操作系统 实验二

Linux下多线程编程实践

胡燕
大连理工大学 软件学院

实践目标

目标：完成Linux环境下多线程实践

Linux环境下多线程实践

要求：

熟悉Linux环境下pthread编程，完成示例代码的编译与执行；

使用pthread库编写多线程程序，实现经典的生产者消费者问题；

POSIX API

Introduction to Several POSIX API

02

pthread_create

pthread_create函数说明

```
int pthread_create(pthread_t *tid,
                   const pthread_attr_t *attr,
                   void *(function)(void *),
                   void *argument);
```

tid: 代表新建线程的数据结构，代表新建立的线程

attr: 新建立线程的属性结构，一般填NULL即可

function: 新创建的线程所要执行的函数

argument: 传递给线程的参数

建立一个无需同步的多线程程序：nosync-ex.c

```
#include <stdio.h>
#include <pthread.h>

int sum = 0;

void thread(void) {
    int i;
    for (i = 0; i< 1000000; i++)
        sum += 1;
}
```

pthread_create

建立一个无需同步的多线程程序：nosync-ex.c

```
int main(void) {
    pthread_t tid1, tid2;
    pthread_create(&tid1, NULL, thread, NULL);
    pthread_create(&tid2, NULL, thread, NULL);
    pthread_join(tid1, NULL);
    pthread_join(tid2, NULL);
    printf ("1000000 + 1000000 = %d\n", sum);
    return (0);
}
```

pthread_create

建立一个无需同步的多线程程序：nosync-ex.c

```
int main(void) {
    pthread_t tid1, tid2;
    pthread_create(&tid1, NULL, thread, NULL);
    pthread_create(&tid2, NULL, thread, NULL);
    pthread_join(tid1, NULL);
    pthread_join(tid2, NULL);
    printf ("1000000 + 1000000 = %d\n", sum);
    return (0);
}
```

执行如下命令，进行编译

```
gcc –o nosync-ex nosync-ex.c -lpthread
```

执行编译后的可执行程序

```
time ./nosync-ex
```

建立一个无需同步的多线程程序：nosync-ex.c

```
int main(void) {
    pthread_t tid1, tid2;
    pthread_create(&tid1, NULL, thread, NULL);
    pthread_create(&tid2, NULL, thread, NULL);
    pthread_join(tid1, NULL);
    pthread_join(tid2, NULL);
    printf ("1000000 + 1000000 = %d\n", sum);
    return (0);
}
```

执行结果：

```
1000000 + 1000000 = 994690

real    0m0.013s
user    0m0.018s
sys     0m0.005s
```

结果讨论：
为何结果不正确？

Mutex相关函数说明

```
#include <pthread.h>
int pthread_mutex_init(pthread_mutex_t *restrict mutex,
                       const pthread_mutexattr_t *restrict attr);
int pthread_mutex_lock(pthread_mutex_t *mutex);
int pthread_mutex_unlock(pthread_mutex_t *mutex);
```

attr: 设定该mutex的属性，通常使用预设属性，此时设为NULL即可

Mutex的使用(mutex-ex.c)

```
#include <stdio.h>
#include <pthread.h>

int sum = 0;
pthread_mutex_t mutex;

void thread(void) {
    int i;
    for (i = 0; i< 1000000; i++) {
        pthread_mutex_lock(&mutex);
        sum += 1;
        pthread_mutex_unlock(&mutex);
    }
}
```

Mutex的使用(mutex-ex.c)

```
int main(void) {
    pthread_t tid1, tid2;
    pthread_mutex_init(&mutex, NULL);

    pthread_create(&tid1, NULL, thread, NULL);
    pthread_create(&tid2, NULL, thread, NULL);
    pthread_join(tid1, NULL);
    pthread_join(tid2, NULL);
    printf ("1000000 + 1000000 = %d\n", sum);
    return (0);
}
```

```
gcc –o mutex-ex mutex-ex.c -lpthread
time ./mutex-ex
```

```
1000000 + 1000000 = 2000000

real    0m0.211s
user    0m0.240s
sys     0m0.180s
```

Semaphore相关函数的说明

```
#include <semaphore.h>
int sem_init(sem_t * sem, int pshared, unsigned int value);

int sem_wait(sem_t * sem);
int sem_post(sem_t * sem);
```

value: 信号量的初始值

信号量的使用

Semaphore的使用(sem-ex.c)

```
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>

int sum = 0;
sem_t sem;

void thread(void) {
    int i;
    for (i = 0; i< 1000000; i++) {
        sem_wait(&sem);
        sum += 1;
        sem_post(&sem);
    }
}
```

信号量的使用

Semaphore的使用(sem-ex.c)

```
int main(void) {
    pthread_t tid1, tid2;
    sem_init(&sem, 0, 1);

    pthread_create(&tid1, NULL, thread, NULL);
    pthread_create(&tid2, NULL, thread, NULL);
    pthread_join(tid1, NULL);
    pthread_join(tid2, NULL);
    printf ("1000000 + 1000000 = %d\n", sum);
    return (0);
}
```

```
gcc –o sem-ex sem-ex.c -lpthread
time ./sem-ex
```

```
1000000 + 1000000 = 2000000

real    0m0.306s
user    0m0.363s
sys     0m0.224s
```

实践任务

Lab Assignments

03

任务内容

任务1：熟悉API

要求：

熟悉Linux环境下pthread编程，完成示例代码（nosync-ex.c，mutex-ex.c，sem-ex.c）的编译与执行。

任务2：实现生产者消费者问题

要求：

基于示例中涉及到的线程同步API，实现生产者消费者问题（具体的生产、消费操作可自行设计）；

任务3：实现其他同步问题

可选

要求：

基于示例中涉及到的线程同步API，实现其他的同步问题（如理发师问题）；

任务内容

任务3：实现其他同步问题

可选

要求：

基于示例中涉及到的线程同步API，实现其他的同步问题（如理发师问题）；

理发师问题描述

有一个理发师，一把理发椅和N把供等候理发的顾客坐的椅子。
如果没有顾客，则理发师便在理发师的椅子上睡觉；
当一个顾客到来时，必须唤醒理发师进行理发；如果理发师正在理发时又有顾客来到，则如果有空椅子可做，他就坐下来等，如果没有空椅子，他就离开。
请为理发师和顾客各编写一段程序（伪码）描述他们的行为，并用信号量正确控制他们之间的并发协作。

提交内容

提交报告要求

要求：

用Markdown文档格式记录实验；

实验过程中的关键步骤需要截图，并配以必要文字说明；

基础实验当堂检查，报告打包后在学习通中提交（会以作业的形式在学习通中发布）

谢 谢!
Thank you!