TÀI LIỆU HỆ ĐIỀU HÀNH **CHƯƠNG 5: ĐỒNG BỘ_PHẦN 3**

DÀNH CHO KHÓA 13 KHÓA NGÀY: 20 April 2020

HỆ THỐNG LÝ THUYẾT

2. Nhóm giải thuật Sleep & Wake up

- **Semaphore**
- Là công cụ **được cung cấp bởi Hệ điều hành** mà không đòi hỏi busy waiting
- Semaphore là một biến số nguyên
- Semaphore có 3 tác vụ cơ bản là
  - Khởi động
  - wait(S): giảm giá trị semaphore 1 đơn vị. Kế đó nếu giá trị này âm thì process thực hiện lệnh wait() và blocked
  - signal(S): tăng giá trị semaphore 1 đơn v. Kế đó nếu giá trị này không dương, một process đang bị blocked bởi một lệnh wait sẽ phục hồi.

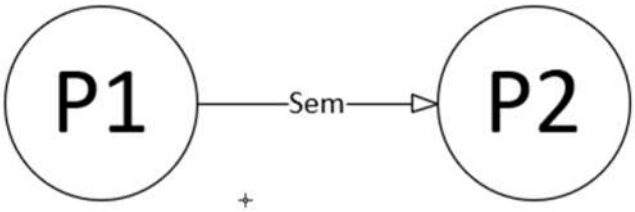
⇨ Wait và signal có tính đơn nguyên và loại trừ tương hỗ.

** Chú thích biến count > 0 thì thực thi. Không dương thì vào hàng đợi.

- Khi một process phải chờ trong semaphore S, nó sẽ bị **blocked** và được đặt trong hàng đợi semaphore
- Tác vụ signal() thường sử dụng **cơ chế FIFO**
- Block: chuyển từ running sang waiting
- Wakeup: Chuyển từ waiting sang running

- Không sử dụng đúng cách sẽ gây ra tình trạng đói tài nguyên hoặc deadlock.
- **Ví dụ 1: Sử dụng Semaphore**
  - Hai process P1; P2
  - Yêu cầu S1 trong P1 phải thực hiện trước S2 trong P2
  - Định nghĩa Semaphore để đồng bộ
  - Bước 1: Khởi động Semaphore
    - Sem.value = 0
  - ChBước 2: Để đồng bộ tiến trình S1 phải thực hiện trước S2

    Ta có sơ đồ

    Theo trình tự hình thì S1 thực hiện trước S2
  - Theo sơ đồ thì tiến trình bị tiến trình khác trỏ vào là **wait.** Tiến trình mà trỏ vào tiến trình khác là **signal.**
  - Vậy ta có định nghĩa

    P1

    {S1;

    Signal(sem);

    }

Rồi sau đó đến P2

P2

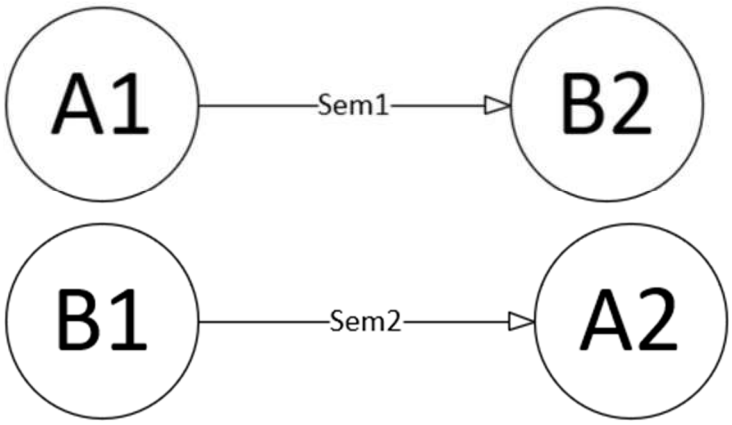
{

Wait(sem);

S2;

}

- **Ví dụ 2: Sử dụng Semaphore**
  - Tiến trình P1{A1;A2} tiến trình P2{B1;B2}
  - Đồng bộ tiến trình làm sau cho A1 và B1 đều hoàn thành trước A2 và B2
  - Khởi tạo s1.value=s2.value = 0
  - Ta có sơ đồ



Từ đây ta có

P1

{

A1;

Signal(s1);

Wait(s2);

A2;

}

P2

{B1;

Signal(s2);

Wait(s1);

}

- **Nhận xét**
  - Khi giá trị value của semaphore S không âm: **thì số số process có thể thực thi bằng S.value** (Không bị blocked).
  - Khi giá trị value của semaphore S âm: **thì số process trong hàng đợi bằng |S.value|** (Process bị blocked).
  - Trong Semaphore đoạn mã định nghĩa các lệnh wait và signal **chính là vùng tranh chấp (miền găng).** Thường khoảng 10 lệnh
  - Giải pháp cho vùng tranh chấp wait và signal
    - Hệ thống đơn chương: Cấm ngắt
    - Hệ thống đa chương: Dekker, Peterson (Phần mềm) hoặc phần mềm TestAndSet, Swap.
  - **CS** nhỏ nên busy thấp
- **CR (Critical Region)**

- Là cấu trúc ngôn ngữ cấp cao
-

- **Monitor**
  - Cũng là một cấu trúc ngôn ngữ cấp cao như CR
  - Có thể thực hiện bằng semaphore
  - Một module phần mềm bao gồm:
    1. Một hoặc nhiều procedure
    2. Một hoặc nhiều đoạn code khởi tạo
    3. Một hoặc nhiều dữ liệu cục bộ
  - Đặc tính monitor
    - ✓ Biến cục bộ được truy xuất bởi các thủ tục của monitor
    - ✓ Tiến trình vào monitor bằng cách gọi các thủ tục
    - ✓ Chỉ có một process vào monitor tại một thời điểm → **Loại trừ tương hỗ.**
- **Các bài toán đồng bộ kinh điển**
- Bài toán bounder buffer: Semaphore full, empty, mutex

producer

```
do {
        ...
        nextp = new_item();
        ...
        wait(empty);
        wait(mutex);
        ...
        insert_to_buffer(nextp);
        ...
        signal(mutex);
        signal(full);
} while (1);
```

consumer

```
do {
    wait(full)
    wait(mutex);
    ...
    nextc = get_buffer_item(out);
    ...
    signal(mutex);
    signal(empty);
    ...
    consume_item(nextc);
    ...
} while (1);
```

- Bài toán triết gia ăn tối
  - 5 người ngồi lên một bàn ăn, có 5 cái đĩa, 5 chiếc đũa. Mỗi người cần 2 chiếc đũa để ăn.
  - Deadlock: (Người A chờ 1 chiếc đũa, người B chờ 1 chiếc đũa → Hậu quả:)
  - Đói: Một ông sẽ đói
  - Giải pháp
    1. Cho nhiều nhất 4 triết gia vào ngồi cùng một lúc.
    2. Cho phép triết gia cầm đũa khi cả 2 chiếc đũa bên trai và phải phải sẵn sàng.
    3. Triết gia ở vị trí lẻ cầm đũa trái trước sau đó cầm đũa phải. Trong khi đó triết gia chẵn cầm đũa phải trước rồi đũa trái.
- Bài toán đọc ghi
  - Không được cập nhật dữ liệu khi người khác đang đọc.

- Tại một thời điểm nào đó cho phép một writer được sửa nội dung trong CSDL.

---------- Hết ----------