操作系统实验一

Nachos01

姓	名	邓语苏
学	号	22920212204066
日	期	2024 年 3 月 14 日
学	院	信息学院

Nachos01

目录

1 安装 Nachos

1. 将压缩包上传至 UNIX 服务器并解压

代码 1 解压 Nachos 文件包

```
tar -xvzf nachos-linux64.tar.gz
```

2. 进入 code 文件夹后运行 make
3. 进入 threads 文件夹后运行 make depend
4. 运行./nahcos 测试文件运行结果如图 1

```
-bash-3.2$ ./nachos
*** thread 0 looped 0 times
*** thread 1 looped 0 times
*** thread 0 looped 1 times
*** thread 1 looped 1 times
*** thread 0 looped 2 times
*** thread 1 looped 2 times
*** thread 0 looped 3 times
*** thread 1 looped 3 times
*** thread 0 looped 4 times
*** thread 1 looped 4 times
No threads ready or runnable, and no pending interrupts.
Assuming the program completed.
Machine halting!

Ticks: total 130, idle 0, system 130, user 0
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
```

图 1: nachos

2 Makefile 文件之间的关系

此处的 Makefile 指/code/threads 路径下的 Makefile 文件

- **Makefile** 这是项目的主要 Makefile 文件。

代码 2 /code/threads/Makefile

```
DEFINES = -DTHREADS # 定义了一个宏 THREADS
INCPATH = -I../threads -I../machine
HFILES = $(THREAD_H)  # 定义了所有头文件
CFILES = $(THREAD_C)    # 定义了所有源文件
C_OFILES = $(THREAD_O)  # 定义了所有目标文件

# 引用 code/Makefile.common、code/Makefile.dep 文件
include ../Makefile.common、
include ../Makefile.dep

main.o: ../threads/main.cc ../threads/copyright.h ../threads/utility.h \
../threads/bool.h ../machine/sysdep.h /usr/include/stdio.h \
/usr/include/features.h /usr/include/sys/cdefs.h \
/usr/include/gnu/stubs.h \
/usr/lib/gcc-lib/i686-pc-linux-gnu/3.3.5/include/stddef.h \
...
```

- **Makefile.common**

  这个文件包含了变量定义，可以被项目中的其他 Makefile 文件引用。

- **Makefile.dep** 这个文件主要用于管理文件之间的依赖关系。

3 nachos 中线程的概念

阅读 nachos-3.4/code/threads/threadtest.cc，可知

1. nachos 中的线程为用户级线程

   - 头文件：代码中包含的头文件”copyright.h” 和”system.h”，是为用户程序提供的
   - 线程调度函数：多个函数使用了 `currentThread->Yield()`，这是一个线程调度函数，通常在用户空间中使用。
   - 线程创建：使用 new Thread(...) 来创建一个线程对象，这也是在用户级别进行的操作

4 初步尝试修改 Nachos 源代码

4.1 实验要求

- 在 nachos-3.4/code/threads/目录下生成 hello.h,hello.cc 两个文件。
- 在 heIo.h 中写入 hello 函数的声明如 void hello();
- 在 helo.cc 中写入 hello 函数的实现，要求能输出 Hello,nachos!’m< 你的学号 >
- 修改 ThreadTest 函数，在其中加入对 hello 函数的调用
- 重新编译 nachos, 使其运行结果能够显示 hello 函数的输出结果。

4.2 实验步骤

1. 将编写好 hello.cc 和 hello.h 拷贝到/code/threads 中
2. 修改/code/Makefile.common

   THREAD_H、THREAD_C、THREAD_O中添加对应的 hello.cc/.o/.h 的路径
3. 修改/code/threads/Makefile

   添加 hello.o 的规则
4. 修改/code/threads/threadtest.cc

   - 在文件首部声明#include ”hello.h”
   - 在 ThreadTest(); 函数的开始处调用 hello(); 函数
5. 重新编译

   在/code/threads/中依次执行 make depend 和 make

6. 验证结果

   修改后运行./nachos，结果如图 2则代表修改成功!

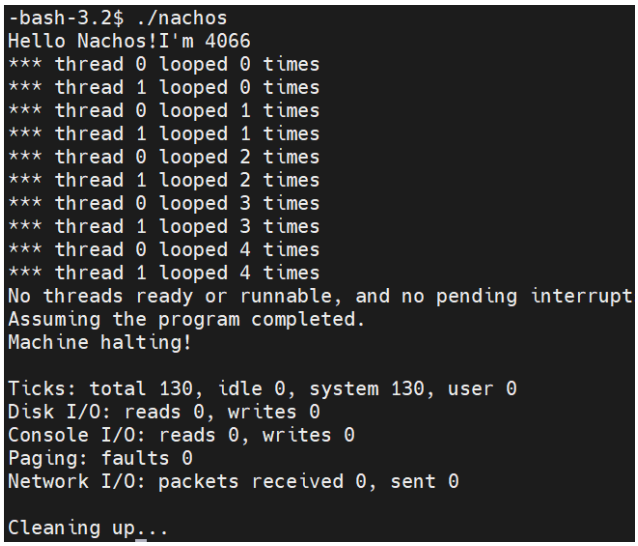

图 2: Hello Nachos!

5 实现双向链表

1. 编写 dllist.h

   内容提供在 nachos-labs.pdf 的 3.1.1 章节;

2. 完成 dllist.cc 和 dllist-driver.cc

3. 修改 threadtest.cc

   实现的功能是，当 testnum=2 时，调用 ThreadTest2 函数。ThreadTest2 函数会使用 for 循环创建 threadnum 个线程，每个线程执行 SimpleThread2() 函数。

   - 在循环中，通过 `new Thread(name)`创建一个新线程对象，其中 name 为线程名。
   - 使用`t->Fork(SimpleThread2,i)`将线程 t 绑定到函数 SimpleThread2,并传递参数 i 到 SimpleThread2.
   - 最后主线程也执行 SimpleThread2() 函数，传递参数 0，作为主线程的编号

代码 3 /code/threads/threadtest.cc SimpleThread2 函数

```
void
SimpleThread2(int which) {
       Insert(L, itemnum, which);
       Remove(L, itemnum, which);
}
```

代码 4 /code/threads/threadtest.cc ThreadTest2 函数

```
void
ThreadTest2(){
    L = new DLList();
    for (int i = 1; i < threadnum; ++i)
    {
        memset(name, 0, sizeof(name));
        snprintf(name, 64, "thread %d", i);
        Thread* t = new Thread(name);
        t->Fork(SimpleThread2, i);
    }
    SimpleThread2(0);
}
```

代码 5 /code/threads/threadtest.cc ThreadTest() 函数

```
void
ThreadTest(){
    hello();
    switch (testnum) {
    case 1:
       ThreadTest1();
       break;
    case 2:
       ThreadTest2();       // test for doubly-linked list goes here
       break;
    default:
       printf("No test specified.\n");
       break;
    }
}
```

4. 修改 main.cc

其中 testnum 用来选择测试函数，1 为 ThreadTest1()，2 为 ThreadTest2()。threadnum 控制在 ThreadTest2() 函数中，for 循环建立线程的个数 itemnum 控制在 Insert 和 Remove 函数中插入和删除数据项的个数

代码 6 /code/threads/main.cc 关键代码

```
extern int testnum;extern int threadnum;extern int itemnum;extern int errorType;
...
int main(int argc, char** argv) {
        for (argc--, argv++; argc > 0; argc -= argCount, argv += argCount) {
                //   ./nachos -q 2 -t [num_threads] -n [num_items] -e [type_of_error]
                argCount = 1;
                switch (argv[0][1]) {
                        case 'q':
                                testnum = atoi(argv[1]);
                                argCount++;
                                break;
                        case 't':                // number of threads
                                threadnum = atoi(argv[1]);
                                argCount++;
                                break;
                        case 'n':
                                itemnum = atoi(argv[1]);
                                argCount++;
                                break;
                        default:
                                testnum = 1;
                                break;
                }
        }
        ThreadTest();
....
```

5. 将编写好 dllist.cc 、dllist-driver.cc、dllist.h 拷贝到/code/threads 中

6. 修改/code/Makefile.common

   THREAD_H、THREAD_C、THREAD_O中添加对应的 dllist.cc/.o/.h、dllist-driver.cc/.o/.h 的路径

7. 修改/code/threads/Makefile

   添加 dllist.o、dllist-driver.o 的规则

8. 重新编译在/code/threads/中依次执行 make depend 和 make

9. 验证结果

   修改后运行./nachos -q 2 -t 3 -n 2 运行 ThreafTest2 函数，使用 for 循环创建 3 个线程，每个线程插入 2 个数据项并删除

```
-bash-3.2$ ./nachos -q 2 -t 3 -n 2
Hello Nachos!I'm 4066
Thread 0 inserts: 11
Thread 0 inserts: 28
Thread 0 removes: 11
Thread 0 removes: 28
Thread 1 inserts: 19
Thread 1 inserts: 14
Thread 1 removes: 14
Thread 1 removes: 19
Thread 2 inserts: 1
Thread 2 inserts: 32
Thread 2 removes: 1
Thread 2 removes: 32
No threads ready or runnable, and no pending interrupts.
Assuming the program completed.
Machine halting!

Ticks: total 50, idle 0, system 50, user 0
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
```

图 3: |./nachos -q 2 -t 3 -n 2|

10. 修改 Nachos 源代码体验并发程序问题

在实验中，我尝试了不同的 Yield 位置之后，发现了四类错误，它们分别是

```
-bash-3.2$ ./nachos -q 2 -t 3 -n 2
Hello Nachos!I'm 4066
Thread 0 inserts: 11
Thread 0 inserts: 28
Thread 1 inserts: 19
Thread 1 inserts: 14
Thread 2 inserts: 1
Thread 2 inserts: 32
Thread 0 removes: 1
Thread 0 removes: 11
Thread 1 removes: 14
Thread 1 removes: 19
Thread 2 removes: 28
Thread 2 removes: 32
No threads ready or runnable, and no pending interrupts.
Assuming the program completed.
Machine halting!

Ticks: total 80, idle 0, system 80, user 0
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
```

链表混乱

```
-bash-3.2$ ./nachos -q 2 -t 3 -n 2
Hello Nachos!I'm 4066
Thread 0 inserts: 11
Thread 0 inserts: 28
Thread 1 inserts: 19
Thread 1 inserts: 14
Thread 2 inserts: 1
Thread 2 inserts: 32
Thread 0 removes: 11
*** glibc detected *** ./nachos: double free or corruption (fasttop): 0x0860b1e0
 ***
```

Double Free 问题

图 4: 错误信息-1

```
-bash-3.2$ ./nachos -q 2 -t 3 -n 2
Hello Nachos!I'm 4066
Thread 0 inserts: 11
Thread 0 inserts: 28
Thread 0 removes: 11
Thread 0 removes: 28
段错误
```

段错误

```
-bash-3.2$ ./nachos -q 2 -t 3 -n 2
Hello Nachos!I'm 4066
Thread 1 inserts: 28
Thread 1 inserts: 19
Thread 1 removes: 11
Thread 1 removes: 19
Thread 0 inserts: 11
Thread 2 inserts: 14
Thread 2 inserts: 32
Thread 2 removes: 14
Thread 2 removes: 28
Thread 0 inserts: 1
Thread 0 removes: 1
*** glibc detected *** ./nachos: free(): invalid next size (fast): 0x0956d1f8 **
*
```

invalid next size 问题

图 5: 错误信息-2