# 操作系统课程设计报告

矩阵乘法

## 1 项目简介

### 1.1 项目目的

- 学习使用Pthread API控制多线程。

### 1.2 项目要求

对于矩阵乘法中矩阵积的每一个元素，使用独立的线程进行运算，最终使用主线程打印结果。

## 2 项目实现

### 2.1 系统版本

- Ubuntu：16.04
- Linux内核：4.15.13

### 2.2 实现过程

#### 2.2.1 创造线程并传递参数

为每一个$C[i][j]$声明线程id:

pthread_t multi[M][N];

使用pthread_creat函数为每一个$C[i][j]$创造一个独立线程，函数原型为：

int pthread_creat(pthread_t *tidp,const pthread_attr_t *attr, (void*)(*start_rtn)(void*),void *arg);

第一个参数为pthread_t类型的指向线程id的指针，第二个参数设置线程属性，第三个参数是线程运行函数的起始地址，最后一个参数是运行函数的参数。

本项目中第一个参数应设为线程id的引用&multi[$i$][$j$]，第二个参数可为NULL，第三个参数为下面实现的线程函数名(计算$C[i][j] = \sum_{n=1}^{K} A_{i,n} \times B_{n,j}$并返回$C[i][j]$)，第四个参数为结构体指针(传递$i$,$j$的值)。

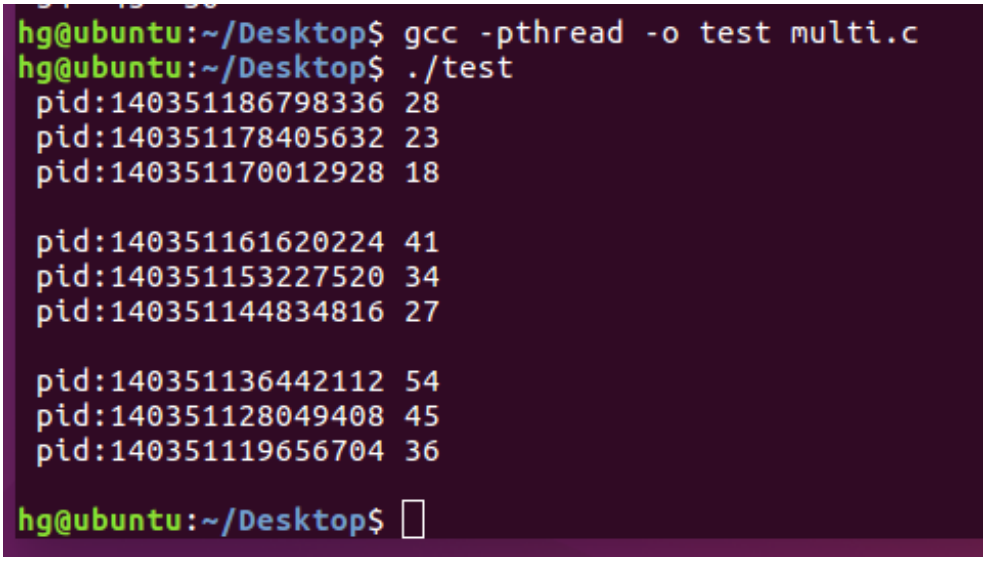


图 1: 线程运算结果

#### 2.2.2 实现线程函数

在计算$C[i][j] = \sum_{n=1}^{K} A_{i,n} \times B_{n,j}$后，使用pthread_exit函数退出线程并返回计算结果，函数原型为

void pthread_exit(void* retval);

将int型的结果转换为void*回传。

#### 2.2.3 等待线程结束

在主线程for循环中使用pthread_join函数等待每一个线程结束并得到该线程返回值，函数原型为：

int pthread_join(pthread_t thread, void **retval);

将线程id(multi[$i$][$j$])与结果打印出来,注意结果要转回int型。

### 2.3 项目中出现的问题及解决方案

1. void*类型转换

   在Pthread API中，pthread_creat中的函数参数需强制转换为void*类型，在线程函数中转回struct v *类型即可。

   pthread_exit中返回值(int)也需强制转换为void*类型，而在pthread_join中使用一个void**类型的指针存储返回值，因此可定义void*型数组void* result[$i$][$j$], 用每个元素的引用&result[$i$][$j$]作为地址参数储存返回值。而在打印返回值时应将void*类型的result[$i$][$j$]转回int型。

   接口函数参数对应关系(参数类型，指针与值)非常复杂，要仔细对照避免产生错误。

2. warning

在返回值的传递与类型转换部分出现warning:

warning: cast to pointer from integer of different size [-Wint-to-pointer-cast]

这是由于void*类型与int型互相转换过程中，系统认为长度可能为32为也可能为64位，可使用(intptr_t)类型保证不会发生数据丢失，从而消除报错。

# 3 总结及反思

通过这次项目实现我进一步熟悉了线程的工作原理与Pthread API函数的使用。