java8 (http://39.106.3.150/tags/#java8)

Java8相关优化

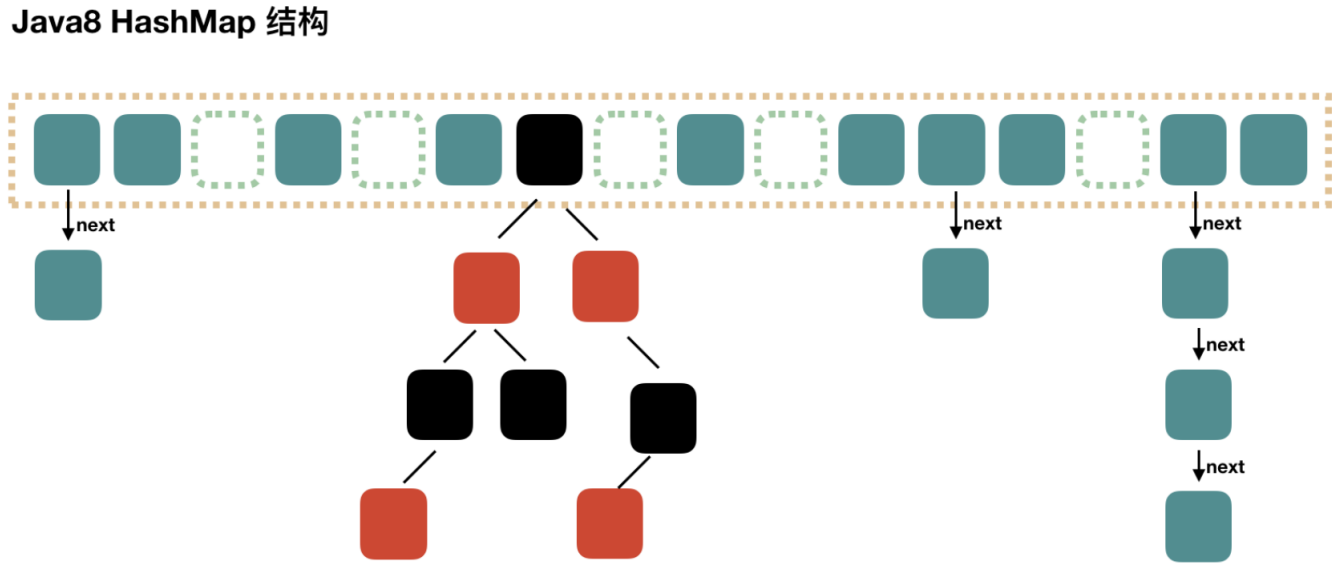
Posted by Palmer on 07-26，2019

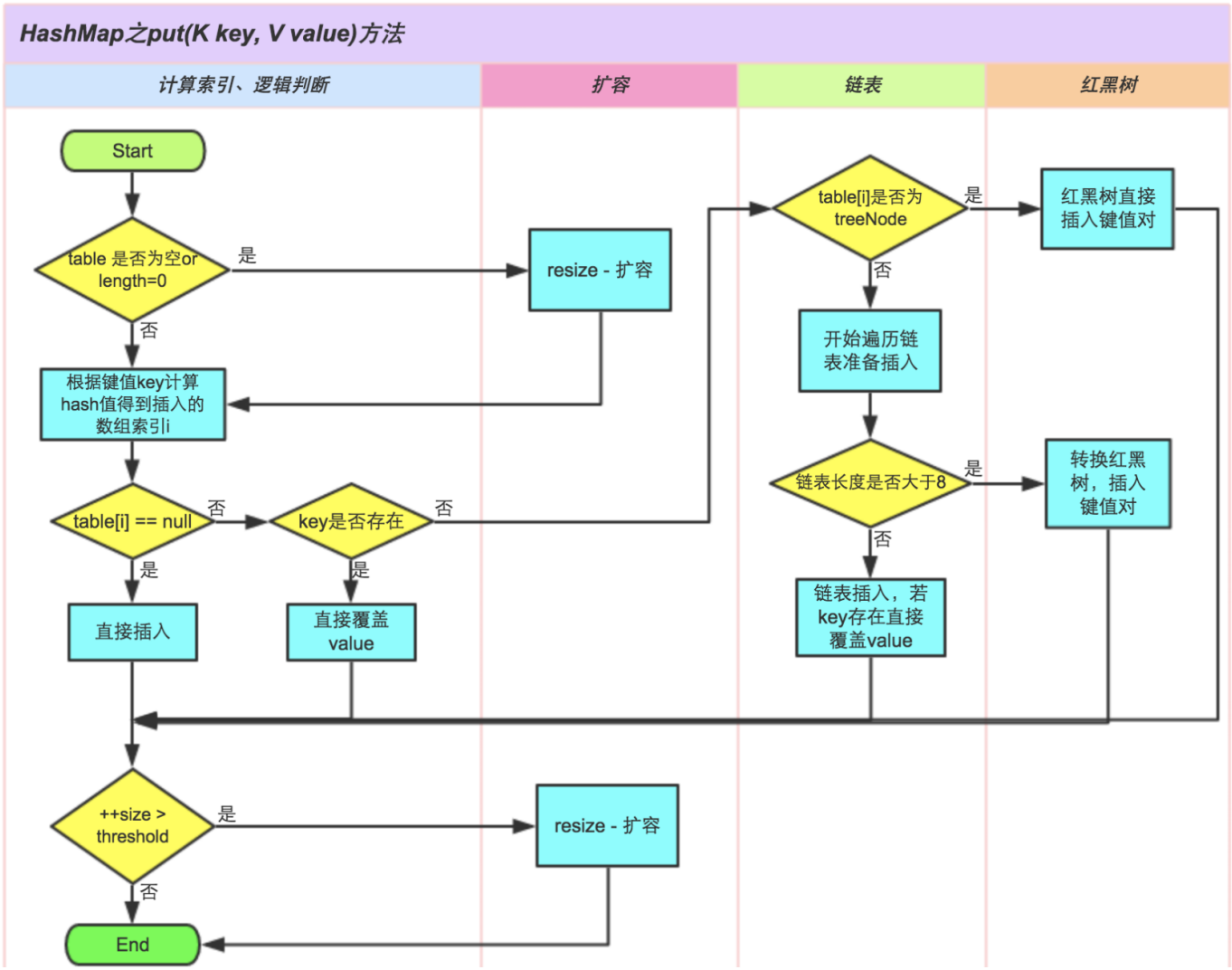
对HashMap的优化

在JDK1.7及以前的版本中，HashMap的数据结构是数组+链表，在往HashMap中存放数据对象的时候，会先根据哈希算法对该对象进行计算，得出该对象的哈希码，再根据哈希码计算该对象在HashMap数组中的位置，以确定该对象应该存放的具体位置。然而计算得来的哈希码虽然不会重复，但是计算出来的位置坐标是会重复的，因为HashMap数组的长度不是无限大的(默认为16)，这样就会产生碰撞(多个对象存放在HashMap数组的同一个坐标位置)。在产生碰撞后，会对同一坐标位置的对象依次进行equals比较，比较后如果内容相同则value值进行替换，否则则会形成一个链表，后进入的对象放在链表的末尾，很显然，链表越长越影响效率。

为了避免链表的长度过长，HashMap有自动扩容机制(默认的加载因子大小为0.75，即容量达到HashMap数组的75%时会自动扩容至原容量的2倍)，自动扩容后又会重新计算存放在HashMap中的对象的哈希码，以计算新的位置坐标，再将这些对象重新存放。因此应该尽量避免产生碰撞和扩容，我们可以在创建HashMap时估算存放数据的数量以指定容量，存放的数据数量最好为指定的容量的75%。

JDK8及其之后的HashMap的数据结构改为了数组+链表+红黑树，当存放在HashMap数组中的对象形成的链表长度超过8或者HashMap的总数据数超过64的时候，HashMap会把每个数组位置上的数据变成一个红黑树，这样做就大大的提升删改查的效率，但是增加数据的效率会降低。





对内存结构的优化

JDK8去掉了堆内存中的永久区，方法区从原来的永久区剥离出来成为元空间(MetaSpace)，并且元空间直接使用物理内存，而不必自己分配内存，一般而言物理内存的容量是足够大的，这样垃圾回收机制运行的概率就会大大降低，内存溢出的概率也会大大降低。在进行内存调优的时候，

原来的PermGenSize和MaxPermGenSize参数在JDK8中就没有用处了，取而代之的是MetaSpaceSize和MaxMetaSpaceSize，也就是说元空间的大小默认是物理内存的大小，但我们也可以指定元空间使用的内容空间大小。

对CAS对优化

由于采用这种 CAS 机制是没有对方法进行加锁的，所以，所有的线程都可以进入 increment() 这个方法，假如进入这个方法的线程太多，就会出现一个问题：每次有线程要执行第三个步骤的时候，i 的值老是被修改了，所以线程又到回到第一步继续重头再来。而这就会导致一个问题：由于线程太密集了，太多人想要修改 i 的值了，进而大部分人都会修改不成功，白白着在那里循环消耗资源。

为了解决这个问题，Java8 引入了一个 cell[] 数组，它的工作机制是这样的：假如有 5 个线程要对 i 进行自增操作，由于 5 个线程的话，不是很多，起冲突的几率较小，那就让他们按照以往正常的那样，采用 CAS 来自增吧。　　但是，如果有 100 个线程要对 i 进行自增操作的话，这个时候，冲突就会大大增加，系统就会把这些线程分配到不同的 cell 数组元素去，假如 cell[10] 有 10 个元素吧，且元素的初始化值为 0，那么系统就会把 100 个线程分成 10 组，每一组对 cell 数组其中的一个元素做自增操作，这样到最后，cell 数组 10 个元素的值都为 10，系统在把这 10 个元素的值进行汇总，进而得到 100，二这，就等价于 100 个线程对 i 进行了 100 次自增操作。

← PREVIOUS POST (HTTP://39.106.3.150/ARCHIVES/DOCKER-)

NEXT POST → (HTTP://39.106.3.150/ARCHIVES/JAVA-8)

开源框架 (http://39.106.3.150/tags/#1560569459781) Spring (http://39.106.3.150/tags/#spring)
设计模式 (http://39.106.3.150/tags/#1559888728999) linux (http://39.106.3.150/tags/#linux)
SpringBoot (http://39.106.3.150/tags/#springboot) 大数据 (http://39.106.3.150/tags/#1559363598973)
区块链 (http://39.106.3.150/tags/#1559363594390) Java (http://39.106.3.150/tags/#java)

FRIENDS

 (https://github.com/PowehiEdge)