CHAPTER 01

第四章 并发计算方案

并发归集方案

1 4.1 Atomic原子类

2 4.2 高性能累加器

4.1 Atomic原子类

Atomic原子类

位于 java.util.concurrent.atomic 包

java.util.concurrent.atomic.AtomicBoolean
java.util.concurrent.atomic.AtomicInteger
java.util.concurrent.atomic.AtomicIntegerArray
java.util.concurrent.atomic.AtomicLong
java.util.concurrent.atomic.AtomicLongArray
java.util.concurrent.atomic.AtomicReference
java.util.concurrent.atomic.AtomicReferenceArray

利用volatile支持可见性；利用Unsafe方法，采用CAS模式，支持原子性。

Atomic原子类 企业案例

案例1

批量处理一批订单数据，采用线程池并行的去处理这些数据，最终要记录成功数量、失败数量。

案例2

批量处理一批订单数据，采用线程池并行的去处理这些数据，最终要记录每一个类型的订单的数量。

HEADLINE HEADLINE

4.2 高性能累加器

高性能累加器

同样位于 java.util.concurrent.atomic 包

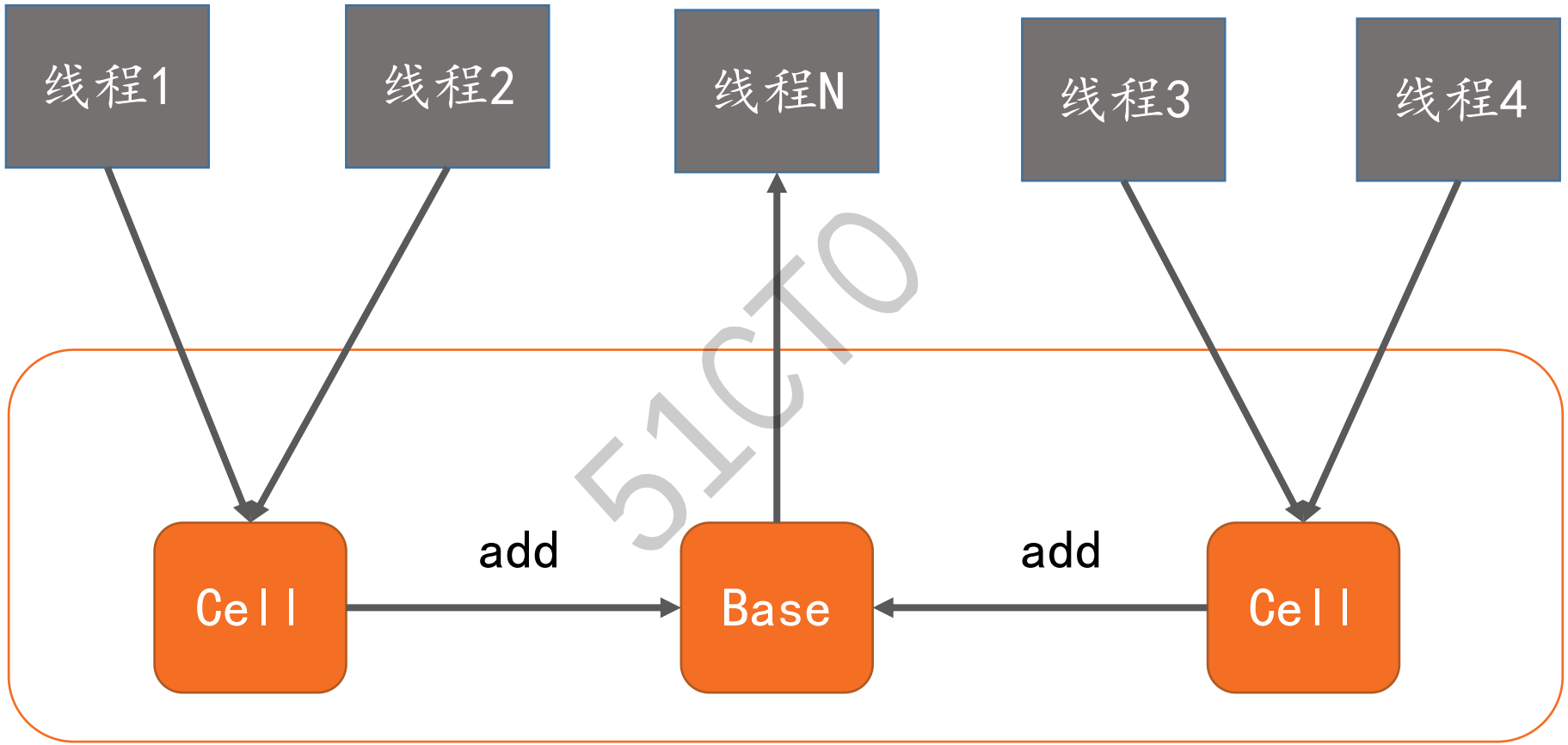
java.util.concurrent.atomic.LongAdder

java.util.concurrent.atomic.DoubleAdder

AtomicLong存在性能瓶颈，由于使用CAS方法。高并发的情况下会造成大量的线程自旋，而只有一个线程会更新成功，浪费CPU资源。

LongAdder的思想是将单一的原子变量拆分为多个变量，从而降低高并发下的资源争抢。

高性能累加器



分治思想：空间换时间策略，将一个数相加，变为多个数相加；获取结果时将多个Cell结果相加。

Accumulator

使用Accumulator可以让你定义算法，不只是累加这么简单了！

java.util.concurrent.atomic.DoubleAccumulator

java.util.concurrent.atomic.LongAccumulator

谢谢观看！