KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN MẠNG MÁY TÍNH VÀ TT DỮ LIỆU

Hướng dẫn Lab 8.1 – Khoá Mutex

Những lệnh phần cứng được giới thiệu ở LAB 7 là nền tảng quan trọng trong lập trình đồng bộ, nhưng chúng có thể phức tạp cho lập trình viên trong nhiều ngữ cảnh đồng bộ khác nhau. Vai trò của Hệ điều hành là tạo ra môi trường lập trình cũng như thư viên tốt để công việc lập trình được thuận lợi.

Trong LAB 8, các kỹ thuật đồng bộ được cung cấp bởi POSIX pthread được giới thiệu. Mở đầu là khoá mutex.

Mục tiêu	**Lý thuyết liên quan**	**Tài nguyên**
Khoá mutex	Ch6.5 Mutex Locks	https://github.com/Trantin84/LAB_IntroOS (mã nguồn ví dụ). Sử dụng image Ubuntu 16 / 18
Đồng bộ thao tác		
Áp dụng cho bài toán P-C		

Yêu cầu sinh viên: Hiểu lý thuyết về khoá mutex. Áp dụng vào các bài toán đồng bộ.

Đánh giá sinh viên: Hỏi đáp các vấn đề lý thuyết. Kỹ năng thực hành. Bài tập.

Yêu cầu nộp bài: các tập tin mã nguồn .c và tập tin khả thực thi .out của các Yêu cầu trong buổi thực hành và Bài tập cuối hướng dẫn trong thời gian cho phép của giảng viên.

Preferences

[1] Abraham Silberschatz, Peter B. Galvin, Greg Gagne, [2018], Operating System Concepts, 10th edition, John Wiley & Sons, New Jersey.

Programming Problems of Chapter 6.

[2] Linux manual page, [2021], on *pthread_mutex_lock (3p)*,

Access https://man7.org/linux/man-pages/man3/pthread_mutex_lock.3p.html

Yêu cầu 1: Mô tả 2 vùng tranh chấp được khoá như thế nào bằng khoá mutex.

Hướng dẫn: Tải về tập tin ***LAB_IntroOS/LAB_8/task1_1_mutexlock.c***, biên dịch và thực thi.

- Dòng 8 là thư viện bao gồm khoá mutex.

- Dòng 16 khai báo khoá.

- Dòng 19 thực hiện khoá, NẾU khoá đang bị sở hữu, tiểu trình phải chờ; ngược lại, tiểu trình chiếm hữu khoá và tiến vào vùng nguy cơ (dòng 20 đến 24).

- Dòng 25 là thao tác trả khoá (sau khi hoàn tất vùng nguy cơ).

- Tại hàm main(), khoá được khởi tạo tại dòng 32 và huỷ bỏ tại dòng 44.

8	`#include<pthread.h>`
16	`pthread_mutex_t lock;`
19	`pthread_mutex_lock( & lock);`
20-24	`//critical section`
25	`pthread_mutex_unlock( & lock);`
32	`pthread_mutex_init( & lock, NULL)`
44	`pthread_mutex_destroy( & lock);`

Yêu cầu 2: Thực hiện đồng bộ bài toán Producer – Consumer bằng khoá mutex.

Hướng dẫn: Tải về tập tin ***LAB_IntroOS/LAB_8/task1_2_PC_mutexlock.c***, biên dịch và thực thi.

- SV xem dòng khai báo khoá, gọi khoá và trả khoá trong từng tiểu trình.

?	- Vùng mã tranh chấp được xác định như thế nào? - Vùng mã đi vào (Entry section) đặt ở đâu? - Vùng mã đi ra (Exit section) đặt ở đâu? - Vùng mã còn lại (Remainer section) có cần phải đặt trong vùng đồng bộ không? Tại sao?	Ch6 p.260	Slide Ch6.9-10

Yêu cầu 3: Ảnh hưởng của thao tác đồng bộ lên hiệu năng tiến trình như thế nào?

Hướng dẫn: Tải về tập tin ***LAB_IntroOS/LAB_8/task1_3_PC_mutexCaution.c***, biên dịch và thực thi. Các dòng mã tương ứng với Yêu cầu 2, tuy nhiên phạm vi vùng nguy cơ (nằm giữa lời gọi khoá và trả khoá) rộng hơn. Hãy đánh giá thời gian chạy và nêu kết luận, thử nghiệm cả với đoạn mã Producer – Consumer không đồng bộ (***LAB_IntroOS/LAB_7/task1_1_PC.c***).

Mỗi dòng thí nghiệm với giá trị đối số: 100, 10000 và 100000, giá trị ghi là thời gian chạy tiến trình.

LAB_7/task1_1_PC.c	Yêu cầu 2	Yêu cầu 3

Bài tập lập trình.

1. Xem bài toán Ước lượng số PI (***LAB_IntroOS/LAB_7/task1_2_PI.c***)
   Hãy thực hiện đồng bộ bằng khoá mutex, so sánh độ chính xác của số PI sau đó. Đồng thời đo thời gian chạy và cho biết sau khi đồng bộ, thời gian chạy đã tăng bao nhiêu %. Có thể cải tiến chương trình để giảm thiểu sự trả giá % thời gian chạy này không?

2. Xem xét lại bài toán Khoá phòng tranh chấp (***LAB_IntroOS/LAB_7/task4_1_RoomTAS.c***), hãy hiện thực lại bằng phương pháp khoá mutex. Hãy nhớ rằng thư viện pthread.h sẽ cần cho khai báo và thao tác trên khoá pthread_mutex_t.