HỆ ĐIỀU HÀNH

Phạm Đăng Hải
haipd-fit@mail.hut.edu.vn

Bộ môn Khoa học Máy tính
Viện Công nghệ Thông tin & Truyền Thông

Ngày 5 tháng 4 năm 2011

Chương 3 Quản lý bộ nhớ

Giới thiệu

- Mục đích của hệ thống máy tính: thực hiện chương trình
  - Chương trình và dữ liệu (*toàn bộ hoặc một phần*) phải nằm trong bộ nhớ chính trong khi thực hiện
  - **Byte tích cực:**Những byte nội dung đang được thực hiện tại thời điểm quan sát:
  - Phần chương trình chưa đưa vào bộ nhớ chính được lưu trên bộ nhớ thứ cấp (*VD: đĩa cứng*)$\Rightarrow$ **Bộ nhớ ảo**
    - Cho phép lập trình viên không lo lắng về giới hạn bộ nhớ vật lý

Giới thiệu

- Mục đích của hệ thống máy tính: thực hiện chương trình
  - Chương trình và dữ liệu (*toàn bộ hoặc một phần*) phải nằm trong bộ nhớ chính trong khi thực hiện
  - **Byte tích cực:**Những byte nội dung đang được thực hiện tại thời điểm quan sát:
  - Phần chương trình chưa đưa vào bộ nhớ chính được lưu trên bộ nhớ thứ cấp (*VD: đĩa cứng*)⇒ **Bộ nhớ ảo**
    - Cho phép lập trình viên không lo lắng về giới hạn bộ nhớ vật lý
- Để s/d CPU hiệu quả và tăng tốc độ đáp ứng của hệ thống:
  - Cần luân chuyển CPU thường xuyên giữa các tiến trình
    - Điều phối CPU (*Phần 3- Chương 2*)
  - Cần nhiều tiến trình sẵn sàng trong bộ nhớ
    - **Hệ số song song của hệ thống**: Số tiến trình đồng thời tồn tại trong hệ thống

Giới thiệu

- Mục đích của hệ thống máy tính: thực hiện chương trình
  - Chương trình và dữ liệu (*toàn bộ hoặc một phần*) phải nằm trong bộ nhớ chính trong khi thực hiện
  - **Byte tích cực:**Những byte nội dung đang được thực hiện tại thời điểm quan sát:
  - Phần chương trình chưa đưa vào bộ nhớ chính được lưu trên bộ nhớ thứ cấp (*VD: đĩa cứng*)⇒ **Bộ nhớ ảo**
    - Cho phép lập trình viên không lo lắng về giới hạn bộ nhớ vật lý
- Để s/d CPU hiệu quả và tăng tốc độ đáp ứng của hệ thống:
  - Cần luân chuyển CPU thường xuyên giữa các tiến trình
    - Điều phối CPU (*Phần 3- Chương 2*)
  - Cần nhiều tiến trình sẵn sàng trong bộ nhớ
    - **Hệ số song song của hệ thống**: Số tiến trình đồng thời tồn tại trong hệ thống
- Tồn tại nhiều sơ đồ quản lý bộ nhớ khác nhau
  - Nhiều sơ đồ đòi hỏi trợ giúp từ phần cứng
  - Thiết kế phần cứng có thể được tích hợp chặt chẽ với HDH

Nội dung chính

Nội dung chính

Đặt vấn đề

- Câu lệnh phải nằm trong bộ nhớ khi thực hiện !

Đặt vấn đề

- Câu lệnh phải nằm trong bộ nhớ khi thực hiện !
- Toàn bộ chương trình phải nằm trong bộ nhớ ?
  - Cấu trúc động; cấu trúc Overlays... : Nạp từng phần
    - Đòi hỏi sự chú ý đặc biệt từ lập trình viên

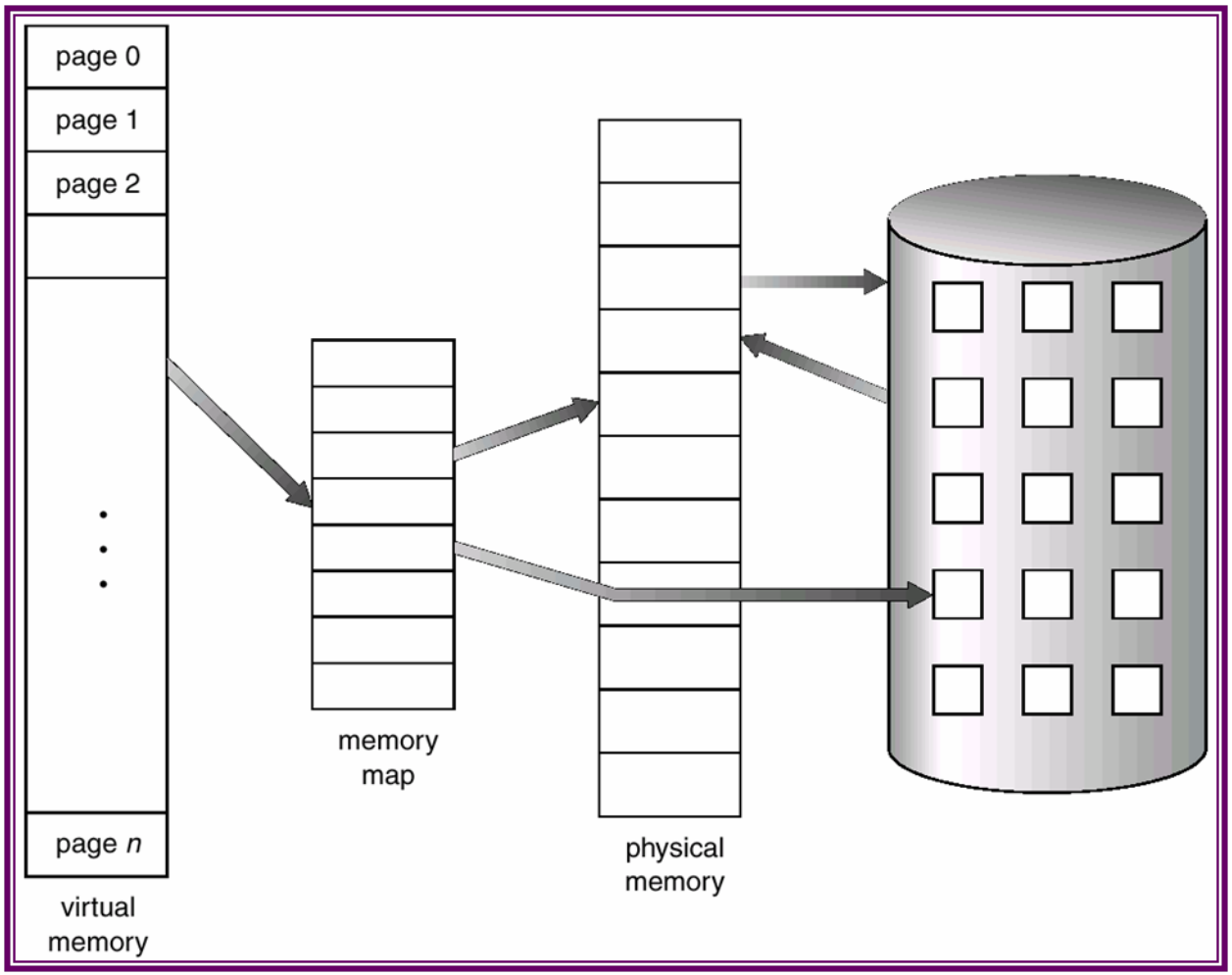
  $\Rightarrow$Không cần thiết
  - Đoạn chương trình xử lý báo lỗi
    - Lỗi ít xảy tra, ít được thực hiện
  - Phần khai báo mảng, danh sách không dùng tới
    - Khai báo ma trận 100x100, sử dụng 10x 10

Đặt vấn đề

- Câu lệnh phải nằm trong bộ nhớ khi thực hiện !
- Toàn bộ chương trình phải nằm trong bộ nhớ ?
  - Cấu trúc động; cấu trúc Overlays... : Nạp từng phần
    - Đòi hỏi sự chú ý đặc biệt từ lập trình viên

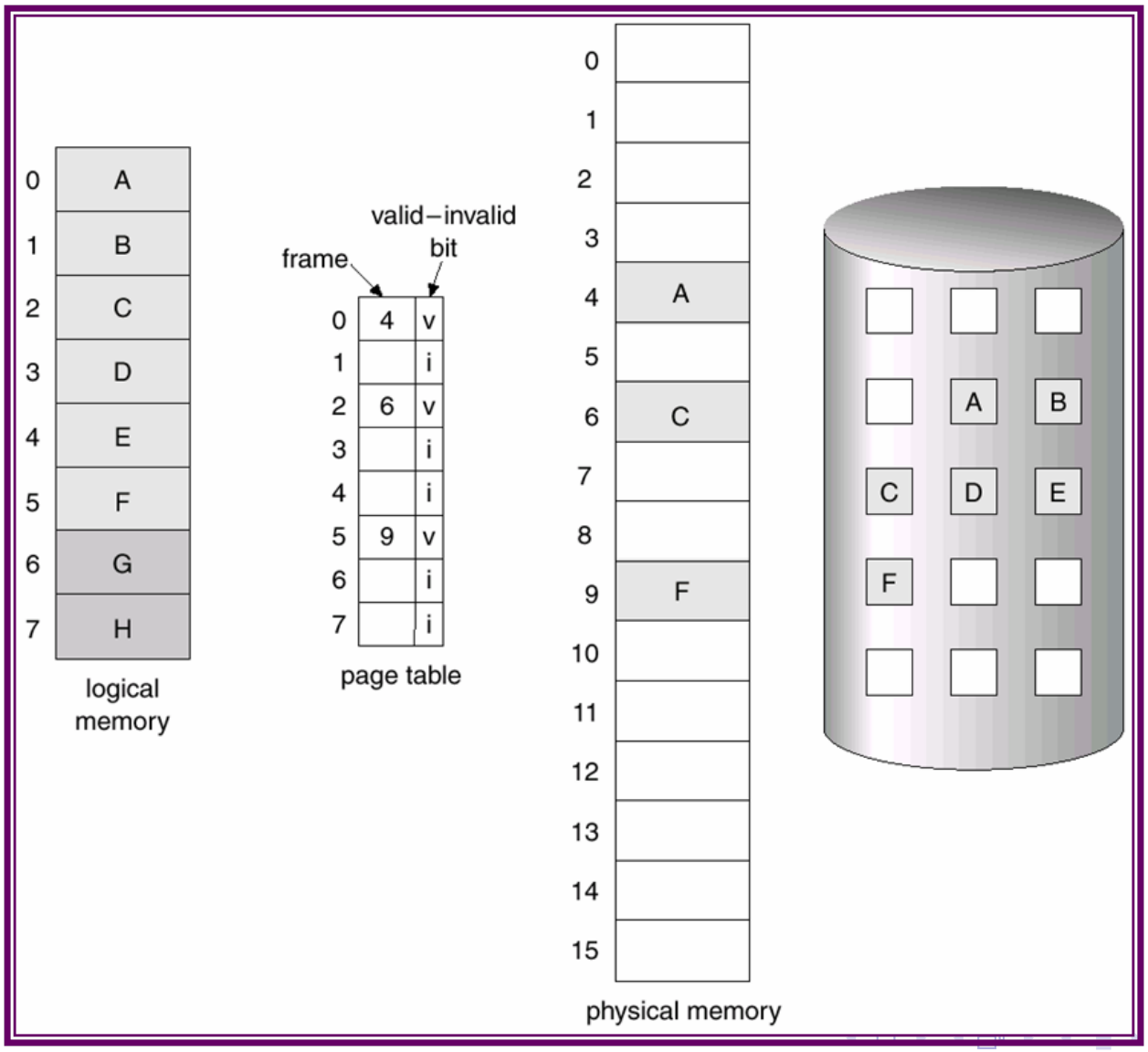
  $\Rightarrow$Không cần thiết
  - Đoạn chương trình xử lý báo lỗi
    - Lỗi ít xảy tra, ít được thực hiện
  - Phần khai báo mảng, danh sách không dùng tới
    - Khai báo ma trận 100x100, sử dụng 10x 10
- Thực hiện chương trình chỉ có 1 phần nằm trong bộ nhớ
  - Cho phép viết chương trình trong không gian địa chỉ áo (*virtual address space*) lớn tùy ý
  - Cho phép nhiều chương trình đồng thời tồn tại, tăng hiệu suất sử dụng CPU
  - Giảm yêu cầu vào ra cho việc nạp và hoán đổi chương trình

Bộ nhớ ảo

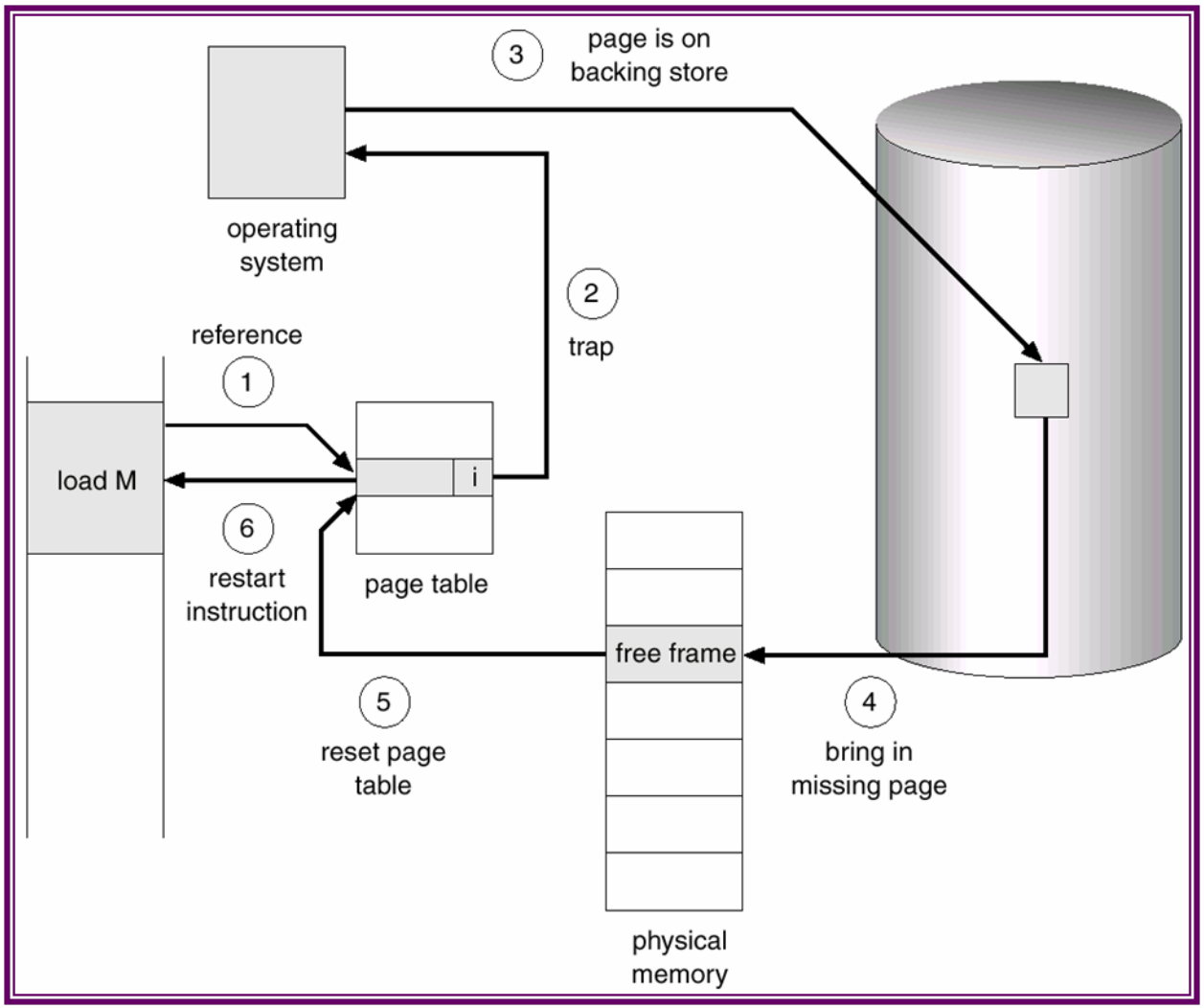


- Phân tách bộ nhớ logic với bộ nhớ vật lý
- Cài đặt theo kiểu phân trang/phân đoạn

Nạp từng phần của trang chương trình vào bộ nhớ



Xử lý lỗi trang



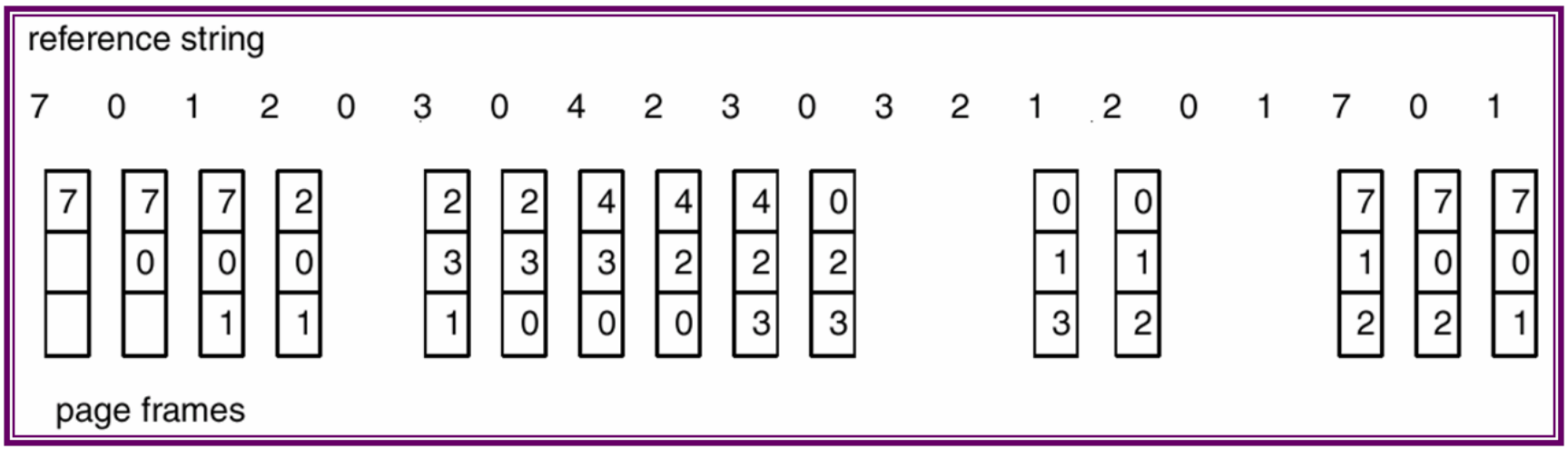
Nếu không có frames tự do, phải tiến hành đổi trang

Các chiến lược

- FIFO (First In First Out): Vào trước ra trước
- OPT/MIN Thuật toán thay thế trang tối ưu
- LRU (Least Recently Used): Trang có lần sử dụng cuối cách lâu nhất
- LFU (Least Frequently used):Tần xuất sử dụng **thấp** nhất
- MFU (Most Frequently used): Tần xuất sử dụng **cao** nhất
- ...

FIFO

Ví dụ

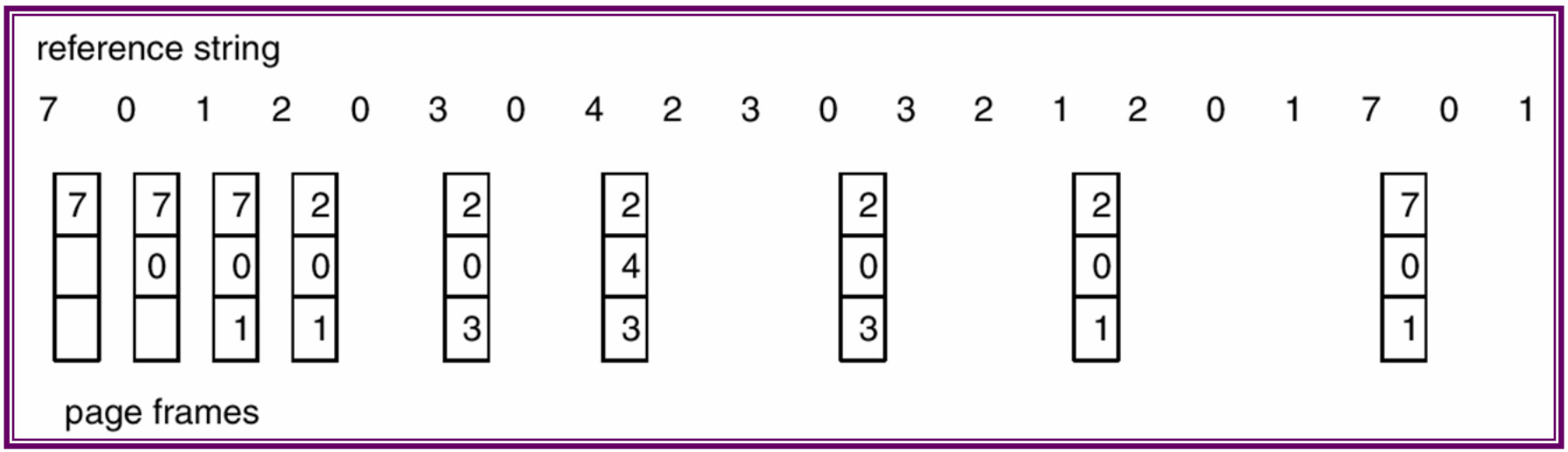


Nhận xét

- Hiệu quả khi chương trình có cấu trúc tuyến tính. Kém hiểu quả khi chương trình theo nguyên tắc lập trình cấu trúc
- Đơn giản dễ thực hiện
  - Dùng hàng đợi lưu các trang của chương trình trong bộ nhớ
  - Chèn ở cuối hàng, Thay thế trang ở đầu hàng
- Tăng trang vật lý, không đảm bảo giảm số lần gặp lỗi trang
  - Dãy truy nhập: 1 2 3 4 1 2 5 1 2 3 4 5
  - 3 frames: 9 lỗi trang; 4 frames: 10 lỗi trang

OPT

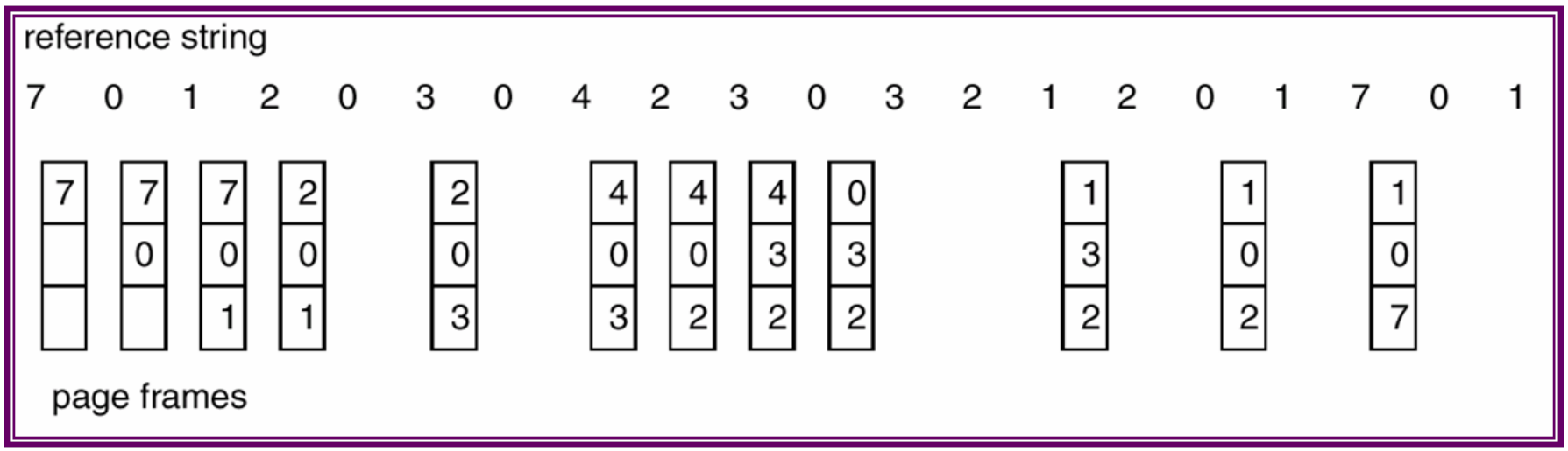
Nguyên tắc: Đưa ra trang có lần sử dụng tiếp theo cách xa nhất



- Số lần gặp lỗi trang ít nhất
- Khó dự báo được diễn biến của chương trình

LRU

Nguyên tắc: Đưa ra trang có lần sử dụng cuối cách xa nhất



- Hiệu quả cho chiến lược thay thế trang
- Đảm bảo giảm số lỗi trang khi tăng số trang vật lý
  - Tập các trang trong bộ nhớ có n frames luôn là tập con của các trang trong bộ nhớ có $n+1$ frames
- Y/cầu sự trợ giúp kỹ thuật để chỉ ra thời điểm truy nhập cuối
- Cài đặt như thế nào?

LRU: Cài đặt

- Bộ đếm
  - Thêm một trường ghi thời điểm truy nhập vào mỗi phần tử của PCB
  - Thêm vào khối điều khiển (C.U) đồng hồ/bộ đếm
  - Khi có yêu cầu truy nhập trang
    - Tăng bộ đếm
    - Chép nội dung bộ đếm vào trường thời điểm truy nhập tại phần tử tương ứng trong PCB
  - Cần có thủ tục cập nhật PCB (*ghi vào trường thời điểm*) và thủ tục tìm kiếm trang có giá trị trường thời điểm nhỏ nhất
  - Hiện tượng tràn số !?
- Dãy số
  - Dùng dãy số ghi số trang
    - Truy nhập tới một trang, cho phần tử tương ứng lên đầu dãy
  - Thay thế trang: Phần tử cuối dãy
  - Thường cài đặt dưới dạng DSLK 2 chiều
    - 4 phép gán con trỏ $\Rightarrow$ tốn thời gian

Thuật toán dựa trên bộ đếm

Sử dụng bộ đếm (*một trường của PCB*) ghi nhận số lần truy nhập tới trang

Thuật toán dựa trên bộ đếm

Sử dụng bộ đếm (*một trường của PCB*) ghi nhận số lần truy nhập tới trang

- LFU: Trang có bộ đếm nhỏ nhất bị thay thế
  - Trang truy nhập nhiều đến
    - Trang quan trọng $\Rightarrow$ hợp lý
    - Trang khởi tạo, chỉ được dùng ở giai đoạn đầu $\Rightarrow$ không hợp lý
      $\Rightarrow$Dịch bộ đếm một bit (chia đôi) theo thời gian

Thuật toán dựa trên bộ đếm

Sử dụng bộ đếm (*một trường của PCB*) ghi nhận số lần truy nhập tới trang

- LFU: Trang có bộ đếm nhỏ nhất bị thay thế
  - Trang truy nhập nhiều đến
    - Trang quan trọng ⇒ hợp lý
    - Trang khởi tạo, chỉ được dùng ở giai đoạn đầu ⇒ không hợp lý
      ⇒Dịch bộ đếm một bit (chia đôi) theo thời gian
- MFU: Trang có bộ đếm lớn nhất
  - Trang có bộ đếm nhỏ nhất, vừa mới được nạp vào và vẫn chưa được sử dụng nhiều

Kết luận

1. **Bộ nhớ ảo**
   - 3.1 Giới thiệu
   - 3.2 Các chiến lược đổi trang