多线程编程原理及实现

邓豪

2017年12月17日星期日

目录

多线程编程原理

- 线程：程序运行的基本单位
- 线程池：管理一个任务队列，一个线程队列，然后每次取一个任务分配给一个线程去做，循环往复
- 多线程编程：核心是线程池的管理，目的是实现线程并发

多线程并发编程典型案例

- 生产者消费者模型
- 单生产者单消费者模型
- 单生产者多消费者模型
- 多生产者单消费者模型
- 多生产者多消费者模型

线程编程-创建线程

包含头文件<thread>；

定义线程对象t，线程对象负责管理以hello()函数作为初始函数的线程；

join()等待线程函数执行完成

```cpp
#include <iostream>
#include <thread>

void hello()
{
    std::cout << "Hello world" << std::endl;
}

int main()
{
    std::thread t(hello);
    t.join(); // 没有这句话，会Debug Error的
    return 0;
}
```

线程编程-线程参数

- 一、以函数作为参数
- 二、以函数对象作为参数
- 三、以类的成员函数作为参数
- 四、以Lambda对象作为参数

线程编程-关键成员函数

- 一、detach() 子线程与父线程分离
- 二、move() 转移线程所有权
- 三、hardware_concurrency() 可运行线程数量
- 四、id 线程识别号

多线程编程-锁与互斥

- 问题：管理一个任务队列，一个线程队列，然后每次取一个任务分配给一个线程去做，循环往复，有什么不对？

多线程编程-锁与互斥

mutex 又称互斥量，C++ 11中与 mutex 相关的类（包括锁类型）和函数都声明在 头文件中，所以如果需要使用 std::mutex，就必须包含 头文件，mutex中包含以下：

mutex系列类

std::mutex，最基本的 mutex类

std::recursive_mutex，递归 mutex类

std::time_mutex，定时 mutex类。

std::recursive_timed_mutex，定时递归 mutex类

lock 类

std::lock_guard，与 mutex RAII 相关，方便线程对互斥量上锁

std::unique_lock，与 mutex RAII 相关，方便线程对互斥量上锁，但提供了更好的上锁和解锁控制

其他类型

std::once_flag

std::adopt_lock_t

std::defer_lock_t

std::try_to_lock_t

函数

std::try_lock()，尝试同时对多个互斥量上锁。

std::lock()，可以同时对多个互斥量上锁。

std::call_once()，如果多个线程需要同时调用某个函数，call_once 可以保证多个线程对该函数只调用一次。

多线程编程-线程池

- 线程管理器
- 工作线程
- 任务接口
- 任务队列

相关资料

- C++11 编程指南
- https://segmentfault.com/a/1190000006691692
- http://blog.csdn.net/jonathan321/article/details/52679471
- https://www.ibm.com/developerworks/cn/aix/library/au-aix-openmp-framework/

Thank You !