# 1.压测准备

## 1.安装SB工具

执行此目录下的env.txt命令。

注意事项：

1. 需要用管理员权限执行安装命令
2. FullyQualifiedErrorId : Unable to unzip package using 7zip. Perhaps try setting $env:chocolateyUseWindowsCompression = 'true' and call install again. Error: 7-Zip signalled an unknown error (code )

解压失败的错误，单独执行下面命令即可：

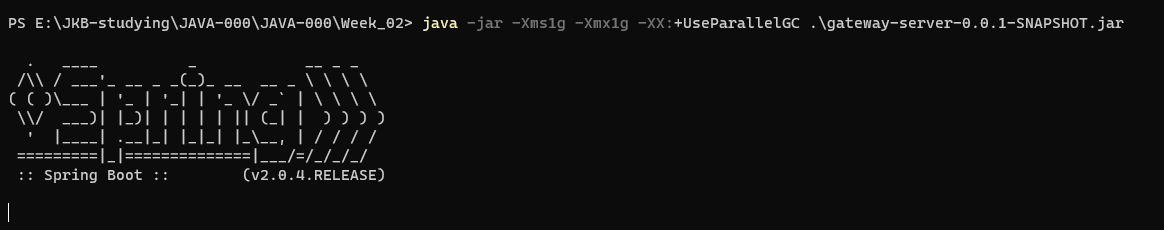
$env:chocolateyUseWindowsCompression = 'true'

## 2.启动jar

老师提供的jar为SprinBoot项目，内置tomcat容器，可以直接启动。启动参数如下

java -jar [启动参数，比如设置GC heap等信息] xxx.jar

java -jar -Xms1g -Xmx1g -XX:+UseParallelGC .\gateway-server-0.0.1-SNAPSHOT.jar



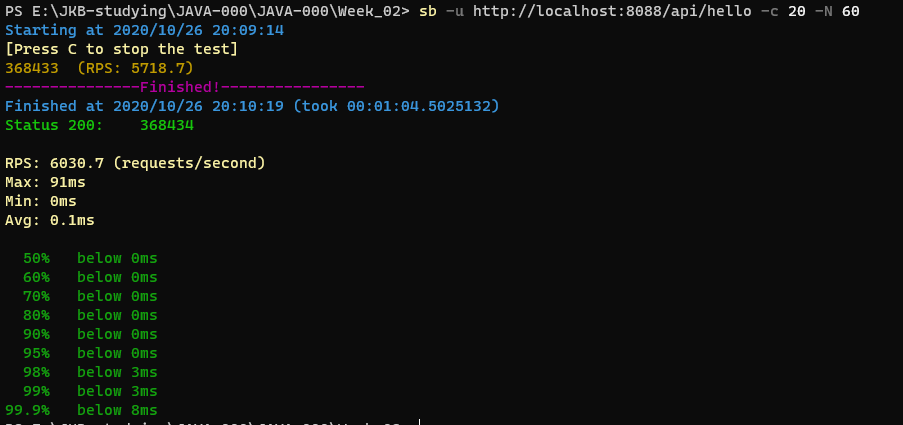
结束任务：Ctrl + c

## 3.启动sb

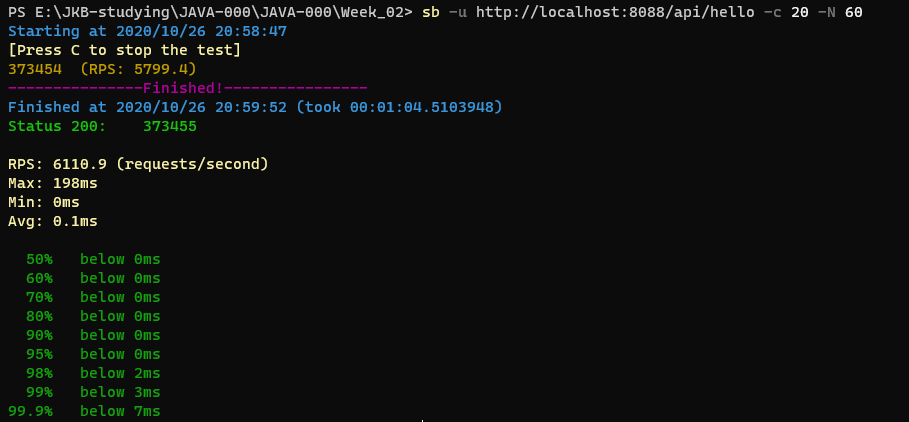
启动sb之后会自动弹出sb的监控界面，如下：

sb -u http://localhost:8088/api/hello -c 20 -N 60 表示每秒的并发数=20个，压测时间=60s，所以压测工具会执行60s的请求

串行GC：



并行GC：



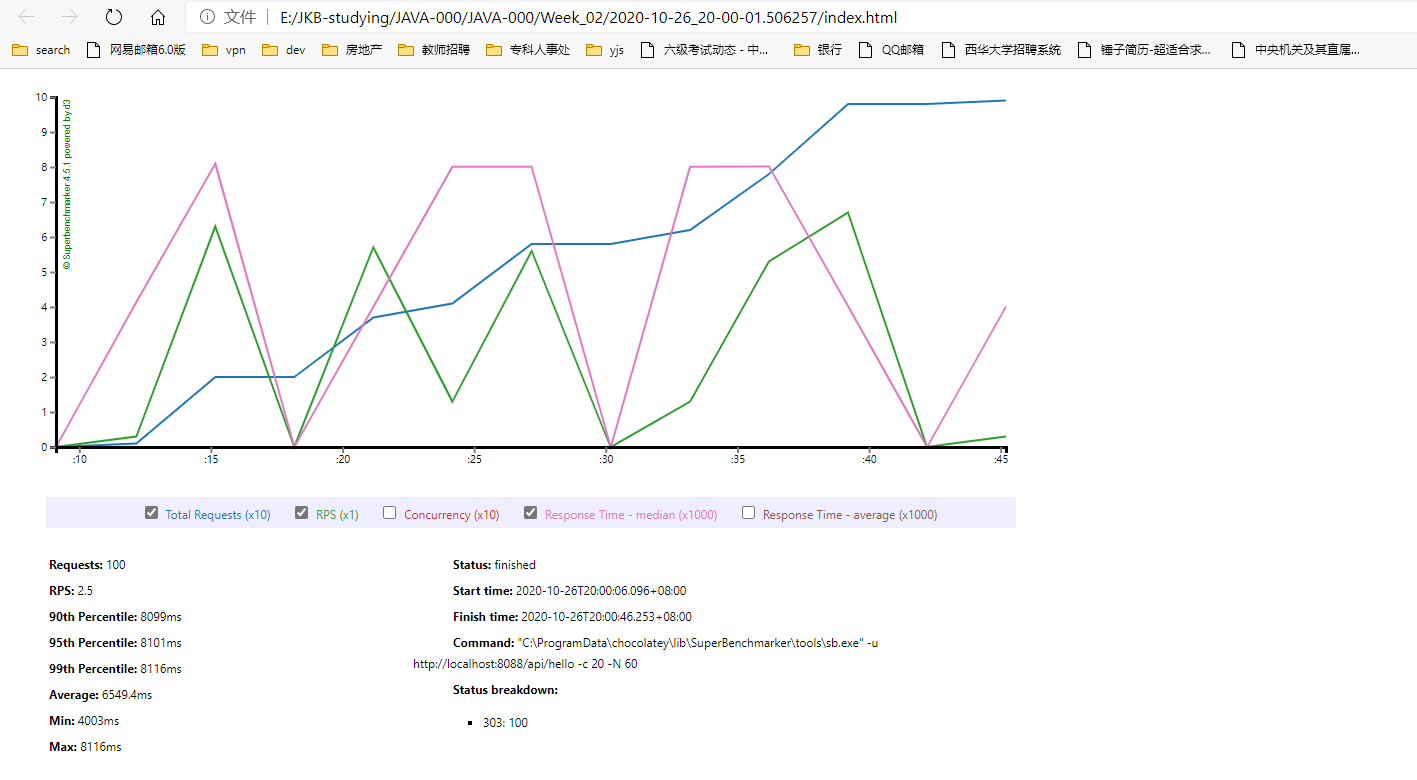
RPS:平均吞吐量

Max：最高延迟

Min：最低延迟

Avg：平均延迟

总结：RPS数差不多，但是并行GC的最大延迟更高，符合并行GC的特点。



根据提示得知，退出sb的命令为C

superbenchMarker比较重要的参数如下：

-u 可用来指定要压测的网站或是API

-n 可用来指定要压的Request数量

-N 表示执行的时间

-c 可用来指定Concurrent Request数量

-m 可用来指定要使用的HTTP Method

-h 可用来指定要显示HTTP Header

-q 可用来指定要显示Cookie

所以我们要发送1000个Request去测试腾讯网的话 可以输入命令

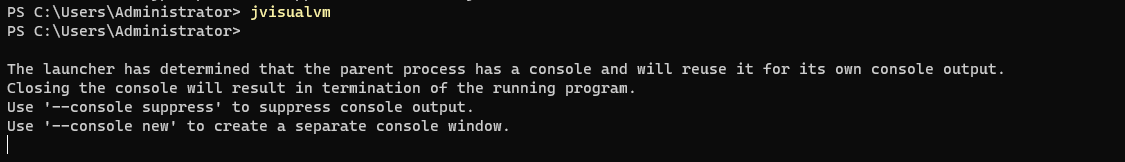
sb -u https://www.qq.com -n 1000

要发送1000个Request, 且同时间可能有10个Concurrent Request去测试腾讯网的话 可以输入命令

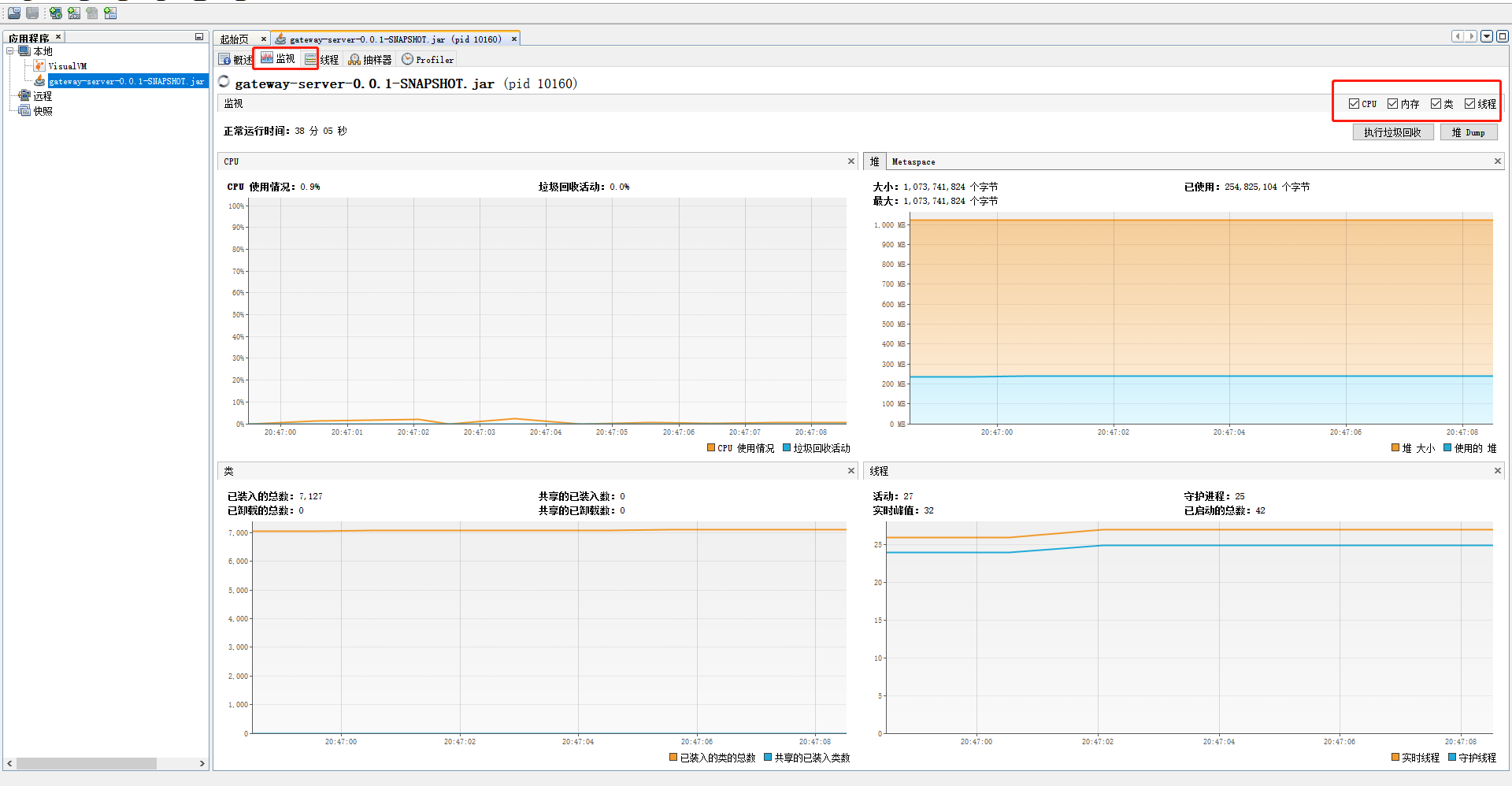
sb -u https://www.qq.com -n 1000 -c 10

## 4.启动jvisualvm

启动方式，如果只想让jvisualvm挂本地启动的java进程，则可以执行命令jvisualvm，会自动寻找jdk中的此工具并启动。也可以进入jdk，手动启动。

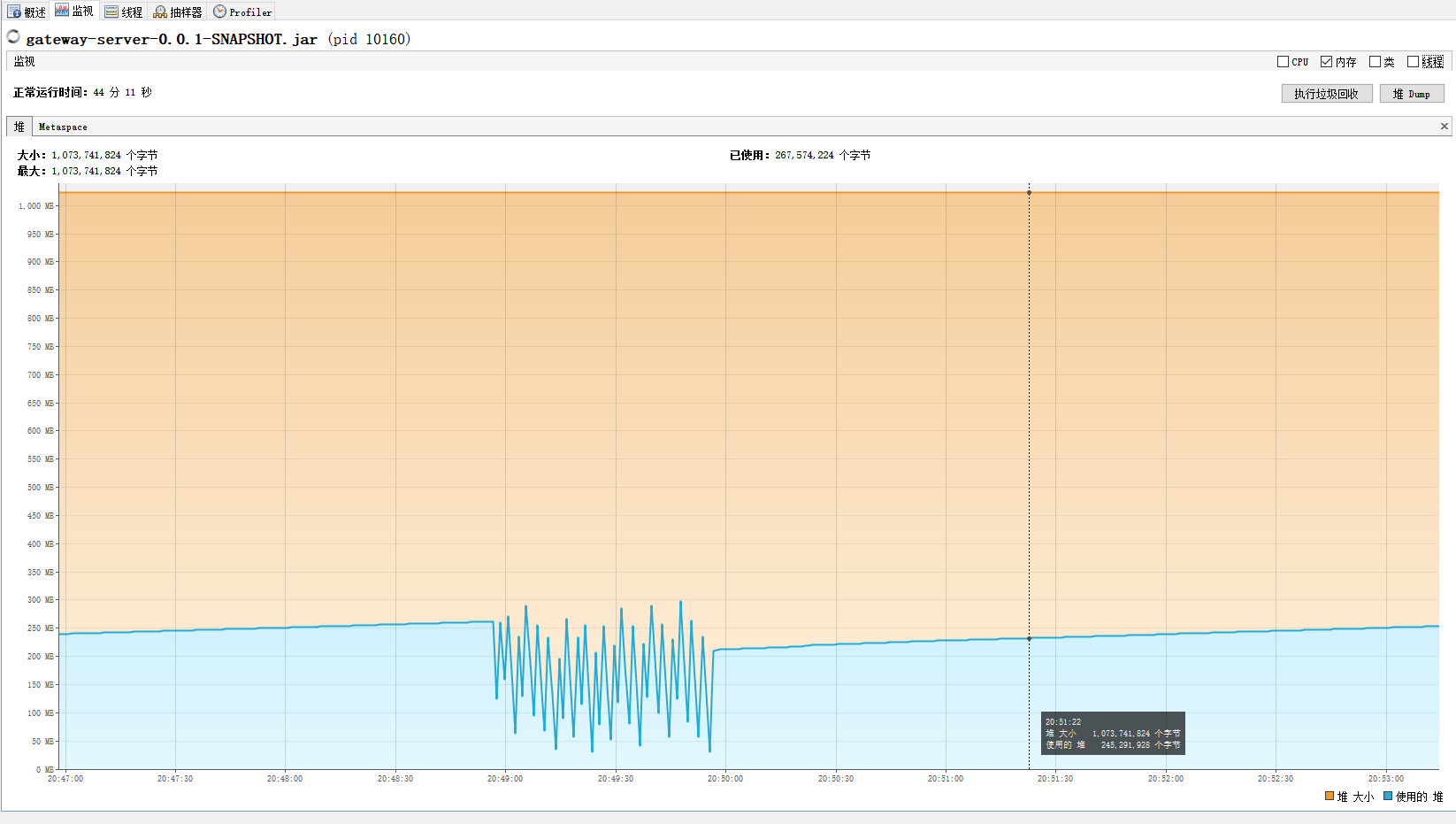


双击需要监视的进程，进入监控界面。如下：

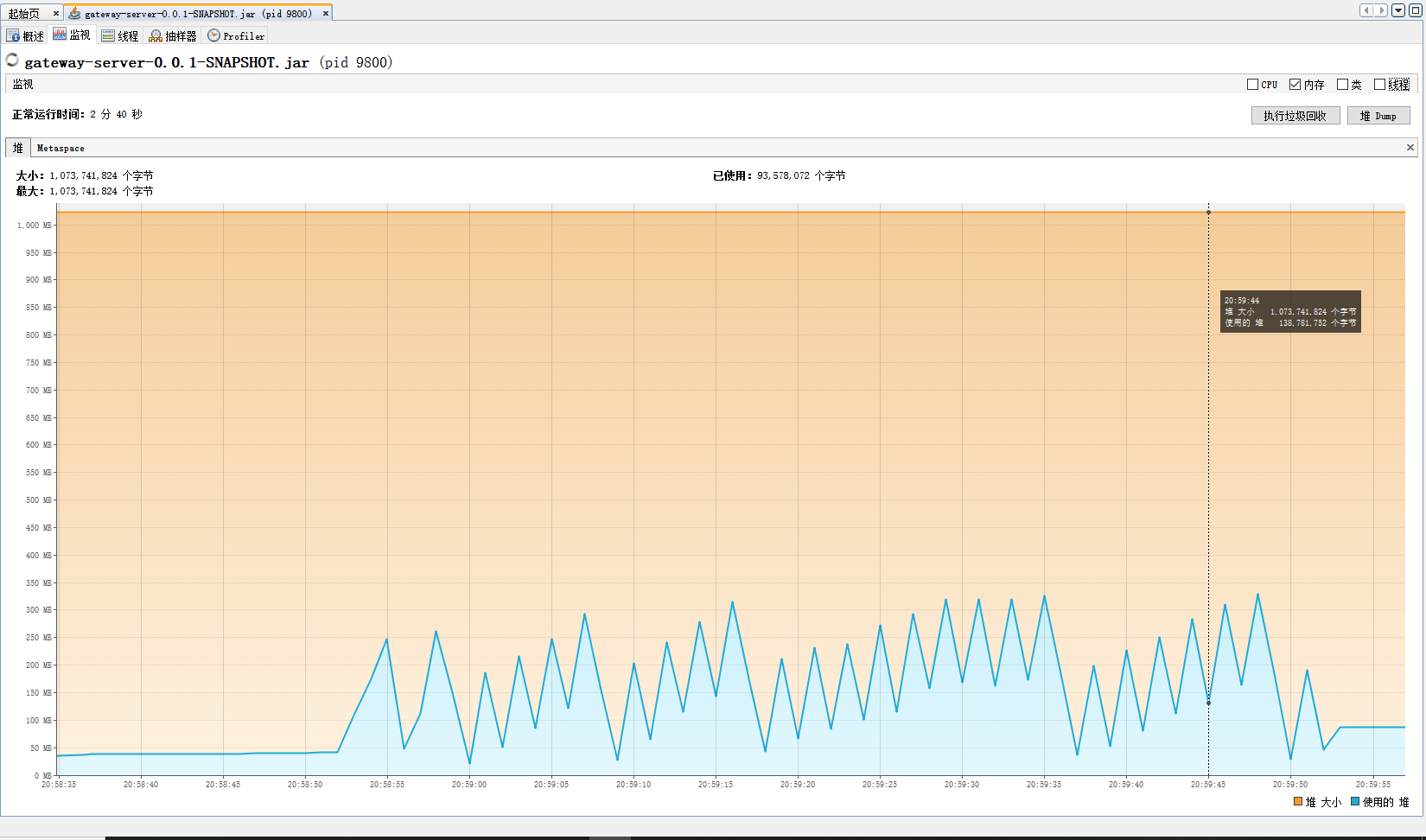


再次压测，结果如下：

串行GC：



并行GC：



内存波形，表示发生了GC。

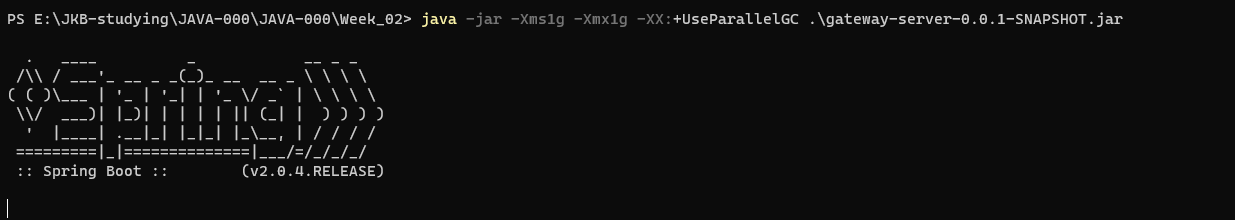
总结：串行GC由于每次GC的力度不够，导致GC次数频繁。

# 2.压测并行GC

## 1.Xmx=Xms=1g -c=20

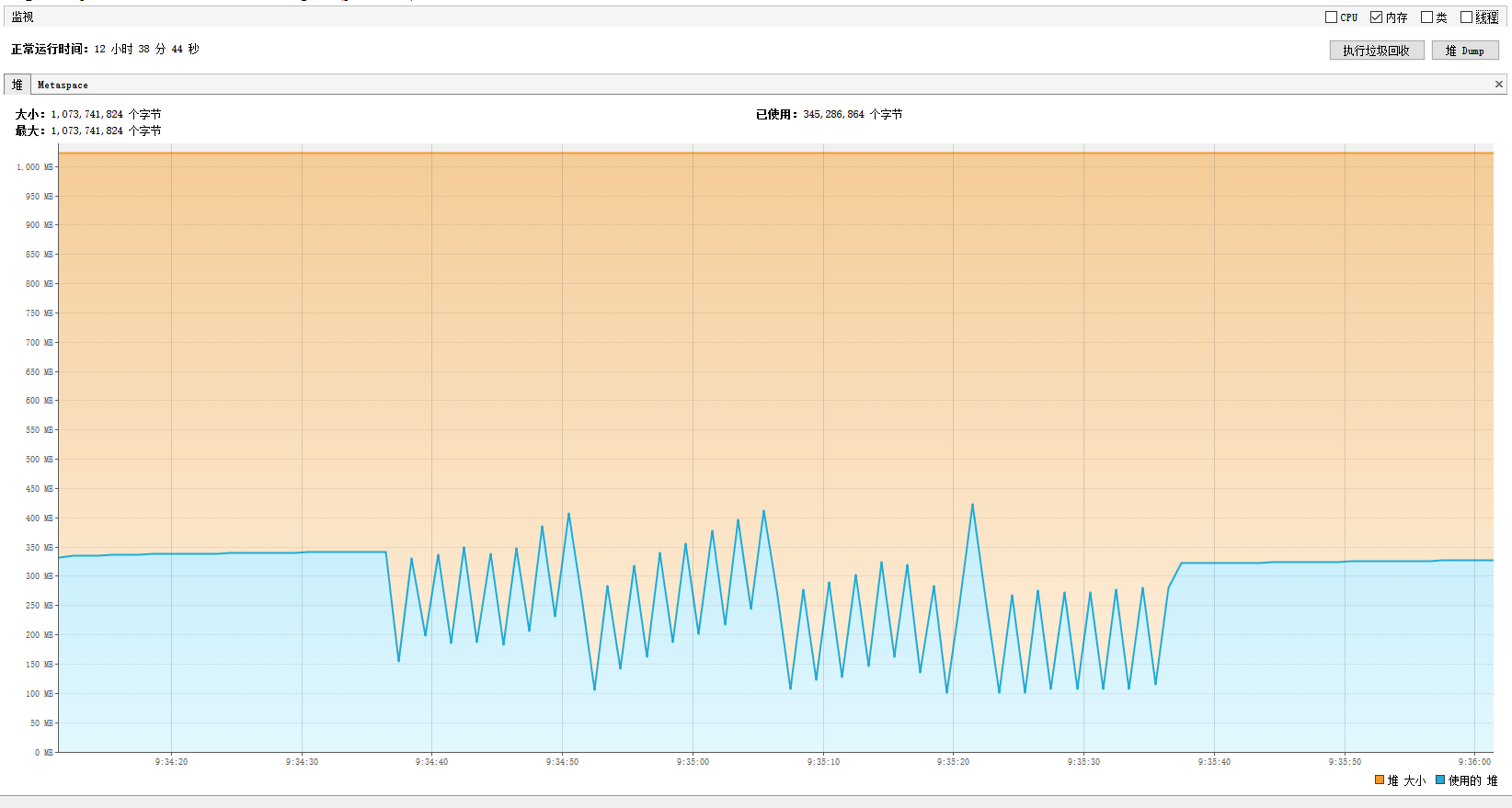
### 启动app并执行压测

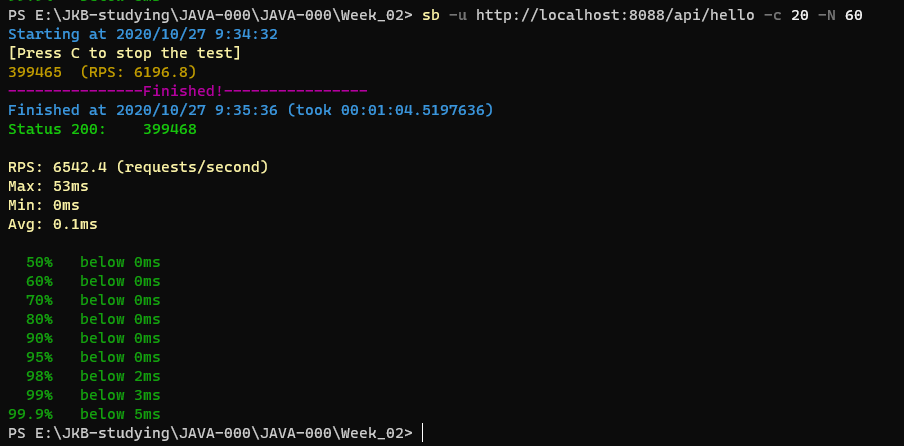
java -jar -Xms1g -Xmx1g -XX:+UseParallelGC .\gateway-server-0.0.1-SNAPSHOT.jar



sb -u http://localhost:8088/api/hello -c 20 -N 60

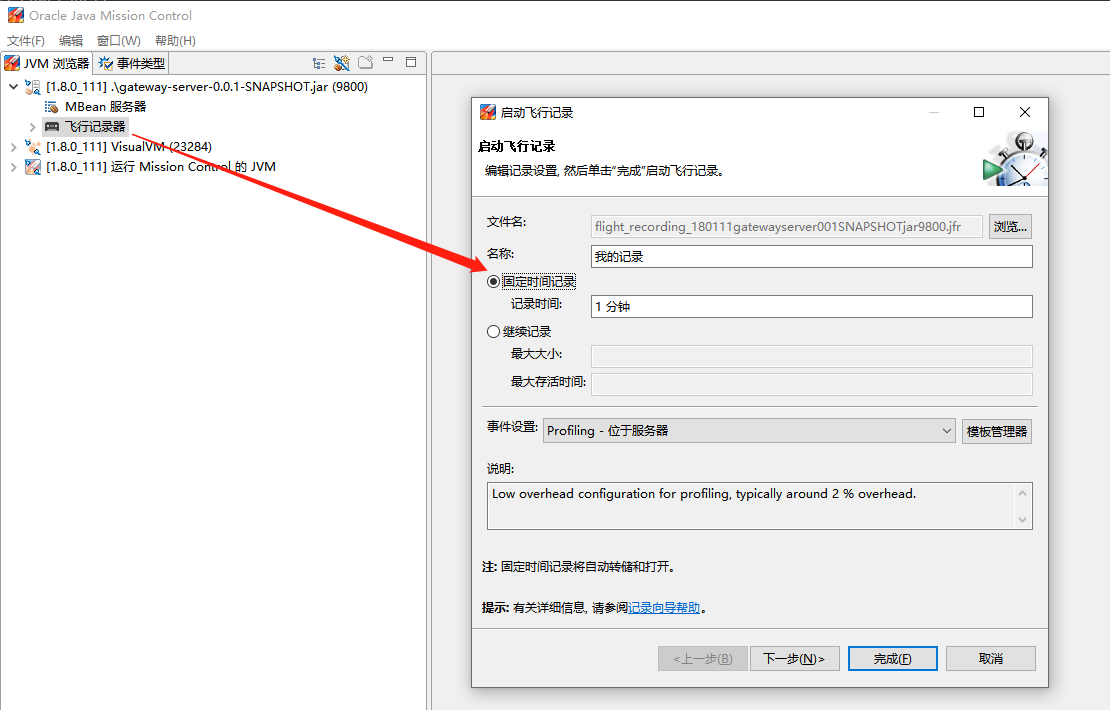
### 挂jvisualvm查看





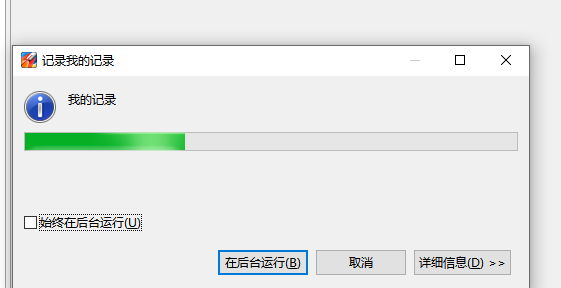
### 挂JMC查看

执行jmc，命令启动jmc



设置为记录一分钟的数据。

注意，在下面这个时候就需要执行压测，否则无数据：



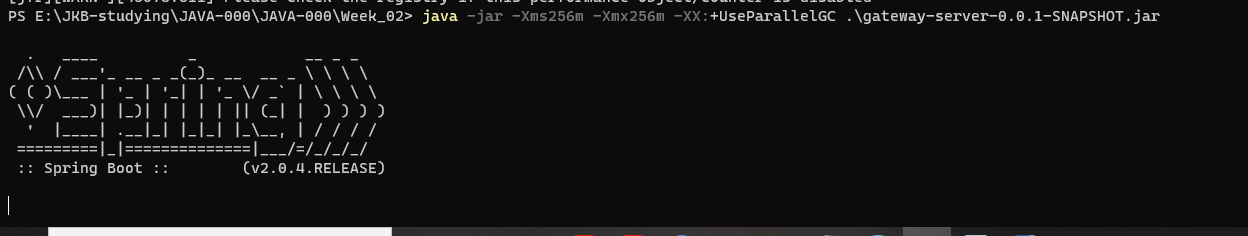
飞行记录如下：  


这里可以看到GC暂停时间平均值为3ms，最大值为5ms。

## 2.Xmx=Xms=256m -c=20

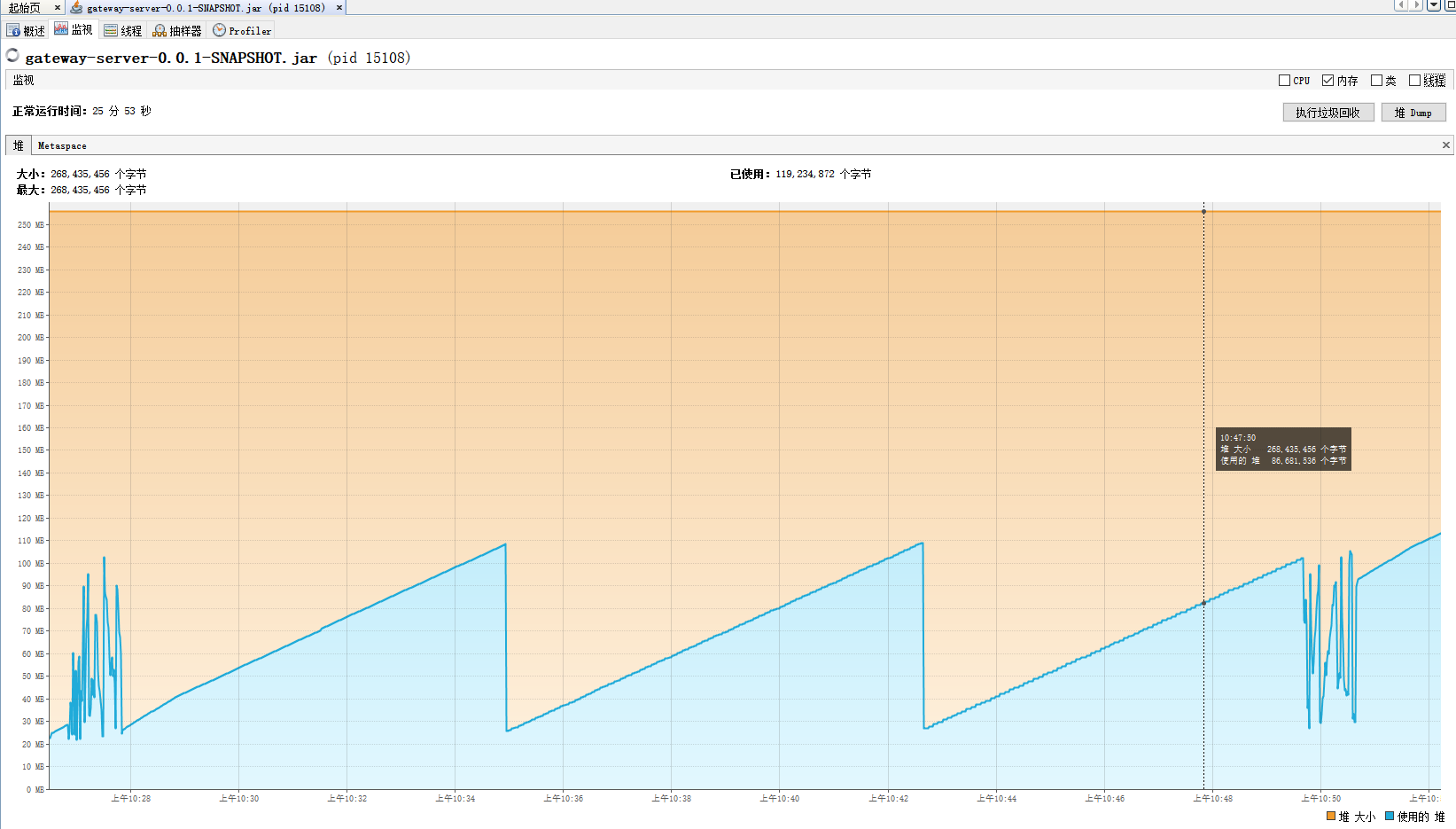
### 启动app并执行压测

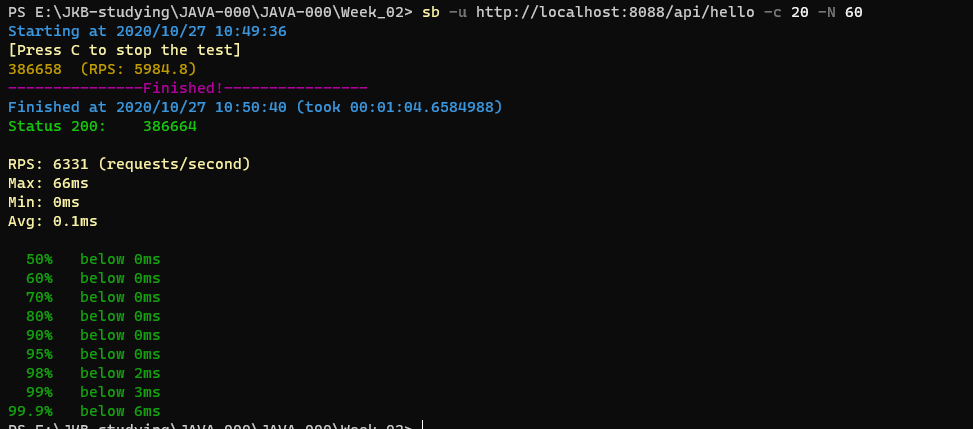
java -jar -Xms256m -Xmx256m -XX:+UseParallelGC .\gateway-server-0.0.1-SNAPSHOT.jar



sb -u http://localhost:8088/api/hello -c 20 -N 60

### 挂jvisualvm查看





总结：相比1g，256的最大延迟更高

### 挂JMC查看

执行jmc，命令启动jmc

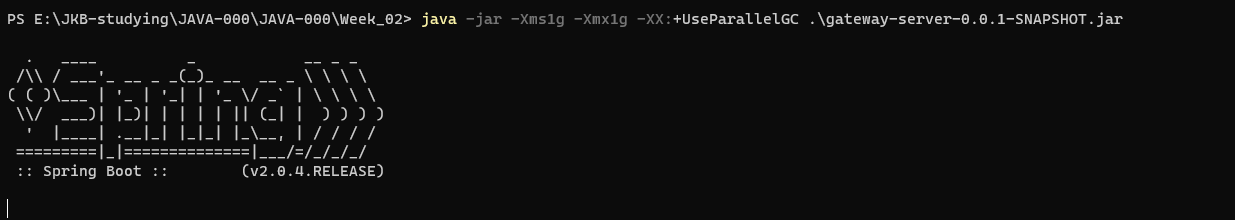
飞行记录如下：  


这里可以看到GC暂停时间平均值为2ms，最大值为3ms相比1g更短。

## 3.Xmx=Xms=1g -c=100

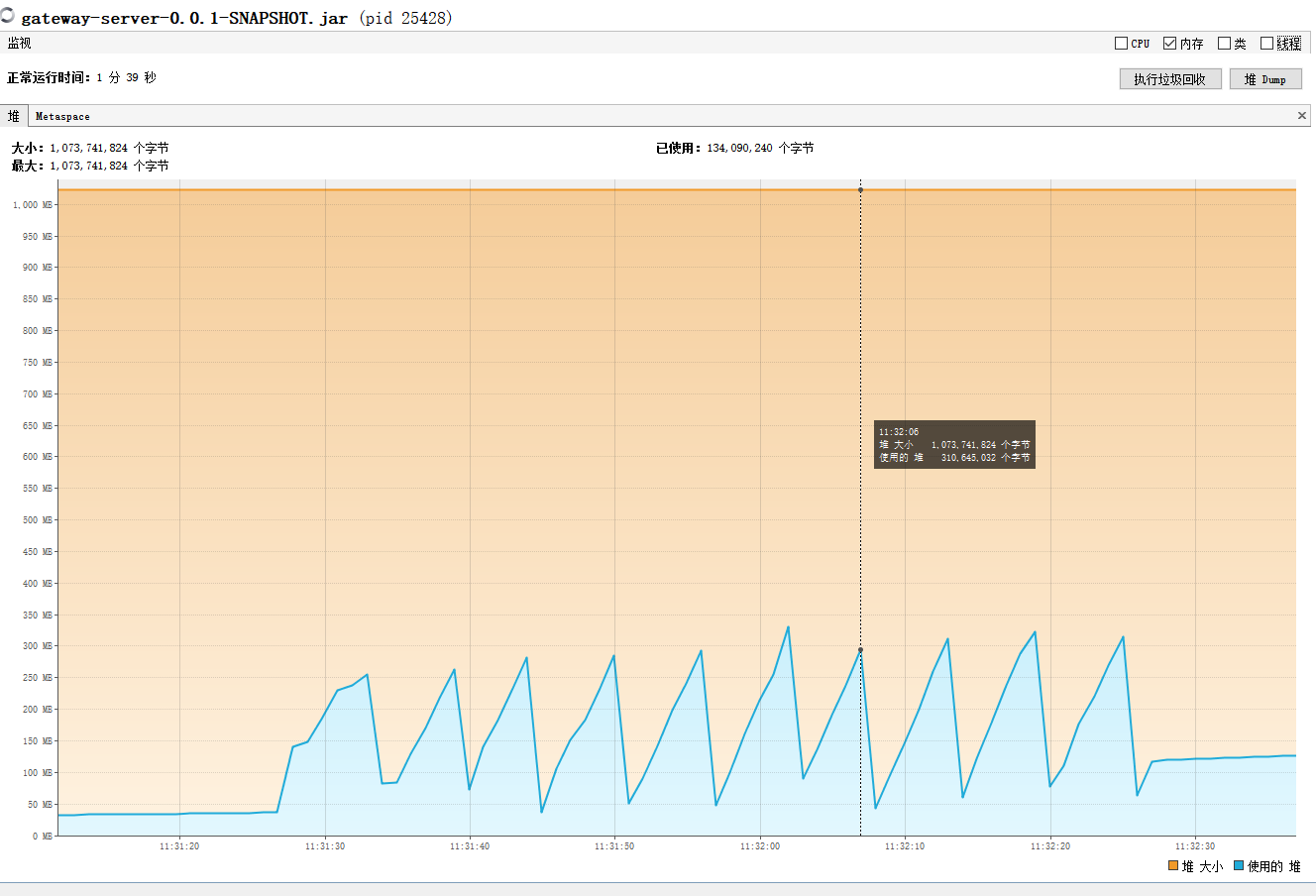
### 启动app并执行压测

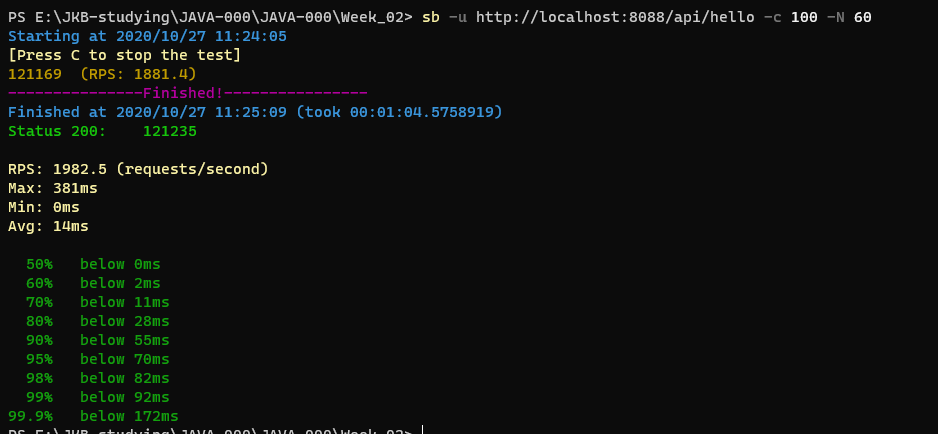
java -jar -Xms1g -Xmx1g -XX:+UseParallelGC .\gateway-server-0.0.1-SNAPSHOT.jar



sb -u http://localhost:8088/api/hello -c 100 -N 60

### 挂jvisualvm查看



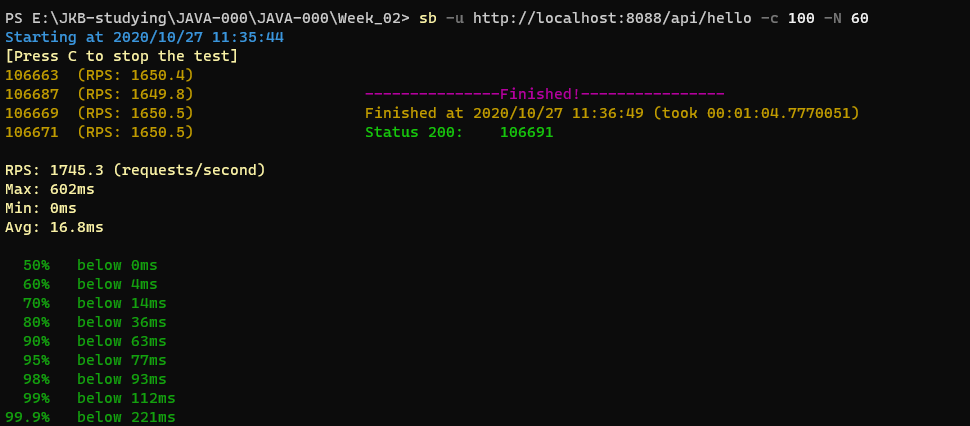


总结：RPS从6542->1982,最大延迟从53->381

### 挂JMC查看

执行jmc，命令启动jmc





总结，对比之前RPS从1982->1745,GC平均时间从3ms->6ms，说明JMC对项目的影响很大，一般不要挂。