تکلیف چهارم سیستم‌های عامل

دکتر زینب زالی

دانشکده مهندسی برق و کامپیوتر

تاریخ تحویل: ۲۶ آذر

۱- یک راه حل صحیح برای مشکل بخش بحرانی (Critical section) باید سه شرط را برآورده کند:

mutual exclusion، progress و bounded waiting

الف) شرط progress را توضیح دهید.

ب) آیا گرسنگی (Starvation) شرط progress را نقض می‌کند؟

ج) هر یک از سه شرط را برای کد زیر چک کنید و اگر برقرار است توضیح دهید چگونه برقرار است در غیر این صورت یک مثال نقض برای موقعیتی که شرط برقرار نمی‌شود بیاورید.

```
P0:
{
    While(True)
    {
        flag[0] = True;
        while(flag[1]);
        /*critical section*/
        flag[0] = False;
        /*remainder_section*/
    }
}

P1:
{
    While(True)
    {
        flag[1] = True;
        while(flag[0]);
        /*critical section*/
        flag[1] = False;
        /*remainder_section*/
    }
}
```

۲- آیا کد زیر برای مسئله Critical section بین دو فرآیند همروند قابل قبول است؟ جواب خود را توضیح دهید و در صورت برقرار نبودن سه شرط لازم، با حداقل تغییر ممکن کد را اصلاح کنید.
(اپراتور AND به معنای اجرای همروند می‌باشد)

```
int turn;
bool flag[2];

proc(int i)
{
    while(true) {
        Compute;
        flag[i] = true;
        turn = (i+1) mod 2;
        while (flag[(i+1) mod 2] && turn == i);
        //critical section
        flag[i] = false;
        //remainder section
    }
}


int main()
{
    turn = 0;
    flag[0] = false;
    flag[1] = false;
    proc(0) AND proc(1);
}
```

۳- کارآمد بودن استفاده از spinlocks و blocking locks را در سناریوهای مختلف با یکدیگر مقایسه کنید و دلیل استفاده از هرکدام را توضیح دهید.

۴- نشان دهید اگر عملیات‌های ()wait و ()signal سمافور به صورت اتمیک اجرا نشوند، ممکن است انحصار متقابل نقض شود.

۵- میخواهیم یک قفل mutex را توسط دستورات اتمیک سخت افزاری پیاده سازی کنیم. این قفل به صورت زیر تعریف میشود:

```
Struct{
    int available;
}lock;
```

"مقدار صفر برای available بیانگر آزاد بودن قفل (در دسترس بودن) و مقادیر یک بیانگر غیر قابل دسترس بودن قفل است."

مشخص کنید چگونه دو تابع زیر(acquire , release) به کمک دستور اتمیک *compare_and_swap* قابل پیاده سازی است (مقادیر اولیه ی لازم را به درستی مشخص کنید).

```
void acquire(lock *mutex), void release(lock *mutex)
int compare_and_swap(int *value, int expected, int new_value)
{
    int temp = *value;
    if(*value==expected)
        *value=new_value;
    return temp;
}
```

۶- کد زیر برای حل مسأله کلاسیک تولیدکننده-مصرف‌کننده داده شده است. count متغیری سراسری با مقدار اولیه صفر و N تعداد خانه‌های بافر است. مشکل این الگوریتم چیست؟ برای پاسخ خود دلیل بیاورید و با استفاده از متغیر شرطی مشکل آن را برطرف کنید (توضیح دهید متغیر شرطی چگونه به این امر کمک می‌کند).

```
void producer(void) {
    while (TRUE) {
        produce_item();  // Produce an item
        if (count == N) sleep();
        enter_item();  // Place the produced item in the buffer
        count = count + 1;
        if (count == 1)
            wakeup(consumer);
    }
}

void consumer(void) {
    while (TRUE) {
        if (count == 0) sleep();
        remove_item();  // Remove an item from the buffer
        count = count - 1;
        if (count == N-1)
            wakeup(producer);
        consume_item();  // Consume the item
    }
}
```

۷- سه فرآیند P1 و P2 و P3 در حالت اجرا (running) هستند و طبق جدول زیر عملیات P(wait) یا V(signal) روی سمافور S (با مقدار اولیه ۱) اجرا می‌شـــود. فرآیندی که شـــماره کوچک‌تری دارد برای راه‌اندازی اولویت دارد. حالت این ســـه فرآیند پس از اجرای دستورات زیر چیست؟ مرحله به مرحله حالت فرآیندها و مقدار سمافور را ذکر کنید.

"زمان های ذکر شده، زمان اجرای هرکدام از فرآیندهاست."

زمان(ms)	فرآیند	دستور
۱۰۰	P1	P(s)
۲۰۰	P1	P(s)
۳۰۰	P2	V(s)
۴۰۰	P3	P(s)
۵۰۰	P1	P(s)
۶۰۰	P2	V(s)
۷۰۰	P2	P(s)
۸۰۰	P1	V(s)
۹۰۰	P1	V(s)

۸- یک غار تاریخی با نقاشی‌های دیواری باشکوه دارای ورودی بسیار باریکی است که تنها می‌تواند به یک بازدیدکننده اجازه ورود/خروج در هر زمان بدهد. برای اطمینان از اینکه هرگز بیش از ۱۵ نفر در غار نباشند، شبه کد زیر را تکمیل کنید. کد توسط هر فرآیند بازدید کننده غار اجرا می شود. بخش های مشخص شده را با مقادیر اولیه سمافور یا فراخوانی های سمافور (wait() یا signal()) پرکنید.

```
semaphore s1 = // your answer here;
semaphore s2 = // your answer here;

Cave_exploration () {
    // your answer here(may contain one or two calls)
    Enter_the_cave ();
    // your answer here(may contain one or two calls)
    Look_at_the_paintings ();
    // your answer here(may contain one or two calls)
    Exit_the_cave ();
    // your answer here(may contain one or two calls)
}
```

۹- مسئله خوانندگان نویسندگان را با اولویت **نویسندگان به خوانندگان** در نظر بگیرید. می‌خواهیم lockهای مخصوصی با نام reader-writer locks داشته باشیم که بتوان در چنین شرایطی از آنها استفاده کرد. سودوکدهای لازم را برای پیاده‌سازی توابع writeLock، writeUnlock، readLock و readUnlock بنویسید. در پیاده‌سازی خود از mutex و conditional variable استفاده کنید (از سمافور استفاده نکنید).

۱۰- میزبان یک مهمانی، N>2 نفر مهمان را به خانه دعوت کرده است. میزبان نمی‌خواهد چندین دفعه در خانه را برای ورود مهمانان باز کند. N مهمان برای یکدیگر صبر می‌کنند و به یک‌باره وارد می‌شوند. میزبان و مهمانان با برنامه چند نخی پیاده سازی می‌شوند.
میزبان برای ورود همه مهمانان صبر می‌کند و سپس openDoor() را فراخوانی می‌کند و یک متغیر شرطی را سیگنال می‌کند. مهمانان باید برای ورود N نفر و باز شدن در صبر کنند و enterHouse() را فراخوانی کنند.

تنها با استفاده از متغیرهای تعریف شده در کد میزبان، کد مهمان را بنویسید.

```
#host
lock(m)
while (guest_count < N) wait(cv_host, m)
openDoor()
signal(cv_guest)
unlock(m)
```

نکات تکمیلی

۱. فرمت نام‌گذاری تکلیف ارسالی باید به صورت زیر باشد:

HW4_LastName_StudentID که LastName نام‌خانوادگی شما و StudentID شماره دانشجویی شما است.

۲. در صورت مشاهده تقلب، نمرات هم مبدا کپی و هم مقصد آن صفر لحاظ می‌شود.

۳. در صورت وجود هرگونه ابهام میتوانید با دستیاران آموزشی از طریق تلگرام در ارتباط باشید.

- Hamidreza Baghiani
- Ashkan Hafezi