TÀI LIỆU HỆ ĐIỀU HÀNH **CHƯƠNG 7: QUẢN LÝ BỘ NHỚ_PHẦN 1**

DÀNH CHO KHÓA 13 KHÓA NGÀY: 6 May 2020

HỆ THỐNG LÝ THUYẾT

1. Các khái niệm

- Quản lý bộ nhớ là công việc của hệ điều hành với sự hỗ trợ của phần cứng, sắp xếp các process trong bộ nhớ sao cho hiệu quả.
- **Mục tiêu:** Càng nhiều process vào bộ nhớ càng tốt
- Trong hầu hết các hệ thống thi kernel sẽ chiếm một phần cố định trong bộ nhớ. Ví dụ: RAM 8GB chỉ sử dụng được 7,88GB còn lại là của kernel.

2. Các yêu cầu đối với việc quản lý bộ nhớ

- Cấp phát tài nguyên cho process
- Tái định vị
- Bảo vệ: Kiểm tra truy xuất bộ nhớ có hợp lệ hay không
- Chia sẻ: Cho phép các process chia sẻ vùng nhớ chung
- Kết gán địa chỉ nhớ luận lý của user vào địa chỉ thực.

3. Các loại địa chỉ

- **Địa chỉ vật lý** là địa chỉ thực trong bộ nhớ chính.
- **Địa chỉ luận lý** là một vị trí nhớ được diễn tả trong chương trình (Còn gọi là địa chỉ ảo).
  - Địa chỉ tương đối là một kiểu địa chỉ luận lý trong đó địa chỉ được biểu diễn tương đối so với một vị trí xác định nào đó trong chương trình

- Địa chỉ tuyệt đối: Địa chỉ tương đương với địa chỉ thực.

- Bộ linker: **Kết hợp các object module** thành một file **nhị phân** khả thực thi gọi là load module.
- Bộ loader: Nạp load module vào bộ nhớ chính.
- Đến giữa quá trình linker và loader mới có địa chỉ thật.
- Chuyển đổi địa chỉ là quá trình ánh xạ một địa chỉ từ không gian địa chỉ này sang không gian địa chỉ khác.
    - Trong source code: **symbolic** (các biến, hằng, con trỏ)
    - Trong biên dịch: Địa chỉ tương đối (Ví dụ: 12 bytes so với vị trí bắt đầu của chương trình).
    - Thời điểm linking/loading: Có thể là địa chỉ thực.

4. Dynamic Linking (Liên kết động)

- Dynamic linking quá trình **liên kết đến một module ngoài** được thực hiện **sau khi đã tạo xong** loader module.
- Ưu điểm
    - Cung cấp tiện ích của OS. Chương trình thực thi có thể dùng các module ngoài phiên bản khác nhau mà không cần sửa đổi.
    - Chia sẻ mã
    - Dynamic linking cần sự hỗ trợ của OS để kiểm tra xem một thủ tục nào đó có thể được chia sẻ giữa các process hay là phần mã của riêng một process (bởi vì chỉ có OS mới có quyền thực hiện việc kiểm tra này).

5. **Dynamic Loading**

- Khi cần gọi đến thì một thủ tục mới được nạp vào bộ nhớ chính → tăng độ hiệu dụng của bộ nhớ chính.
- Hiệu quả trong trường hợp tồn tại khối lượng lớn. Mã chương trình có tần suất sử dụng thấp, không được sử dụng thường xuyên.
- User thiết kế chương trình có dynamic loading. Hệ điều hành cung cấp một số thư viện hỗ trợ, tạo điều kiện dễ dàng cho lập trình viên,

6. **Cơ chế phủ lắp, cơ chế hoán vị**

- **Cơ chế phủ lắp(overlay)**: Tại mỗi thời điểm **chỉ giữ lại** trong bộ nhớ chính những lệnh hoặc dữ liệu cần thiết, **giải phóng** những lệnh và giữ liệu không cần thiết hoặc chưa sử dụng. Ví dụ như thu hồi cấp phát bộ nhớ động (**Không cần sự hỗ trợ của hệ điều hành**).
- **Cơ chế hoán vị:** Một tiến trình có thể tạm thời bị swap ra khỏi bộ nhớ chính và lưu trên một hệ thống lưu trữ phụ. Sau đó, tiến trình có thể được nạp lại vào bộ nhớ để tiếp tục quá trình thực thi. Hiện nay ít hệ thống sử dụng cơ chế này.

7. **Mô hình quản lý bộ nhớ**

- Một process phải được nạp **hoàn toàn** vào bộ nhớ thì mới được thực thi. (Ngoại trừ cơ chế phủ lắp -overlay)

8. **Phân mảnh ngoại, phân mảnh nội**

- **Phân mảnh ngoại:** Kích thước không gian nhớ còn trống đủ để thỏa mãn một yêu cầu cấp phát, tuy nhiên không gian bộ nhớ này không liên tục=> Có thể dùng cơ chế kết khối để gom lại thành một vùng nhớ liên tục.

- **Phân mảnh nội:** Kích thước vùng nhớ có thể hơi lớn so với yêu cầu. Cần 1GB mà cấp 2GB sẽ lãng phí 1GB. Hệ điều hành sẽ quản lý 1GB không dùng sử dụng cho một mục dích khác.

9. Fixed partitioning và chiến lược placement

- Khi khởi động hệ thống, bộ nhớ chính được chia thành nhiều phần rời nhau được gọi là partition có kích thước bằng nhau hoặc khác nhau. Tiến trình có kích thước < hoặc = kích thước partition thì mới có thể nạp vào partition đó được. Nếu chương trình có kích thước lớn hơn **dùng cơ chế overlay.**
- Chiến lược placement

Nếu partition bằng nhau	Nếu partition có kích thước khác nhau
- Nếu partition còn trống thì process được nạp vào partition đó. - Nếu partition không còn trống nhưng trong partition có một process đang bị blocked. Swap process đó ra bộ process bị blocked ra bộ nhớ phụ nhường chỗ cho process mới.	**Giải pháp 1** - Gán process vào partition nhất phù hợp với nó. - Có hàng đợi cho mỗi partition - Giảm thiểu phân mảnh nội - Vấn đề: Hàng đợi trống không và dày đặc **Giải pháp 2** - Chỉ có hàng đợi chung cho mọi partition - Khi cần nạp chọn partition nhỏ nhất còn trống.

10. Dynamic partitioning và các chiến lược placement

- Số lượng partition là **không cố định** có thể có kích thước khác nhau
- Mỗi process được cấp dung lượng chính xác cần thiết.
- Gây ra tình trạng phân mảnh ngoại.
- Chiến lượng placement
  - **Best fit:** Chọn khối nhớ trống nhỏ nhất
  - **First fit:** Chọn khối nhớ trống phù hợp đầu tiên kể từ bộ nhớ
  - **Next fit:** Chọn khối nhớ trống phù hợp đầu tiên kể từ vị trí cấp phát cuối cùng.
  - **Worst fit:** Chọn khối nhớ trống lớn nhất.

Bài tập 1.

Giả sử bộ nhớ chính được cấp phát các phân vùng có kích thước là 600K, 500K, 200K, 300K (theo thứ tự), sau khi thực thi xong, các tiến trình có kích thước 212K, 417K, 112K, 426K (theo thứ tự) sẽ được cấp phát bộ nhớ như thế nào, nếu sử dụng: Thuật toán First fit, Best fit, Next fit, Worst fit? Thuật toán nào cho phép sử dụng bộ nhớ hiệu quả nhất trong trường hợp trên?

Giải bài tập

Fisrt fit: 212K sẽ được cấp cho vùng nhớ 600K, 417K được cấp phát cho vùng nhớ 500K, 112K được cấp phát cho vùng nhớ 200K (Có thể lấy 600 -212K = 388K) để cấp cho 112K, 426K phải đợi do không còn vùng nhớ thỏa yêu cầu.

Best fit: 212K sẽ được cấp cho bộ nhớ 300K, 417K sẽ được cấp cho bộ nhớ 500K, 112K sẽ được cấp cho bộ nhớ 200K, 426K sẽ được cấp cho bộ nhớ 600K.

Worst fit: 212K sẽ được cấp cho vùng nhớ 600K, 417K được cấp phát cho vùng nhớ 500K, 112K được cấp phát cho vùng nhớ 300K, 426K phải đợi do không còn vùng nhớ thỏa yêu cầu.

Next fit: 212K sẽ được cấp cho vùng nhớ 600K, 417K được cấp phát cho vùng nhớ 500K, 112K được cấp phát cho vùng nhớ 200K (Có thể lấy 600 -212K = 388K) để cấp cho 112K, 426K phải đợi do không còn vùng nhớ thỏa yêu cầu.

Bài tập 2.

Gỉa sử bộ nhớ chính được phân thành các phân vùng có kích thước là 900K, 400K, 600K, 300K ***(theo thứ tự)*** và con trỏ đang trỏ vào vùng nhớ 400K, cho biết các tiến trình có kích thước 700K, 400K, 300K và 200K ***(theo thứ tự)*** sẽ được cấp phát bộ nhớ như thế nào, nếu sử dụng: First fit, Best fit, Next fit, Worst fit.

---------- Hết ----------

NMN TRAINING study hard