不同类型垃圾回收器比较

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 作用域 | 算法 | 特性 |
| 串行收集器 | Serial GC作用于新生代 | 二者皆采用了串行回收与" Stop-the- | 一个成熟、经过长时间生产环境考验的极为高效的收集器。单CPU或较小应用内存等硬件平台不是特别优越的场合，它的性能表现可以超过并行回收器和并发回收器。 |
| Serial | Serial O Id GC作用于老年代垃 | World", Serial使用的是复制算法，而 |
|  | 圾收集 | Serial Old使用的是标记－压缩算法 |
| 并行收集器 |  | 并行回收和" Stop-the-World" , Parallel new使用的是复制算法，Parallel Old使用的是标记－压缩算法 | 程序吞吐量优先的应用场景中，在服务模 |
| Parallel new作用于新生代 | 式下内存回收的性能较为不错。 |
| Parallel | Parallel Old作用千老年代 | 并发收集的吞吐量是最好的，收集的时候 |
|  |
|  |  | 是准确收集，不会产生浮动垃圾 |
|  |  | 使用了标记清除算法，分为初始标记 | 并发低延迟，吞吐量较低。经过CMS收集 |
| 并发收集器 |  | (Initial-Mark, Stop-the-World)、并发标记 | 的堆会产生空间碎片，会带来堆内存的浪费 |
| Concurrent-Mark-Sweep | 老年代垃圾回收器 | (Concurrent-Mark)、再次标记(Remark, | CMS因为需要额外的空间存储代际的引用 |
|  |  | Stop-the-World)、并发清除(concurrent-Sweep) | 关系，所以有额外的消耗，同时CMS不是准 |
|  |  | 确收集，会产生浮动垃圾 |
|  | 按照分区进行收集，新生代的 |  | 基于并行和并发、低延迟以及暂停时间更 |
| 垃圾优先收集器 | 分区总是会回收，老生代则是并 | 使用了复制算法，同时辅以并发标记 | 加可控的区域化分代式服务器类型的垃圾回 |
| Gl,Garbage First | 发标记后选择部分回收效果最好 | 对整个堆空间标记 | 收器。因为分区设计，Gl引用关系的存储占 |
|  | 的分区 |  | 用的额外消耗很大，Gl也不是准确回收，也 |
|  |  |  | 会产生浮动垃圾 |
|  | 分区设计，不区分新生代和老 | 使用了复制算法，同时辅以并发标记 | 支持的内存更大，到TB级，暂停时间更 |
| Shenandoah | 生代，只按照分区收集 | 对整个堆空间标记，以并发回收对内存 | 短，也不是准确回收，占用的额外开销也最 |
|  |  | 做转移 | 大，在JDK 12中作为实验性质项目加入 |
|  | 可变大小分区（页）设计，不 | 使用并发标记和并发转移。和Shenan | 支持的内存更大，到TB级，暂停时间更 |
| ZGC | 区分新生代和老生代，只按照分 | 短，也不是准确回收，在JDK 11中作为实验 |
|  | doah类似，但是在实现上有所不同 |
|  | 区收集 | 性质项目加入 |