1、 执行语句是：java -Xmx1G -XX:+UseParallelGC GCLogAnalysis

结论：并行 GC 最优而配置是 8G，最大吞吐量是 18158

2、 分析并行 GC 的 GCLog:

语句：java -Xmx512M -Xms512M -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:+UseParallelGC GCLogAnalysis

结论：发生 YoungGC 次数大概是 32 次，时间大概是 0.0089569 secs 左右。FullGC 次数大概是 12 次，时间 0.04 secs

特点：

适用场景：

3、 串行 GC：

语句： java -Xmx512M -Xms512M -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:+UseSerialGC GCLogAnalysis

结论：串行 GC, 发生 YoungGC 次数 18 次，每次大概时间 0.0223625 secs，FULL GC 次数 2 次，0.04 secs。YoungGC 的时间远远大于并行 GC。

4、 串行 GC

结论：串行最优配置是 1G，最大吞吐量是 15340

　　　堆太小，OOM

　　　堆太大，不一定是最优

　　相同的配置情况下，并行 GC 虽然发生的 YoungGC 次数比串行的多，但是整体暂停时间比串行 GC 少很多。

5、 CMS GC

命令：java -Xmx512M -Xms512M -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX: +UseConcMarkSweepGC GCLogAnalysis

YoungGC，大概每次时间是 0.0169421 secs，暂停时间长，比串行 GC 的短一点，比并行 GC 的长。

6、 CMS 比对：

分析结论：

最优配置是 1G, 最大吞吐量是 16648

堆太小，会导致应用出现 OOM。

堆太大，可能导致 YGC、FULLGC，消耗很长时间用来做 GC, 导致吞吐量小。

CMS GC 的 YoungGC 比串行 GC 的好一点，比并行 GC 差。

7、 G1GC

命令：java -Xmx16G -Xms16G -XX:+PrintGCDateStamps -XX:+UseG1GC GCLogAnalysis

```
root@minhu-System-Product-Name:/home/qingteng/桌面# java -Xmx16G -Xms16G -XX:+PrintGCDateStamps -XX:+UseG1GC GCLogAnalysis
Java HotSpot(TM) 64-Bit Server VM warning: INFO: os::commit_memory(0x00000003c0000000, 17179869184, 0) failed; error='Cannot allocate memory' (errno=12)
#
# There is insufficient memory for the Java Runtime Environment to continue.
# Native memory allocation (mmap) failed to map 17179869184 bytes for committing reserved memory.
# An error report file with more information is saved as:
# /home/qingteng/桌面/hs_err_pid27055.log
root@minhu-System-Product-Name:/home/qingteng/桌面# java -Xmx8G -Xms8G -XX:+PrintGCDateStamps -XX:+UseG1GC GCLogAnalysis
正在执行...
执行结束!共生成对象次数:18422
root@minhu-System-Product-Name:/home/qingteng/桌面# java -Xmx4G -Xms4G -XX:+PrintGCDateStamps -XX:+UseG1GC GCLogAnalysis
正在执行...
执行结束!共生成对象次数:16958
root@minhu-System-Product-Name:/home/qingteng/桌面# java -Xmx2G -Xms2G -XX:+PrintGCDateStamps -XX:+UseG1GC GCLogAnalysis
正在执行...
执行结束!共生成对象次数:17953
root@minhu-System-Product-Name:/home/qingteng/桌面# java -Xmx2G -Xms2G -XX:+PrintGCDateStamps -XX:+UseG1GC GCLogAnalysis
正在执行...
执行结束!共生成对象次数:16208
root@minhu-System-Product-Name:/home/qingteng/桌面# java -Xmx1G -Xms1G -XX:+PrintGCDateStamps -XX:+UseG1GC GCLogAnalysis
正在执行...
执行结束!共生成对象次数:13523
root@minhu-System-Product-Name:/home/qingteng/桌面# java -Xmx512M -Xms512M -XX:+PrintGCDateStamps -XX:+UseG1GC GCLogAnalysis
正在执行...
执行结束!共生成对象次数:10940
root@minhu-System-Product-Name:/home/qingteng/桌面# java -Xmx256M -Xms256M -XX:+PrintGCDateStamps -XX:+UseG1GC GCLogAnalysis
正在执行...
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
        at java.util.Arrays.copyOfRange(Arrays.java:3664)
        at java.lang.String.<init>(String.java:207)
        at java.lang.StringBuilder.toString(StringBuilder.java:407)
        at GCLogAnalysis.generateGarbage(GCLogAnalysis.java:58)
        at GCLogAnalysis.main(GCLogAnalysis.java:25)
```

最优的配置是 8G，最大吞吐量是 18422.

8、 G1GC 详细分析：

```
2021-01-22T20:24:59.492+0800: [GC concurrent-root-region-scan-start]
2021-01-22T20:24:59.492+0800: [GC concurrent-root-region-scan-end, 0.0000174 secs]
2021-01-22T20:24:59.492+0800: [GC concurrent-mark-start]
2021-01-22T20:24:59.492+0800: [GC pause (G1 Evacuation Pause) (young) (to-space exhausted), 0.0015166 secs]
   [Parallel Time: 0.8 ms, GC Workers: 4]
      [GC Worker Start (ms): Min: 1063.3, Avg: 1063.5, Max: 1064.1, Diff: 0.7]
      [Ext Root Scanning (ms): Min: 0.0, Avg: 0.1, Max: 0.1, Diff: 0.1, Sum: 0.4]
      [Update RS (ms): Min: 0.0, Avg: 0.0, Max: 0.1, Diff: 0.1, Sum: 0.2]
         [Processed Buffers: Min: 0, Avg: 1.2, Max: 3, Diff: 3, Sum: 5]
      [Scan RS (ms): Min: 0.0, Avg: 0.0, Max: 0.0, Diff: 0.0, Sum: 0.0]
      [Code Root Scanning (ms): Min: 0.0, Avg: 0.0, Max: 0.0, Diff: 0.0, Sum: 0.0]
      [Object Copy (ms): Min: 0.0, Avg: 0.0, Max: 0.0, Diff: 0.0, Sum: 0.0]
      [Termination (ms): Min: 0.0, Avg: 0.4, Max: 0.6, Diff: 0.6, Sum: 1.8]
         [Termination Attempts: Min: 1, Avg: 1.2, Max: 2, Diff: 1, Sum: 5]
      [GC Worker Other (ms): Min: 0.0, Avg: 0.0, Max: 0.0, Diff: 0.0, Sum: 0.0]
      [GC Worker Total (ms): Min: 0.0, Avg: 0.6, Max: 0.8, Diff: 0.8, Sum: 2.3]
      [GC Worker End (ms): Min: 1064.1, Avg: 1064.1, Max: 1064.1, Diff: 0.0]
   [Code Root Fixup: 0.0 ms]
   [Code Root Purge: 0.0 ms]
   [Clear CT: 0.0 ms]
   [Other: 0.7 ms]
      [Evacuation Failure: 0.0 ms]
      [Choose CSet: 0.0 ms]
      [Ref Proc: 0.6 ms]
      [Ref Enq: 0.0 ms]
      [Redirty Cards: 0.0 ms]
      [Humongous Register: 0.0 ms]
      [Humongous Reclaim: 0.0 ms]
      [Free CSet: 0.0 ms]
   [Eden: 1024.0K(25.0M)->0.0B(25.0M) Survivors: 0.0B->0.0B Heap: 423.7M(512.0M)->423.7M(512.0M)]
 [Times: user=0.00 sys=0.00, real=0.00 secs]
2021-01-22T20:24:59.494+0800: [Full GC (Allocation Failure)  423M->348M(512M), 0.0343124 secs]
   [Eden: 0.0B(25.0M)->0.0B(25.0M) Survivors: 0.0B->0.0B Heap: 423.7M(512.0M)->348.3M(512.0M)], [Metaspace: 2536K->2536K(1056768K)]
 [Times: user=0.04 sys=0.00, real=0.03 secs]
执行结束!共生成对象次数:113072021-01-22T20:24:59.528+0800: [GC concurrent-mark-abort]
```

结论：G1 GC 暂停的时间 0.8ms 很少，大部分时间在工作。最优堆配置是 8G.最大吞吐量为：18422

jdk1.8 默认是并行 GC