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

Hướng dẫn Lab 8.5 – Khoá Bộ đọc – Bộ ghi.

Nếu trong bài toán P – C quan trọng vấn đề khoá biến đếm chung count, thì trong bài toán W – R lại cần "không khoá" các tiến trình Bộ đọc với nhau, vì các Bộ đọc không gây thay đổi và sai sót dữ liệu. Trên thực tế, mô hình W – R xuất hiện rất nhiều, ví dụ một CSDL điểm số sinh viên thì thao tác ghi chỉ được thực hiện bởi một vài nhân viên Phòng Đại học trong khi người đọc có thể lên đến vài chục ngàn sinh viên.

Bài toán W – R có 3 biến thể và trong LAB này, biến thể 1 được giới thiệu với sự xuất hiện của khoá WR được cung cấp bởi POSIX. Biến thể 2 cũng được nêu ra để so sánh.

Mục tiêu	**Lý thuyết liên quan**	**Tài nguyên**
Bài toán R-W	Ch7.1.2 The Readers–Writers Problem	https://github.com/Trantin84/LAB_IntroOS (mã nguồn ví dụ). Sử dụng image Ubuntu 16 / 18
Khoá POSIX R-W		
Biến thể 2		

Yêu cầu sinh viên: Hiểu lý thuyết về bài toán Bộ đọc – Bộ ghi, hiểu về khoá Wrlock. 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 7.

[2] ProgrammerSought , [2020], Operating system-C language to achieve reader writer problem (reader first),

Access https://programmersought.com/article/49114389739/

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

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

Yêu cầu 1: Mô tả và đồng bộ bài toán Bộ đọc – Bộ ghi (Biến thể 1) bằng semaphore. Tham khảo sách [1] p.290 và Slide Ch7.7-10

Hướng dẫn: Tải về tập tin ***LAB_IntroOS/LAB_8/task5_1_WR_SEM.c***, biên dịch và thực thi.

- Dòng 21 là 2 semaphores cần thiết. Dòng 22 là biến số nguyên đếm số người đọc.

- Dòng 24-28 là cấu trúc một Bộ đọc hoặc ghi. Bao gồm id, thời gian thực thi và thời điểm đến.

- Dòng 39 và 43 (Tương ứng dòng 50 và 54) bảo vệ thao tác thay đổi biến readCount. Tại sao?

- Dòng 40 là khi Bộ đọc đến và Dòng 51 là khi Bộ đọc rời đi.

- Dòng 41 và 42 có ý nghĩa Bộ đọc đầu tiên sẽ khoá vùng bộ nhớ đối với các Bộ ghi đến sau lại.

- Dòng 52 và 53 có ý nghĩa Bộ đọc cuối cùng sẽ mở khoá vùng bộ nhớ để Bộ ghi có thể thao tác (nếu có).

- Dòng 65 và 70 là thao tác đồng bộ của Bộ ghi chiếm hữu vùng nhớ chung.

- Dòng 83 đến 86 khởi tạo dữ liệu cần thiết. Dòng 111 và 112 huỷ bỏ các semaphore đã dùng.

```
21      sem_t RWMutex, mutex;
22      int readCount;

24      struct data {
25        int id;
26        int opTime;
27        int lastTime;
28      };

        //reader
39        sem_wait( & mutex);
40        readCount++;
41        if (readCount == 1)
42          sem_wait( & RWMutex);
43        sem_post( & mutex);

          /* reading is performed */

50        sem_wait( & mutex);
```

```
51        readCount--;
52        if (readCount == 0)
53          sem_post( & RWMutex);
54        sem_post( & mutex);

        //Writer
65        sem_wait( & RWMutex);
          /* writing is performed */
70        sem_post( & RWMutex);

        //main
83        sem_init( & mutex, 0, 1);
84        sem_init( & RWMutex, 0, 1);
86        readCount = 0;
111       sem_destroy( & mutex);
112       sem_destroy( & RWMutex);
```

Thực thi: định dạng dữ liệu được hiểu như sau: Bộ đọc và bộ ghi được truyền vào thông qua lời gọi, mỗi Bộ có 4 tham số `id R/W t_opTime t_lastTime`

```
$ ./wr.out 1 R 3 5 2 W 4 5 3 R 5 2 4 R 6 5 5 W 7 3
```

Có ý nghĩa: Người Đọc #1 đến lúc 3 và thao tác đọc mất 5 đơn vị thời gian. Người Ghi #2 đến lúc 4 và ghi mất 5 đơn vị thời gian. Người đọc #3 đến lúc 5 và đọc mất 2 đơn vị thời gian. …

?	- Người ghi #2 được thao tác khi Người đọc mấy kết thúc? - Người ghi #5 đến có được thao tác ngay không? - Nếu trong khi Người ghi đang chờ mà có Người đọc đến thêm thì sao?	Ch7 p.290	Slide Ch7. 7-10

Yêu cầu 2: Sử dụng khoá WRlock của POSIX thay thế cho semaphore ở bài toán Bộ đọc – Bộ ghi (Biến thể 1) và nhận định về 2 phương pháp.

Hướng dẫn: Tải về tập tin ***LAB_IntroOS/LAB_8/task5_2_WR_WRlock.c***, biên dịch và thực thi.

- Dòng 20 là khai báo khoá rw của pthread.h.

- Dòng 75 là khởi tạo khoá có thuộc tính mặc định. Còn dòng 99 huỷ bỏ khoá sau khi sử dụng.

- Dòng 37 là yêu cầu thao tác vùng nhớ chung với vai trò ĐỌC. Trong khi dòng 56 là yêu cầu của bộ GHI.

- Dòng 44 và 63 trả lại vùng nhớ chung sau khi thao tác.

```
20        pthread_rwlock_t lock;

          //reader
```

```
37        pthread_rwlock_rdlock( & lock);
          /* reading is performed */
44        pthread_rwlock_unlock( & lock);

        //Writer
56        pthread_rwlock_wrlock( & lock);
          /* writing is performed */
63        pthread_rwlock_unlock( & lock);

        //main()
75        pthread_rwlock_init( & lock, NULL);
99        pthread_rwlock_destroy( & lock);
```

- SV so sánh hai đoạn mã đã thực thi, các semaphore đã được thay thế ra sao? Biến số nguyên read_count ở Yêu cầu 1 còn cần thiết không? Vì sao? Trong bài toán này, phương pháp nào có mã gọn hơn, dễ quản lý hơn?

Yêu cầu 3: (Tự học) Mô tả và đồng bộ bài toán Bộ đọc – Bộ ghi (Biến thể 2) bằng semaphore.

Hướng dẫn: Tải về tập tin ***LAB_IntroOS/LAB_8/task5_3_ WR_varation2_SEM.c***, biên dịch và thực thi. Sinh viên tự xem mã, và đánh giá kết quả thực thi, so sánh với Biến thể 1.

Bài tập lập trình.

1. SV nộp các đoạn mã đã thực thi trong các yêu cầu.