竞态条件与临界区

原文链接 **作者：**Jakob Jenkov **译者：**He Jianjun **校对：**丁一

在同一程序中运行多个线程本身不会导致问题，问题在于多个线程访问了相同的资源。如，同一内存区（变量，数组，或对象）、系统（数据库，web services等）或文件。实际上，这些问题只有在一或多个线程向这些资源做了写操作时才有可能发生，只要资源没有发生变化,多个线程读取相同的资源就是安全的。

多线程同时执行下面的代码可能会出错：

```
public class Counter {
    protected long count = 0;
    public void add(long value){
        this.count = this.count + value;
    }
}
```

想象下线程A和B同时执行同一个Counter对象的add()方法，我们无法知道操作系统何时会在两个线程之间切换。JVM并不是将这段代码视为单条指令来执行的，而是按照下面的顺序：

```
从内存获取 this.count 的值放到寄存器
将寄存器中的值增加value
将寄存器中的值写回内存
```

观察线程A和B交错执行会发生什么：

```
    this.count = 0;
A:  读取 this.count 到一个寄存器 (0)
B:  读取 this.count 到一个寄存器 (0)
B:  将寄存器的值加2
B:  回写寄存器值(2)到内存. this.count 现在等于 2
A:  将寄存器的值加3
A:  回写寄存器值(3)到内存. this.count 现在等于 3
```

两个线程分别加了2和3到count变量上，两个线程执行结束后count变量的值应该等于5。然而由于两个线程是交叉执行的，两个线程从内存中读出的初始值都是0。然后各自加了2和3，并分别写回内存。最终的值并

不是期望的5，而是最后写回内存的那个线程的值，上面例子中最后写回内存的是线程A，但实际中也可能是线程B。如果没有采用合适的同步机制，线程间的交叉执行情况就无法预料。

竞态条件 & 临界区

当两个线程竞争同一资源时，如果对资源的访问顺序敏感，就称存在竞态条件。导致竞态条件发生的代码区称作临界区。上例中add()方法就是一个临界区,它会产生竞态条件。在临界区中使用适当的同步就可以避免竞态条件。

夕水溪下
2013/03/05 4:46下午

如果有多个线程试图同时访问临界区，那么在有一个线程进入后其他所有试图访问此临界区的线程都会被挂起（阻塞），并一直持续到进入临界区的线程离开。临界区在被释放后，其他线程可以继续抢占，但是访问临界区的代码时始终是原子的，这就是同步。

小Q
2013/03/12 7:32下午

临界区的代码是原子的吗?

假设 有一个临界区的方法

test(){

a=a+1;

}

假设线程a 刚刚进入 test 方法,线程b 在另一个方法里 对 a 进行了输出,请问 输出的结果是不是一定的?

原子性是指,cpu在执行这一段代码的时候不能被打断,不过,临界区的代码是 同时只能被一个线程访问,而不是不能执行临界区的代码的cpu 不能被打断.

不知道理解的对不对?

丁一
2013/03/12 9:18下午

我们把对共享内存进行访问的程序片段称作临界区。准确的说，临界区的代码访问，我们需要的是互斥。

匿名
2013/03/12 11:03下午

上面的a=a+1例子必须是原子的。

匿名
2013/03/28 12:04下午

no,no,a=a+1不是原子的，只有对除long和double外的基本类型进行简单的赋值或返回操作，才属于原子的。比如a=1;

小Q
2013/03/13 1:50下午

互斥和原子还是有差别的吧

夕水溪下
2013/03/15 9:18下午

个人认为互斥和原子这两个概念表达的都是同一个意思：当一个对象被线程修改的时候，可以阻止另一个线程观察到对象内部的不一致状态。

小Q
2013/03/12 7:33下午

而不是不能执行临界区的代码的cpu 不能被打断 =====> 而不是正在执行临界区的代码的cpu 不能被打断

打错了,这里更正一下

小Q
2013/03/21 3:03下午

> 夕水溪下 :
>
> 个人认为互斥和原子这两个概念表达的都是同一个意思：当一个对象被线程修改的时候，可以阻止另一个线程观察到对象内部的不一致状态。

相当不同意….有没有别的同学发表看法?

bsniao
2014/01/13 5:47下午

互斥是这个资源只有我用，你不许用；

原子是我一口气把活干完，不间断；

根本就是两码事吧，即便起到的效果是一样的，也不能说表达的是同一个意思

夕水溪下
2014/01/13 6:57下午

嗯，是的，之前的理解有些问题。

匿名
2014/04/09 5:43下午

> bsniao :
>
> 互斥是这个资源只有我用，你不许用；
>
> 原子是我一口气把活干完，不间断；
>
> 根本就是两码事吧，即便起到的效果是一样的，也不能说表达的是同一个意思

javaer
2014/04/09 5:44下午

> bsniao :
>
> 互斥是这个资源只有我用，你不许用；
>
> 原子是我一口气把活干完，不间断；
>
> 根本就是两码事吧，即便起到的效果是一样的，也不能说表达的是同一个意思

fangqiang08
2014/09/12 4:45下午

> 匿名 :
>
> no,no,a=a+1不是原子的，只有对除long和double外的基本类型进行简单的赋值或返回操作，才属于原子的。
>
> 比如a=1;

a=a+1应该不是原子的。 但是long和double的赋值和返回操作不是原子吗？为什么？

> dongzh
> 2014/10/07 3:10下午
>
> JVM规范允许对64位的操作分为两次32位操作进行。

可乐
2014/10/22 11:30上午

你好，Java虚拟机的解释执行引擎是基于“栈”的，而不是寄存器，那么对于this.count=this.count+value,这个操作就不应该是存入寄存器中，而是操作数数栈

> 程序小院
> 2015/04/09 11:49上午
>
> 人家也没说存到寄存器中啊，简单数据结构是存放到栈中，栈也是内存的一部分，而数据的运算是需要在寄存器中计算的，你把那段话中的内存换成栈就可以了

u5828361
2016/02/17 3:12下午

对于如下这段话有疑问

“从内存获取 this.count 的值放到寄存器

将寄存器中的值增加value

将寄存器中的值写回内存”

这里寄存器应该不是指pc硬件上的寄存器吧，应该指的是线程的工作内存吧。从其他地方了解到，java线程的工作内存也是一个虚拟概念，与pc的真实寄存器和高速缓存相关。

所以这里想求解下这些细节。即JVM是如何管理线程数据与CPU寄存器和高速缓存 等关联的。 或者求相关文章推荐。

hunter
2016/06/01 1:50下午

推荐你看下“深入理解Java虚拟机：JVM高级特性与最佳实践（最新第二版）”这本书

huduku
2018/07/02 3:33下午

从数据库中查出来1000条数据，然后进行遍历并更新。

请问这1000条数据属于临街资源吗？