  - Các giải thuật thay trang (Page Replacement Algorithms)
- Vấn đề cấp phát Frames
- Vấn đề Thrashing
- Cài đặt bộ bộ nhớ ảo: Demand Segmentation

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Tổng quan bộ nhớ ảo

- Nhận xét: không phải tất cả các phần của một process cần thiết phải được nạp vào bộ nhớ chính tại cùng một thời điểm
- Ví dụ:
  - Đoạn mã điều khiển các lỗi hiếm khi xảy ra
  - Các arrays, list, tables được cấp phát bộ nhớ (cấp phát tĩnh) nhiều hơn yêu cầu thực sự
  - Một số tính năng ít khi được dùng của một chương trình
  - Cả chương trình thì cũng có đoạn code chưa cần dùng
- Bộ nhớ ảo (virtual memory): Bộ nhớ ảo là một kỹ thuật cho phép xử lý một tiến trình không được nạp toàn bộ vào bộ nhớ vật lý



Logical memory có 8 pages, nhưng chỉ đang có 3 pages đang trong physical memory

Bộ nhớ ảo (tt)

- Ưu điểm của bộ nhớ ảo
  - Số lượng process trong bộ nhớ nhiều hơn
  - Một process có thể thực thi ngay cả khi kích thước của nó lớn hơn bộ nhớ thực
  - Giảm nhẹ công việc của lập trình viên (lập trình viên không phải lo về giới hạn memory khi lập trình)
- Không gian tráo đổi giữa bộ nhớ chính và bộ nhớ phụ (swap space).
- Ví dụ:
  - swap partition trong Linux
  - file pagefile.sys trong Windows

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Cài đặt bộ nhớ ảo

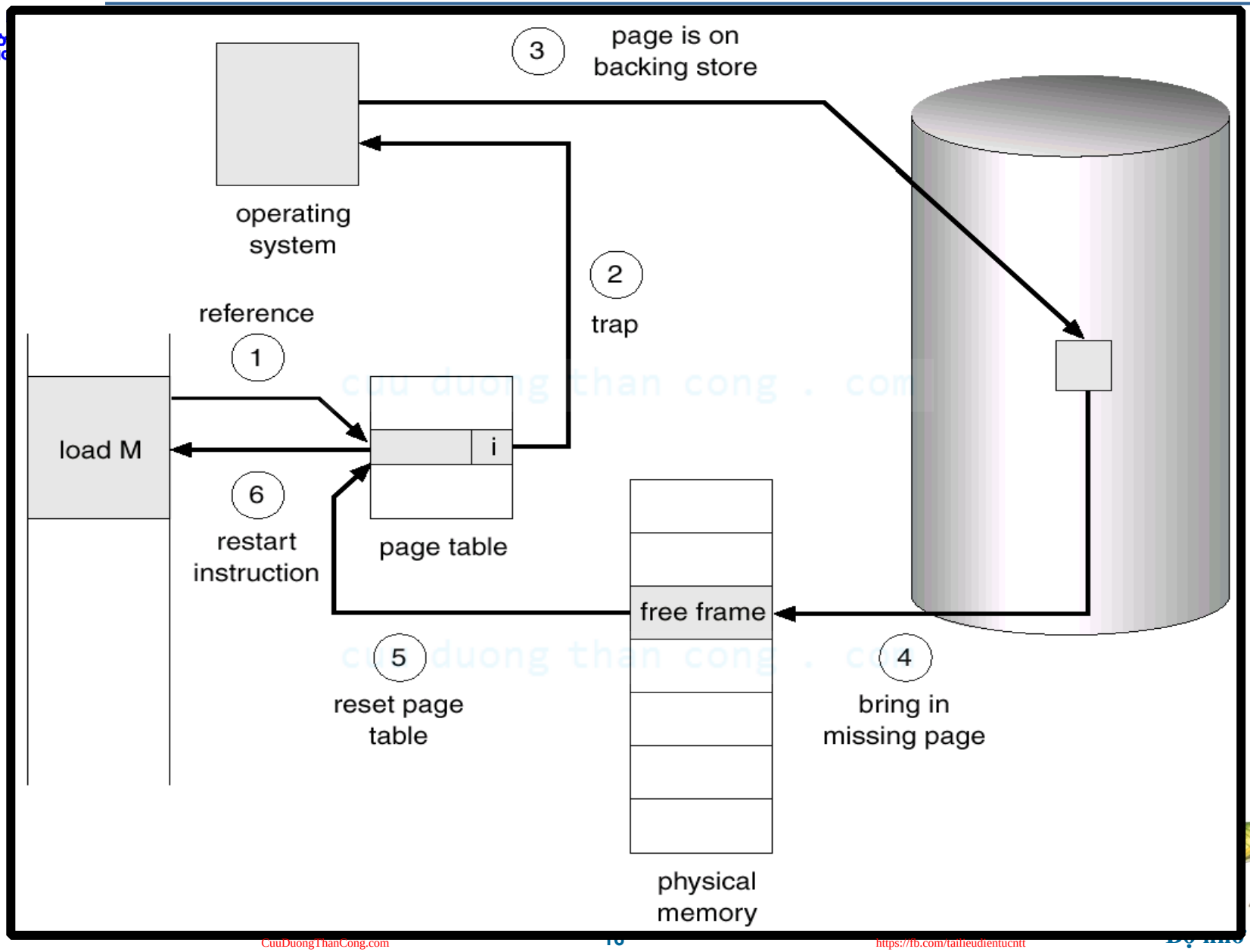
- Có hai kỹ thuật:
  - Phân trang theo yêu cầu (Demand Paging)
  - Phân đoạn theo yêu cầu (Segmentation Paging)
- Phần cứng memory management phải hỗ trợ paging và/hoặc segmentation
- OS phải quản lý sự di chuyển của trang/đoạn giữa bộ nhớ chính và bộ nhớ thứ cấp
- Trong chương này,
  - Chỉ quan tâm đến paging
  - Phần cứng hỗ trợ hiện thực bộ nhớ ảo
  - Các giải thuật của hệ điều hành

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Phân trang theo yêu cầu

- Demand paging: các trang của quá trình chỉ được nạp vào bộ nhớ chính khi được yêu cầu.
- Khi có một tham chiếu đến một trang mà không có trong bộ nhớ chính (valid bit) thì phần cứng sẽ gây ra một ngắt (gọi là page-fault trap) kích khởi page-fault service routine (PFSR) của hệ điều hành.
- PFSR:
  - Chuyển process về trạng thái blocked
  - Phát ra một yêu cầu đọc đĩa để nạp trang được tham chiếu vào một frame trống; trong khi đợi I/O, một process khác được cấp CPU để thực thi
  - Sau khi I/O hoàn tất, đĩa gây ra một ngắt đến hệ điều hành; PFSR cập nhật page table và chuyển process về trạng thái ready.

Lỗi trang và các bước xử lý

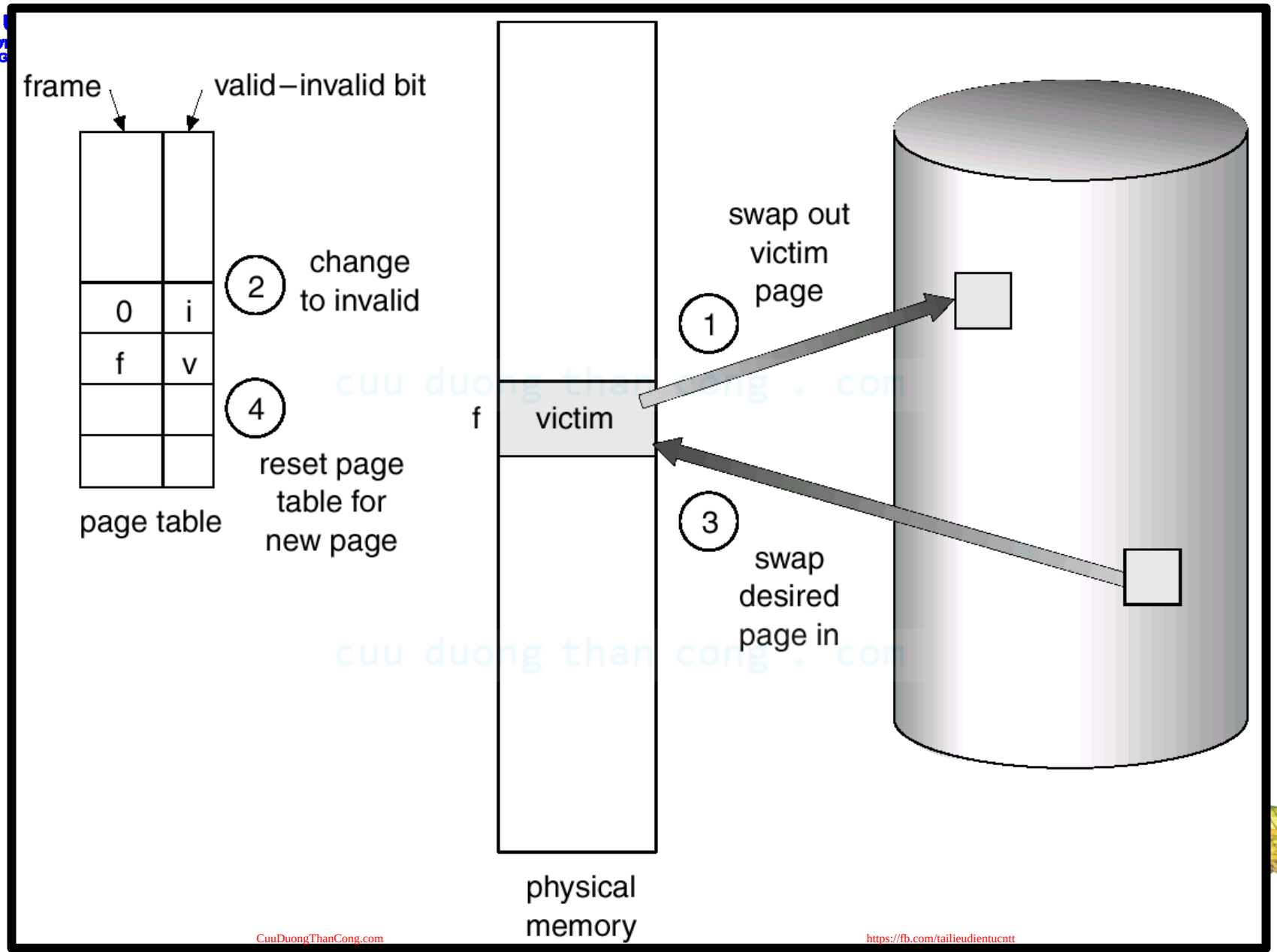


UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Thay thế trang nhớ

- Bước 2 của PFSR giả sử phải thay trang vì không tìm được frame trống, PFSR được bổ sung như sau:
  - Xác định vị trí trên đĩa của trang đang cần
  - Tìm một frame trống:
    - Nếu có frame trống thì dùng nó
    - Nếu không có frame trống thì dùng một giải thuật thay trang để chọn một trang hy sinh (victim page)
    - Ghi victim page lên đĩa; cập nhật page table và frame table tương ứng
  - Đọc trang đang cần vào frame trống (đã có được từ bước 2); cập nhật page table và frame table tương ứng.

Thay thế trang nhớ (tt)



UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Các giải thuật thay thế trang

Hai vấn đề chủ yếu:

- Frame-allocation algorithm
  - Cấp phát cho process bao nhiêu frame của bộ nhớ thực?
- Page-replacement algorithm
  - Chọn frame của process sẽ được thay thế trang nhớ
  - Mục tiêu: số lượng page-fault nhỏ nhất
  - Được đánh giá bằng cách thực thi giải thuật đối với **một chuỗi tham chiếu** bộ nhớ (memory reference string) và xác định số lần xảy ra page fault
- **Ba giải thuật thay thế trang sẽ được xem xét:**
  - **FIFO**
  - **OPT**
  - **LRU**

Các giải thuật thay thế trang

UIT
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

Chuỗi tham chiếu là gì?

Ví dụ: Xét một process với các địa chỉ luận lý như sau:

0100, 0432, 0101, 0612, 0102, 0103, 0104, 0101, 0611, 0102, 0103, 0104, 0101, 0610, 0102, 0103, 0104, 0101, 0609, 0102, 0105

Biết page-size = 100

➔ Các địa chỉ trên sẽ lần lượt ở các trang nhớ:

1, 4, 1, 6, 1, 1, 1, 1, 6, 1, 1, 1, 1, 6, 1, 1, 1, 1, 6, 1, 1

Như vậy, các trang nhớ mà process sẽ tham chiếu đến (làm gọn) là:

1, 4, 1, 6, 1, 6, 1, 6, 1, 6, 1

Chuỗi này gọi là **chuỗi tham chiếu** của process.

Giải thuật thay trang FIFO

UIT
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

Các dữ liệu cần biết ban đầu:

- Số khung trang
- Tình trạng ban đầu
- Chuỗi tham chiếu

Ví dụ: Cho một process có 8 trang (page) và bộ nhớ chính có 3 khung trang (frame), ban đầu các frame này trống. Xét giải thuật thay trang FIFO với chuỗi tham chiếu như sau:

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

*Dấu * tức là có page-fault*

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
*	*	*	*		*	*	*	*	*	*			*	*		.	*	*	*

➔ Có tổng cộng **15 page faults**

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Giải thuật thay trang FIFO

Nhận xét về thay trang theo FIFO:

Thông thường, số frame tăng thì số page faults nên giảm.

Tuy nhiên, với FIFO số frame tăng thì số page faults có thể cũng tăng theo. Điều nghịch lý này còn gọi là **nghịch lý Belady**.

Ví dụ: xét chuỗi tham chiếu sau:

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

trong 2 trường hợp:

- ✓ Sử dụng 3 frame
- ✓ Sử dụng 4 frame

Nghịch lý Belady

UIT TRƯỜNG ĐẠ CÔNG NGHỆ TH

Sử dụng 3 khung trang , sẽ có 9 lỗi trang phát sinh

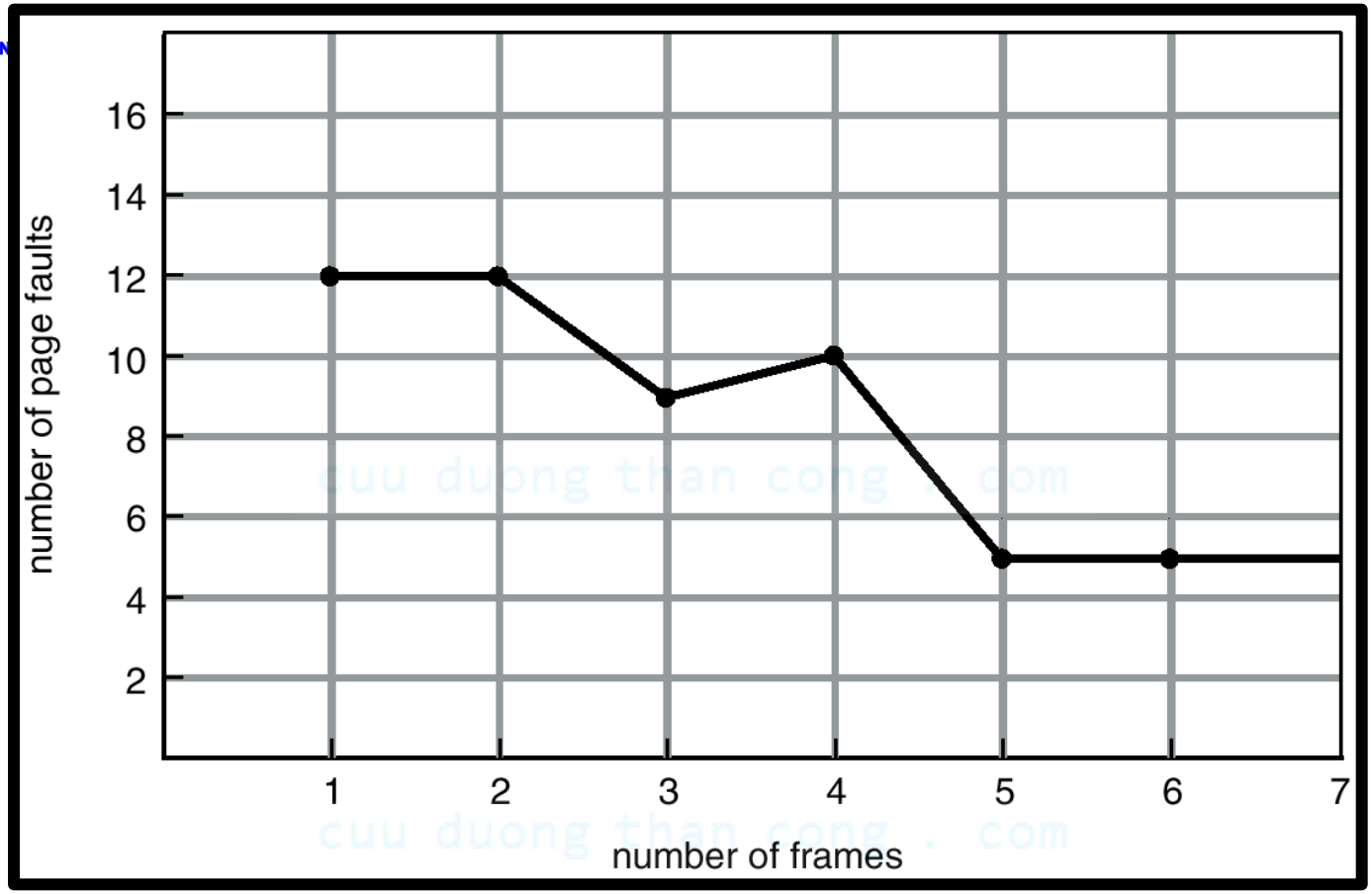
1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	4	4	4	5	5	5	5	5	5
	2	2	2	1	1	1	1	1	3	3	3
		3	3	3	2	2	2	2	2	4	4
*	*	*	*	*	*	*		*	*		

Sử dụng 4 khung trang , sẽ có 10 lỗi trang phát sinh

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

ộ nhớ ảo

Nghịch lý Belady



Bất thường/Nghịc lý (Anomaly) Belady: số page fault tăng mặc dầu quá trình đã được cấp nhiều frame hơn.

Giải thuật thay trang OPT

- Giải thuật thay trang OPT (Optimal Page Replacement)

Trang nhớ bị thay thế sẽ là trang nhớ được tham chiếu trễ nhất trong tương lai

Ví dụ: một process có 8 page, và được cấp 3 frame trống lúc đầu. Xét giải thuật thay trang OPT với chuỗi tham chiếu sau:

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

sử dụng 3 khung trang, khởi đầu 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	2	2	2	2	2	2	2	2	2	2	7	7	7
	0	0	0	0	0	0	4	4	4	0	0	0	0	0	0	0	0	0	0
		1	1	1	3	3	3	3	3	3	3	3	1	1	1	1	1	1	1
*	*	*	*		*		*			*			*				*		

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Giải thuật thay trang LRU

- Giải thuật thay trang LRU (least recently used)

  Trang nhớ bị thay thế sẽ là trang nhớ gần đây ít được sử dụng nhất.

Mỗi trang được ghi nhận (trong bảng phân trang) thời điểm được tham chiếu; từ đó trang bị thay sẽ là trang nhớ có thời điểm tham chiếu nhỏ nhất (OS phải tốn thêm chi phí tìm kiếm trang nhớ bị thay thế này mỗi khi có page fault)

Do vậy, LRU cần sự hỗ trợ của phần cứng và chi phí cho việc tìm kiếm.

Giải thuật thay trang LRU

- Giải thuật thay trang LRU (least recently used)

  Trang nhớ bị thay thế sẽ là trang nhớ gần đây ít được sử dụng nhất.

Ví dụ: một process có 8 page, và được cấp 3 frame trống lúc đầu. Xét giải thuật thay trang LRU với chuỗi tham chiếu sau:

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

sử dụng 3 khung trang, khởi đầu đều trống:

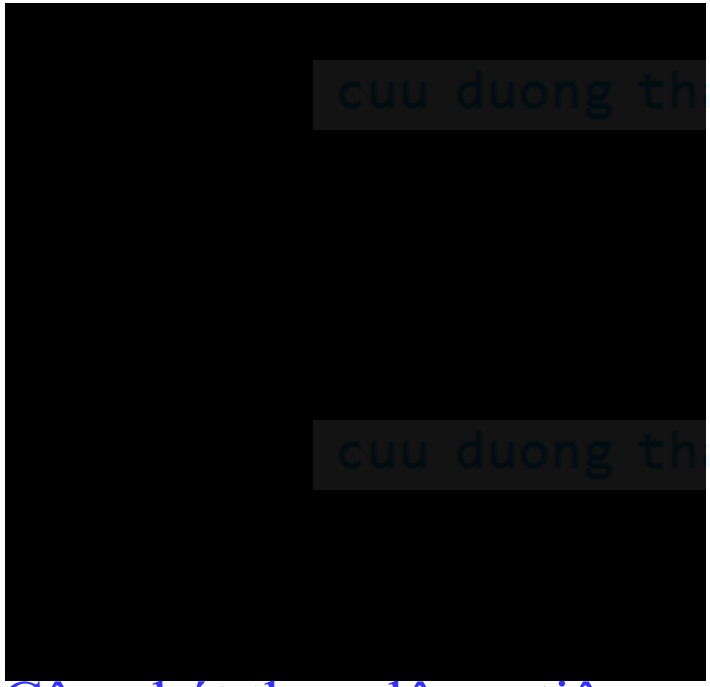
7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2	2	2	2	4	4	4	0	0	0	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0	3	3	3	3	3	3	0	0	0	0	0
		1	1	1	3	3	3	2	2	2	2	2	2	2	2	2	7	7	7
*	*	*	*		*		*	*	*	*			*		*		*		

Số lượng frame cấp cho process

- OS phải quyết định cấp cho mỗi process bao nhiêu frame.
  - Cấp ít frame ⇒ nhiều page fault
  - Cấp nhiều frame ⇒ giảm mức độ multiprogramming
- Chiến lược **cấp phát tĩnh** (fixed-allocation)
  - Số frame cấp cho mỗi process không đổi, được xác định vào thời điểm loading và có thể tùy thuộc vào từng ứng dụng (kích thước của nó, …)
- Chiến lược **cấp phát động** (variable-allocation)
  - Số frame cấp cho mỗi process có thể thay đổi trong khi nó chạy
    - Nếu tỷ lệ page-fault cao ⇒ cấp thêm frame
    - Nếu tỷ lệ page-fault thấp ⇒ giảm bớt frame
  - OS phải mất chi phí để ước định các process

Chiến lược cấp phát tĩnh

- *Cấp phát bằng nhau*: Ví dụ, có 100 frame và 5 process → mỗi process được 20 frame
- *Cấp phát theo tỉ lệ*: dựa vào kích thước process

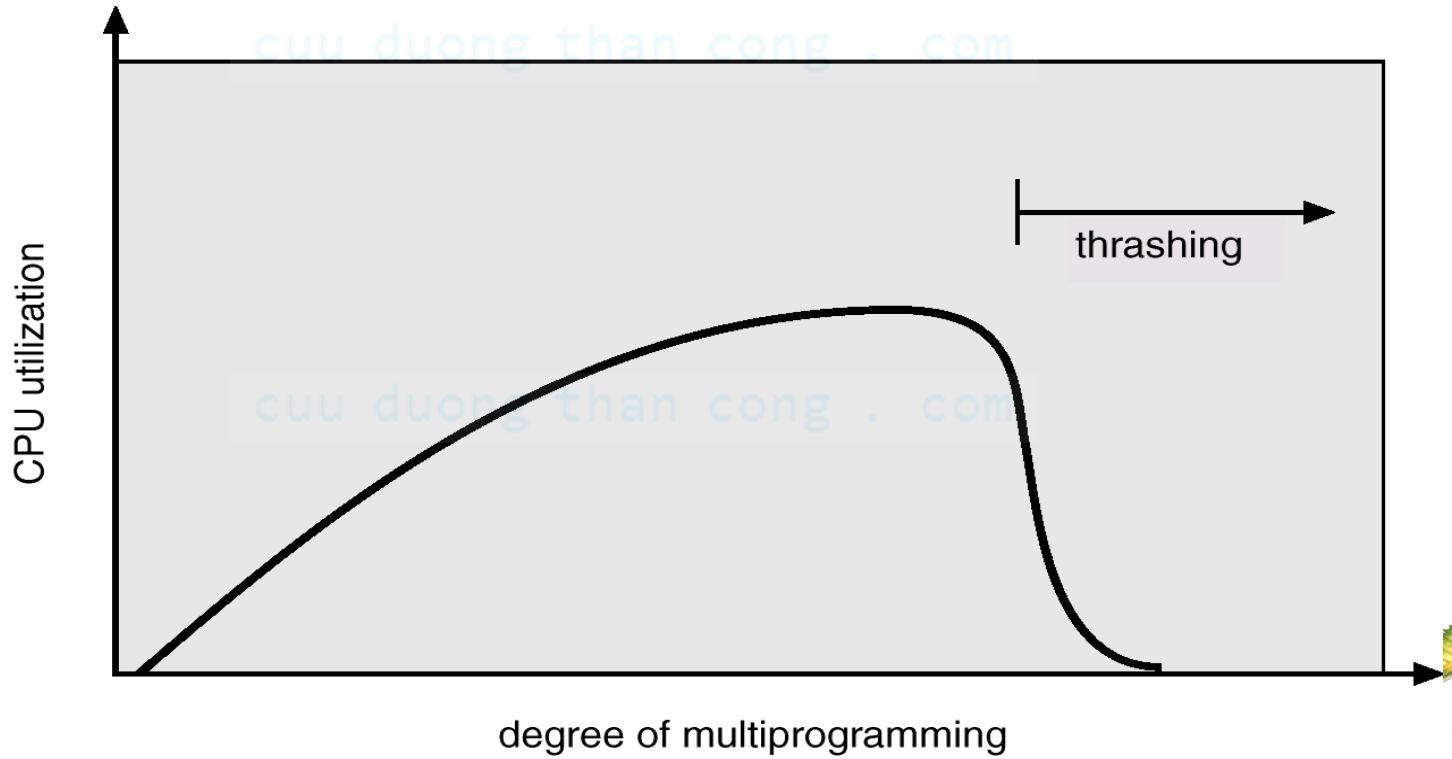


Ví dụ:

- Cấp phát theo độ ưu tiên

Trì trệ trên toàn bộ hệ thống

- Nếu một process không có đủ số frame cần thiết thì tỉ số page faults/sec rất cao.
- Thrashing: hiện tượng các trang nhớ của một process bị hoán chuyển vào/ra liên tục.



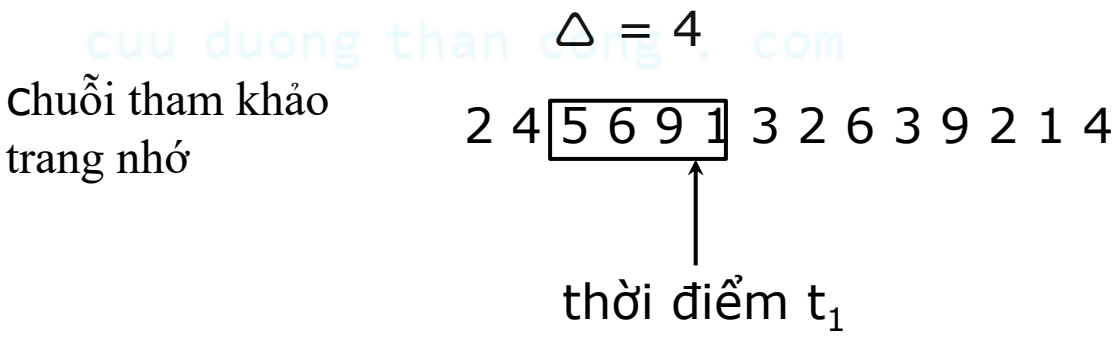
UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Mô hình cục bộ

- Để hạn chế thrashing, hệ điều hành phải cung cấp cho process càng "đủ" frame càng tốt. Bao nhiêu frame thì đủ cho một process thực thi hiệu quả?
- Nguyên lý locality (locality principle)
  - Locality là tập các trang được tham chiếu gần nhau
  - Một process gồm nhiều locality, và trong quá trình thực thi, process sẽ chuyển từ locality này sang locality khác
- Vì sao hiện tượng thrashing xuất hiện?
  Khi Σ size of locality > memory size

Giải pháp tập làm việc

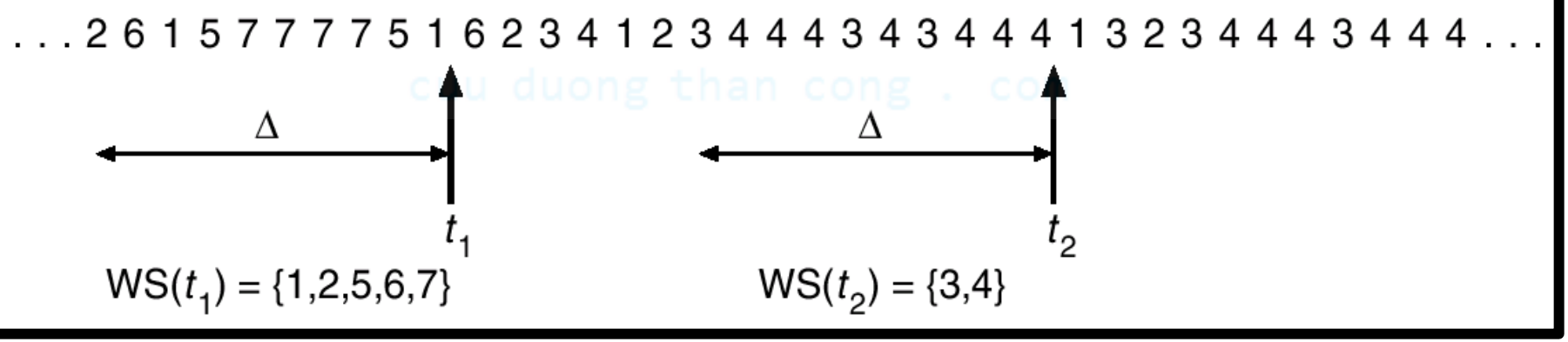
- Được thiết kế dựa trên nguyên lý locality.
- Xác định xem process thực sự sử dụng bao nhiêu frame.
- Định nghĩa:
  - WS(t) - số lượng các tham chiếu trang nhớ của process gần đây nhất cần được quan sát.
  - Δ - khoảng thời gian tham chiếu
- Ví dụ:



Giải pháp tập làm việc (tt)

- Định nghĩa: working set của process Pi , ký hiệu WSi , là tập gồm Δ các trang được sử dụng gần đây nhất.

Ví dụ: Δ = 10 và chuỗi tham khảo trang



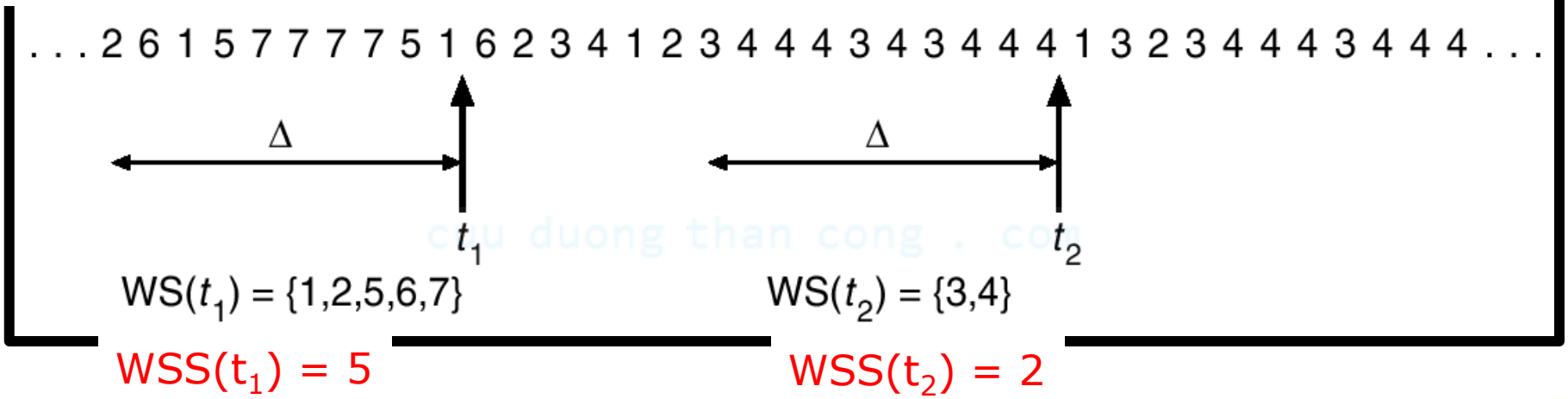
- Nhận xét:
  - Δ quá nhỏ ⇒ không đủ bao phủ toàn bộ locality.
  - Δ quá lớn ⇒ bao phủ nhiều locality khác nhau.
  - Δ = ∞ ⇒ bao gồm tất cả các trang được sử dụng.

Dùng working set của một process để xấp xỉ locality của nó.

Giải pháp tập làm việc (tt)

- Định nghĩa: WSSi là kích thước của working set của Pi:
  - WSSi = số lượng các trang trong WSi

Ví dụ: Δ = 10 và chuỗi tham khảo trang



Giải pháp tập làm việc (tt)

- Đặt D = Σ WSSi = tổng các working-set size của mọi process trong hệ thống.
  - Nhận xét: Nếu D > m (số frame của hệ thống) ⇒ sẽ xảy ra thrashing.
- Giải pháp working set:
  - Khi khởi tạo một tiến trình: cung cấp cho tiến trình số lượng frame thỏa mãn working-set size của nó.
  - Nếu D > m ⇒ tạm dừng một trong các process.
    - Các trang của tiến trình được chuyển ra đĩa cứng và các frame của nó được thu hồi.

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Giải pháp tập làm việc (tt)

- WS loại trừ được tình trạng trì trệ mà vẫn đảm bảo mức độ đa chương
- Theo vết các WS? ➔ WS xấp xỉ (đọc thêm trong sách)
- Đọc thêm:
  - Hệ thống tập tin
  - Hệ thống nhập xuất
  - Hệ thống phân tán

Kết thúc chương 8