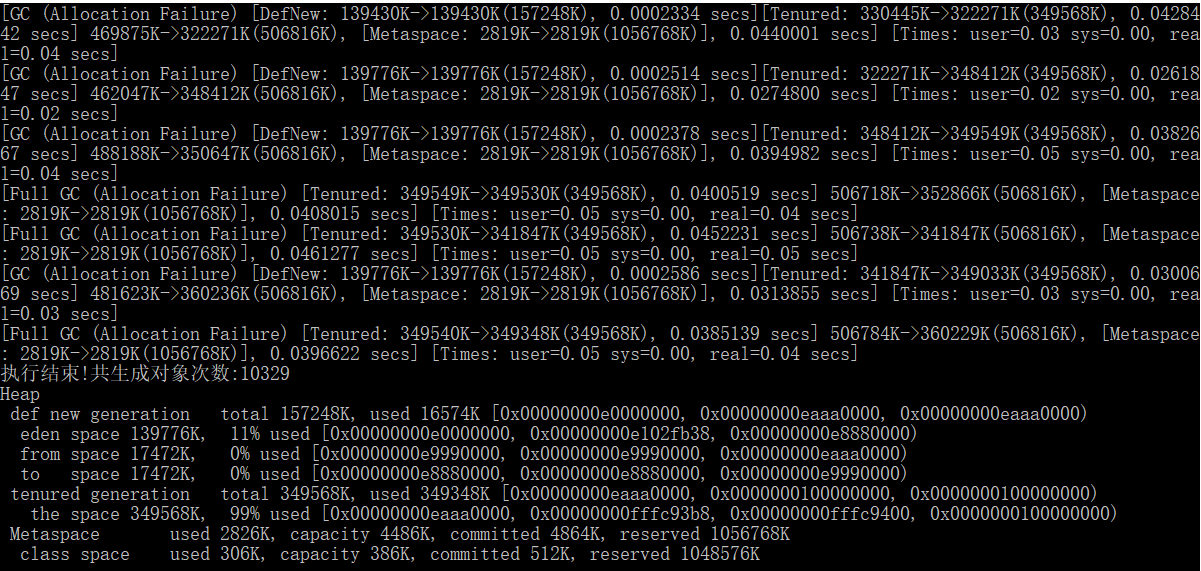
作业演示

一、使用 GCLogAnalysis.java 自己演练一遍串行/并行/CMS/G1的案例。

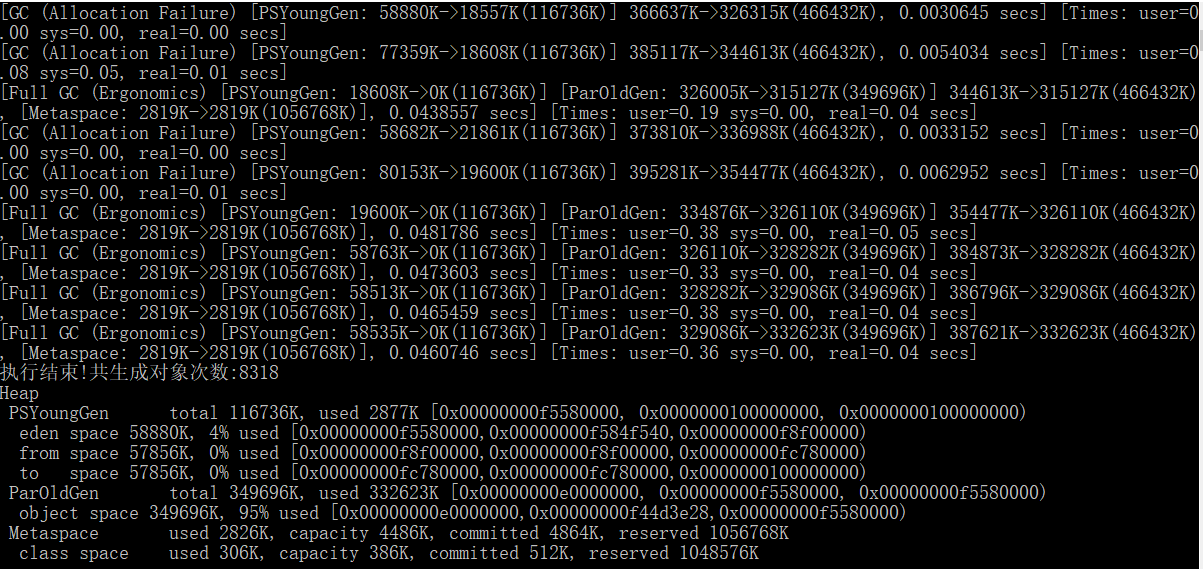
1、串行GC

java -XX:+UseSerialGC -Xms512m -Xmx512m -XX:+PrintGCDetails GCLogAnalysis



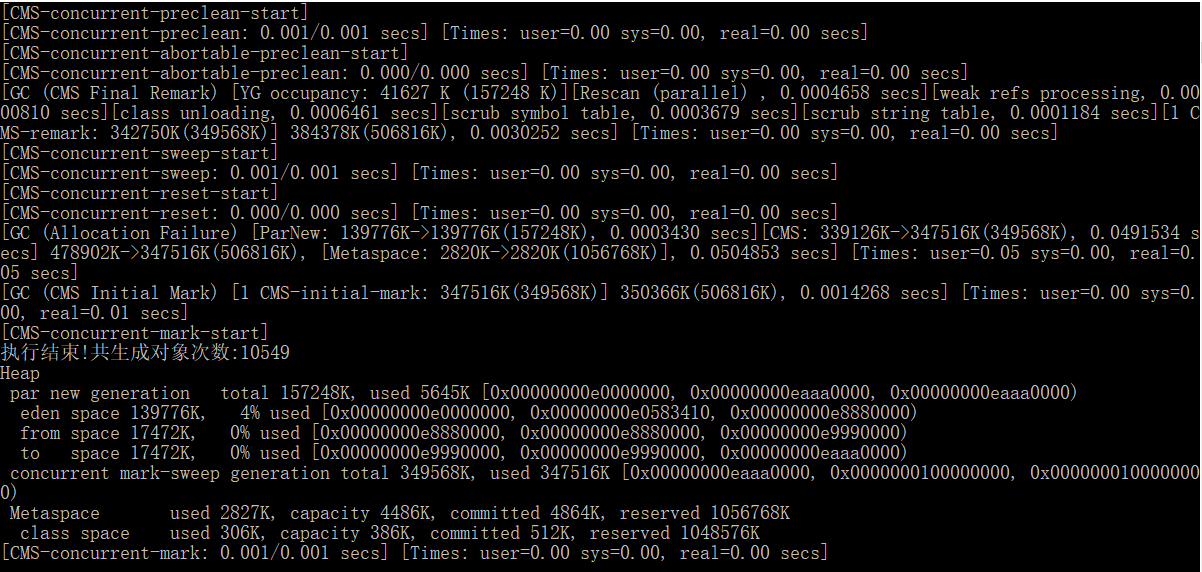
2、并行GC

java -XX:+UseParallelGC -Xms512m -Xmx512m -XX:+PrintGCDetails GCLogAnalysis



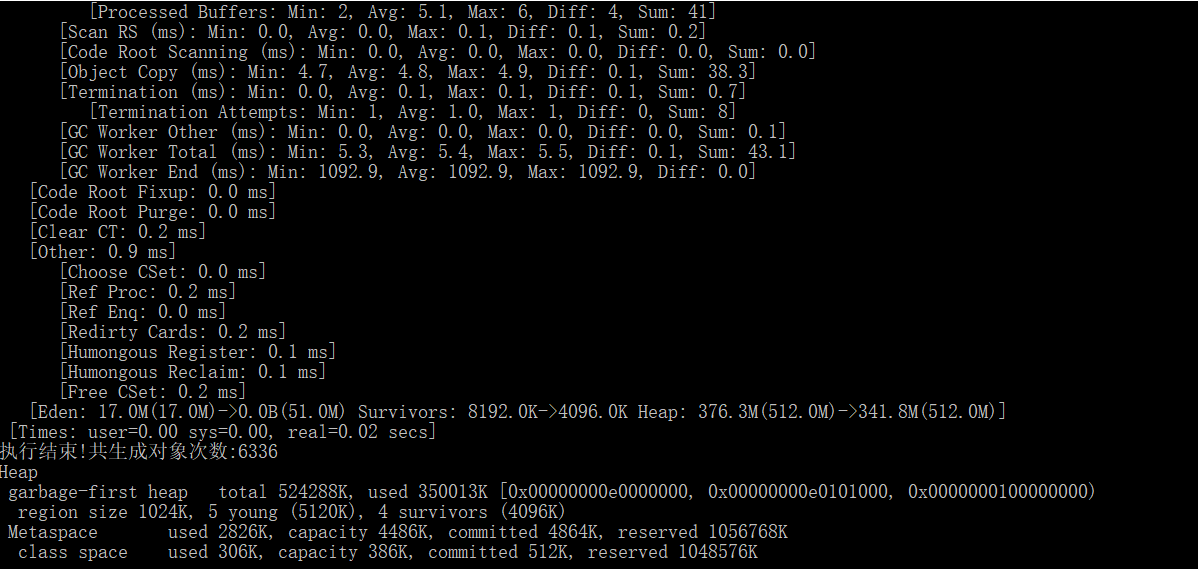
3、CMS GC

java -XX:+UseConcMarkSweepGC -Xms512m -Xmx512m -XX:+PrintGCDetails GCLogAnalysis



4、G1 GC

java -XX:+UseG1GC -Xms512m -Xmx512m -XX:+PrintGCDetails GCLogAnalysis

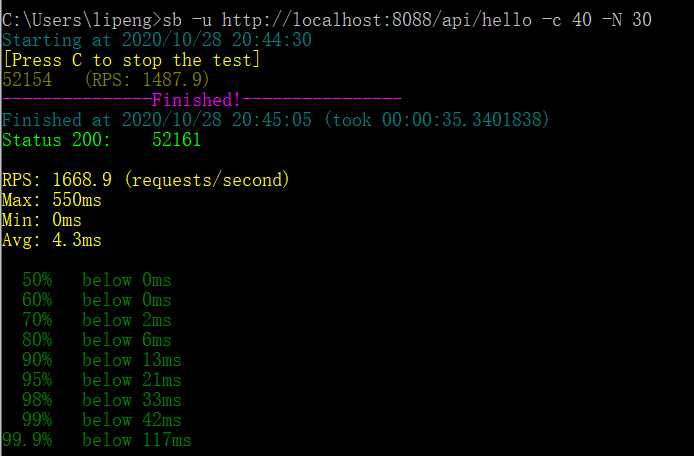


二、使用压测工具(wrk或sb)，演练gateway-server-0.0.1-SNAPSHOT.jar 示例

1、并行GC

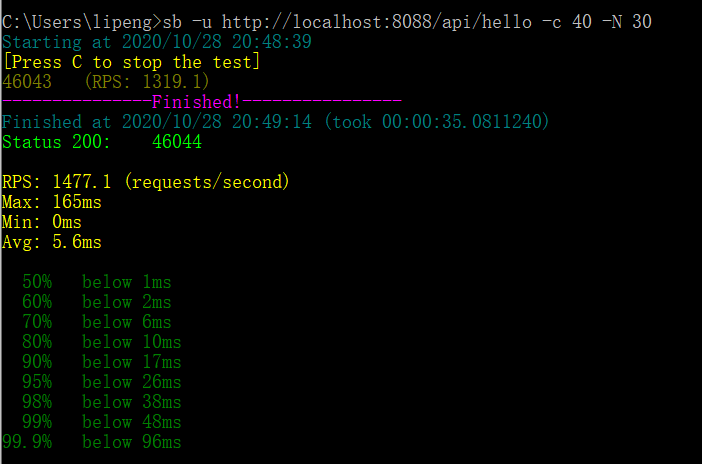
java -jar -XX:+UseParallelGC -Xms512m -Xmx512m gateway-server-0.0.1-SNAPSHOT.jar

sb -u http://localhost:8088/api/hello -c 40 -N 30



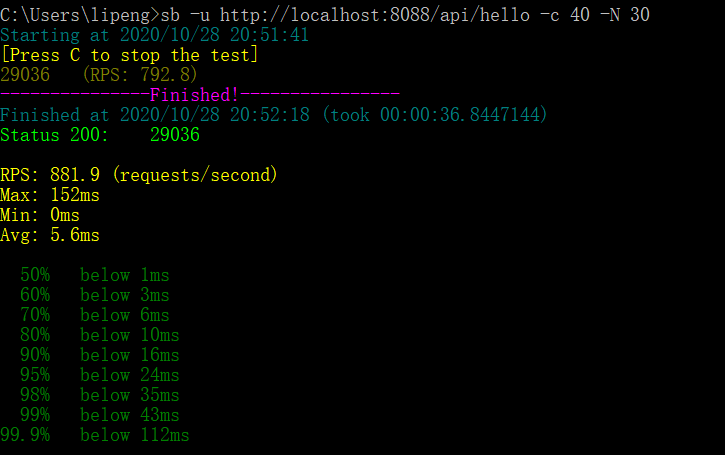
2、CMS GC

java -jar -XX:+UseConcMarkSweepGC -Xms512m -Xmx512m gateway-server-0.0.1-SNAPSHOT.jar



3、G1 GC

java -jar -XX:+UseG1GC -Xms4g -Xmx4g gateway-server-0.0.1-SNAPSHOT.jar



三、GC总结：

1、串行

(1) 年轻代使用拷贝-复制，老年代使用标记-清除-整理。

(2) 发生GC时，会暂停

(3) 内存越小，GC次数越多

2、并行GC

(1) 年轻代使用拷贝-复制，老年代使用标记-清除-整理。

(2) GC处理时，暂停业务处理，所有线程处理GC垃圾回收。平常运行时，所以线程都去处理业务。因此，吞吐量比较高。

(3) 内存越小，GC次数越多

3、CMS GC

(1) 年轻代使用拷贝-复制，老年代使用标记-清除

(2) CMS默认GC线程数是1/4，并且老年代只清除，无整理。所以当GC发生时，吞吐量不如并行 GC

(3) CMS GC 6个阶段

A、初始化标记 - 暂停GC

B、并行标记

C、并行预清理

D、最终标记 - 暂停GC

E、并行清理

F、并行重置

G、因为无整理，并且CMS GC 6阶段 暂停时间短，所以延迟比较低

(4) 内存越小，GC次数越多

1. G1 GC

(1) 不分代,使用 region(2048) 存储数据，分为：

A、Eden区 （标记-复制 算法）

B、存活区

C、老年区 (标记-复制-整理 算法)

(2) GC 3个阶段

A、(G1 Evacuation Pause) (young)

B、类似 CMS GC 的 并发标记

initial-mark

concurrent-root-region-scan

concurrent-mark

remark

concurrent-cleanup

C、(G1 Evacuation Pause) (mix)

(3) 使用 -XX:MaxGCPauseMills 参数可以控制 GC暂停时间