```
# 查找到performance-1.0.0.jar的进程号
jcmd | grep "performance-1.0.0.jar" | awk '{print $1}'
# jmap 打印heap的概要信息，GC使用的算法，heap的配置及wise heap的使用情况
jmap -heap $(jcmd | grep "performance-1.0.0.jar" | awk '{print $1}')
# 收集GC日志（日志离线分析，主要用于检查故障看出是不是因为GC导致的程序卡顿）
# 不建议直接输出 java -Xmx1024m -XX:+PrintGCDetails -XX:+PrintGCTimeStamps -jar performance-1.0.0.jar
java -Xmx1024m -Xloggc:/netease/gc1.log -jar performance-1.0.0.jar
# 分析GC日志（）
GCViewer工具，辅助分析GC日志文件 https://github.com/chewiebug/GCViewer
# jstat 动态监控GC统计信息，间隔1000毫秒统计一次，每10行数据后输出列标题
jstat -gc -h10 $(jcmd | grep "performance-1.0.0.jar" | awk '{print $1}') 1000
```

```
[mwuser@SZ-CX-XZadmin-01 chinacri-pcarcore-platform-service]$ jmap -heap $(jcmd |grep "chinacri-pbase-admin-service"| awk '{print $1}
Attaching to process ID 22538, please wait...
Debugger attached successfully.
Server compiler detected.
JVM version is 25.171-b11

using thread-local object allocation.
Parallel GC with 4 thread(s)

Heap Configuration:
   MinHeapFreeRatio         = 0
   MaxHeapFreeRatio         = 100
   MaxHeapSize              = 536870912 (512.0MB)
   NewSize                  = 178782208 (170.5MB)
   MaxNewSize               = 178782208 (170.5MB)
   OldSize                  = 358088704 (341.5MB)
   NewRatio                 = 2
   SurvivorRatio            = 8
   MetaspaceSize            = 21807104 (20.796875MB)
   CompressedClassSpaceSize = 1073741824 (1024.0MB)
   MaxMetaspaceSize         = 17592186044415 MB
   G1HeapRegionSize         = 0 (0.0MB)

Heap Usage:
PS Young Generation
Eden Space:
   capacity = 175636480 (167.5MB)
   used     = 172325072 (164.3419952392578MB)
   free     = 3311408 (3.1580047607421875MB)
   98.1146240234375% used
From Space:
   capacity = 1572864 (1.5MB)
   used     = 1081344 (1.03125MB)
   free     = 491520 (0.46875MB)
   68.75% used
To Space:
   capacity = 1572864 (1.5MB)
   used     = 0 (0.0MB)
   free     = 1572864 (1.5MB)
   0.0% used
PS Old Generation
   capacity = 358088704 (341.5MB)
   used     = 131227176 (125.14798736572266MB)
   free     = 226861528 (216.35201263427734MB)
   36.64655559757618% used

46202 interned Strings occupying 4714296 bytes.
[mwuser@SZ-CX-XZadmin-01 chinacri-pcarcore-platform-service]$
```

Parallel GC with 4 thread(s) 并发收集器

Heap Configuration: //堆内存初始化配置

MinHeapFreeRatio=0 //对应jvm启动参数-XX:MinHeapFreeRatio设置JVM堆最小空闲比率(default 0)

MaxHeapFreeRatio=100 //对应jvm启动参数 -XX:MaxHeapFreeRatio设置JVM堆最大空闲比率(default 100)

MaxHeapSize=512.0MB //对应jvm启动参数-XX:MaxHeapSize=设置JVM堆的最大大小

NewSize = 170MB //对应jvm启动参数-XX:NewSize=设置JVM堆的‘新生代’的默认大小

MaxNewSize =170MB //对应jvm启动参数-XX:MaxNewSize=设置JVM堆的‘新生代’的最大大小

OldSize = 341MB //对应jvm启动参数-XX:OldSize=<value>:设置JVM堆的‘老生代’的大小

NewRatio = 2 //对应jvm启动参数-XX:NewRatio=:‘新生代’和‘老生代’的大小比率

```
                SurvivorRatio = 8        //对应jvm启动参数-XX:SurvivorRatio=设置年轻代中Eden区与Survivor区的大小比值

                  Heap Usage:                                  //堆内存分步
                  PS Young Generation
                  Eden Space:                        //Eden区内存分布
                  capacity =167MB //Eden区总容量
                     used          = 164MB    //Eden区已使用
                  free          = 3.15MB //Eden区剩余容量
                      98.114% used    //Eden区使用比率
                From Space:                 //其中一个Survivor区的内存分布
                    capacity = 1.5MB
                   used          = 1.03MB
                   free          = 0.468MB
                        68.75% used
              To Space:                             //另一个Survivor区的内存分布
                      capacity = 1.5MB
                      used          = 0 (0.0MB)
                free          =1.5MB
                0.0% used
              PS Old Generation    //当前的Old区内存分布
                capacity = 341MB
                used          = 125MB
                free          = 216MB
                35.64% used
```

```
applogs  appsystem  gc1.log
[mwuser@SZ-CX-XZadmin-01 mwbase]$ tail -f gc1.log
27.173: [GC (Allocation Failure)  134552K->51040K(260608K), 0.0146156 secs]
27.378: [GC (Allocation Failure)  135008K->51888K(260608K), 0.0092532 secs]
27.579: [GC (Allocation Failure)  135856K->52520K(260608K), 0.0060750 secs]
27.931: [GC (Allocation Failure)  136488K->53356K(260608K), 0.0081914 secs]
28.405: [GC (Allocation Failure)  137324K->56124K(258560K), 0.0096474 secs]
28.680: [GC (Allocation Failure)  138044K->56900K(259584K), 0.0085658 secs]
28.990: [GC (Allocation Failure)  138820K->57773K(259584K), 0.0062232 secs]
29.296: [GC (Allocation Failure)  139693K->58549K(258048K), 0.0068543 secs]
29.465: [GC (Allocation Failure)  140469K->58909K(259072K), 0.0075914 secs]
39.654: [GC (Allocation Failure)  140829K->59269K(259072K), 0.0075738 secs]
56.716: [GC (Allocation Failure)  141189K->59533K(260096K), 0.0097153 secs]
57.181: [GC (Allocation Failure)  142477K->60109K(260096K), 0.0097485 secs]
57.519: [GC (Allocation Failure)  143053K->60269K(260096K), 0.0062081 secs]
58.108: [GC (Allocation Failure)  143213K->60813K(260096K), 0.0118765 secs]
```

因为在年轻代中没有足够的空间能够存储新的数据,进行了一次垃圾回收，进行Minor GC

143213K->60813k(260096k)

堆区垃圾回收前的大小143213k，堆区垃圾回收后的大小60813k，堆区总大小260096k

heap区总共减少 143213k-60813k =82400k =80M

耗时 0.0119765 秒

```
[mwuser@SZ-CX-XZadmin-01 [redacted]pbase-admin-service]$ jstat -gc -h10 $(jcmd | grep "[redacted]pbase-admin-servic
e.jar" | awk '{print $1}') 1000
 S0C    S1C    S0U    S1U      EC       EU        OC         OU       MC     MU    CCSC   CCSU   YGC     YGCT
FGC    FGCT     GCT
3072.0 3072.0 288.0   0.0   168448.0 39215.2   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.
004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 39770.8   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 40105.2   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 40517.6   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 40898.2   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 41327.7   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 41788.0   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 42360.3   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 42776.6   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 43157.0   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
 S0C    S1C    S0U    S1U      EC       EU        OC         OU       MC     MU    CCSC   CCSU   YGC     YGCT    FGC    FGCT     GCT
3072.0 3072.0 288.0   0.0   168448.0 43589.7   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 43962.9   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 44360.8   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 44895.6   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 45220.9   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 45782.5   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 46192.2   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
3072.0 3072.0 288.0   0.0   168448.0 46868.7   349696.0   66363.2   101120.0 96839.8 13824.0 13060.8    116    1.004   4      0.857    1.861
```

发生fullgc次数4次,累计耗时

S0C：第一个幸存区的大小

S1C：第二个幸存区的大小

S0U：第一个幸存区的使用大小

S1U：第二个幸存区的使用大小

EC：egen的大小

EU：egen区的使用大小

OC：老年代大小

OU：老年代使用大小

MC：方法区大小

MU：方法区使用大小

CCSC:压缩类空间大小

CCSU:压缩类空间使用大小

YGC：年轻代垃圾回收次数

YGCT：年轻代垃圾回收消耗时间

FGC：老年代垃圾回收次数

FGCT：老年代垃圾回收消耗时间

GCT：垃圾回收消耗总时间

jmap转储可以使用如下方式:

jmap -dump:file=DumpFileName.txt,format=b pid

例如:

C:\Users\Administrator\Desktop>jmap -dump:file=D:/javaDump.hprof,format=b 3614

Dumping heap to D:\javaDump.hprof ...

Heap dump file created

D:\daily\java\jdk\jdk1.8.0_131\bin>jhat -J-Xmx4096m D:/24420.dump