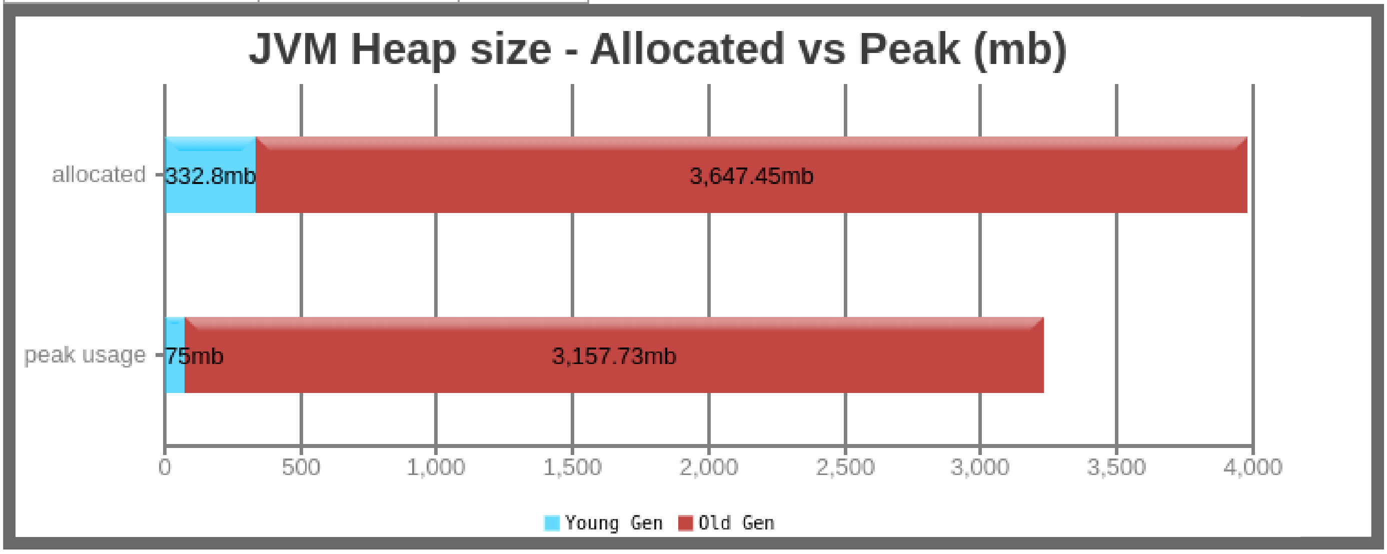
Analysis Report

JVM Heap Size

Generation	Allocated ❓	Peak ❓
Young Generation	332.8 mb	75 mb
Old Generation	3.56 gb	3.08 gb
Total	3.89 gb	3.09 gb



Key Performance Indicators

(Important section of the report. To learn more about KPIs, click here (https://blog.gceasy.io/2016/10/01/garbage-collection-kpi/))

1 **Throughput ❓ : 58.112%**

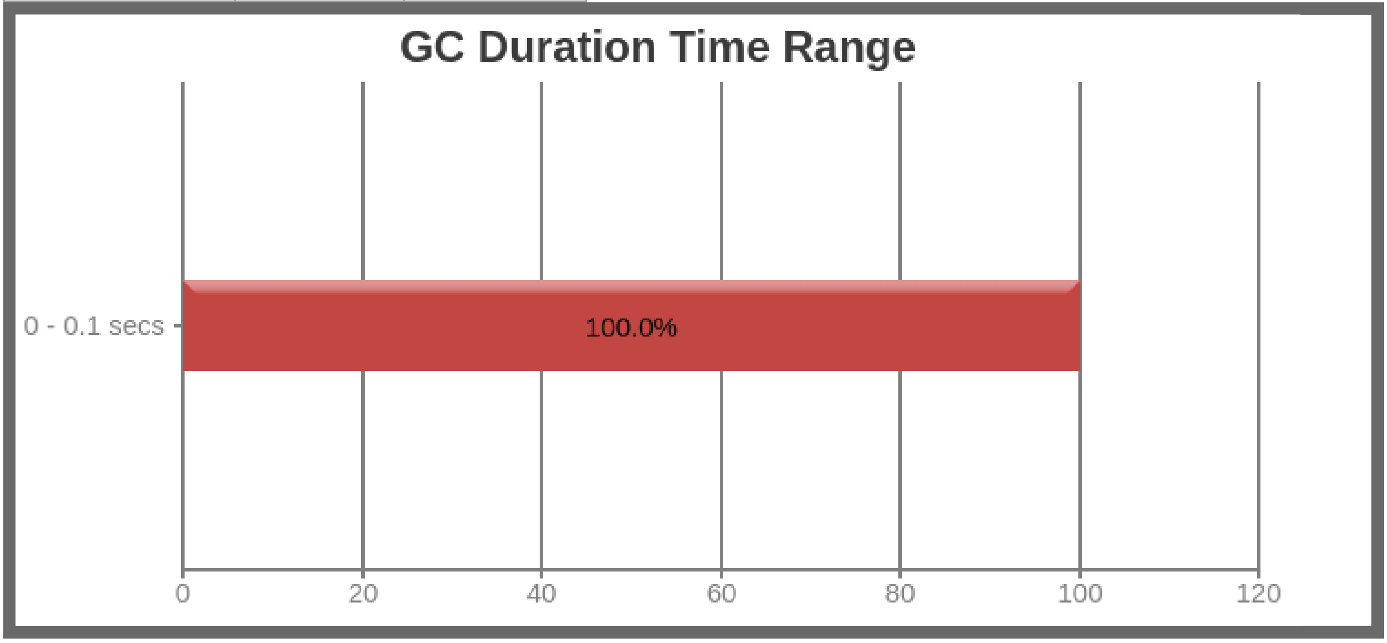
2 **Latency:**

Avg Pause GC Time ❓	**15 ms**
Max Pause GC Time ❓	**30 ms**

GC **Pause** Duration Time Range ❓:

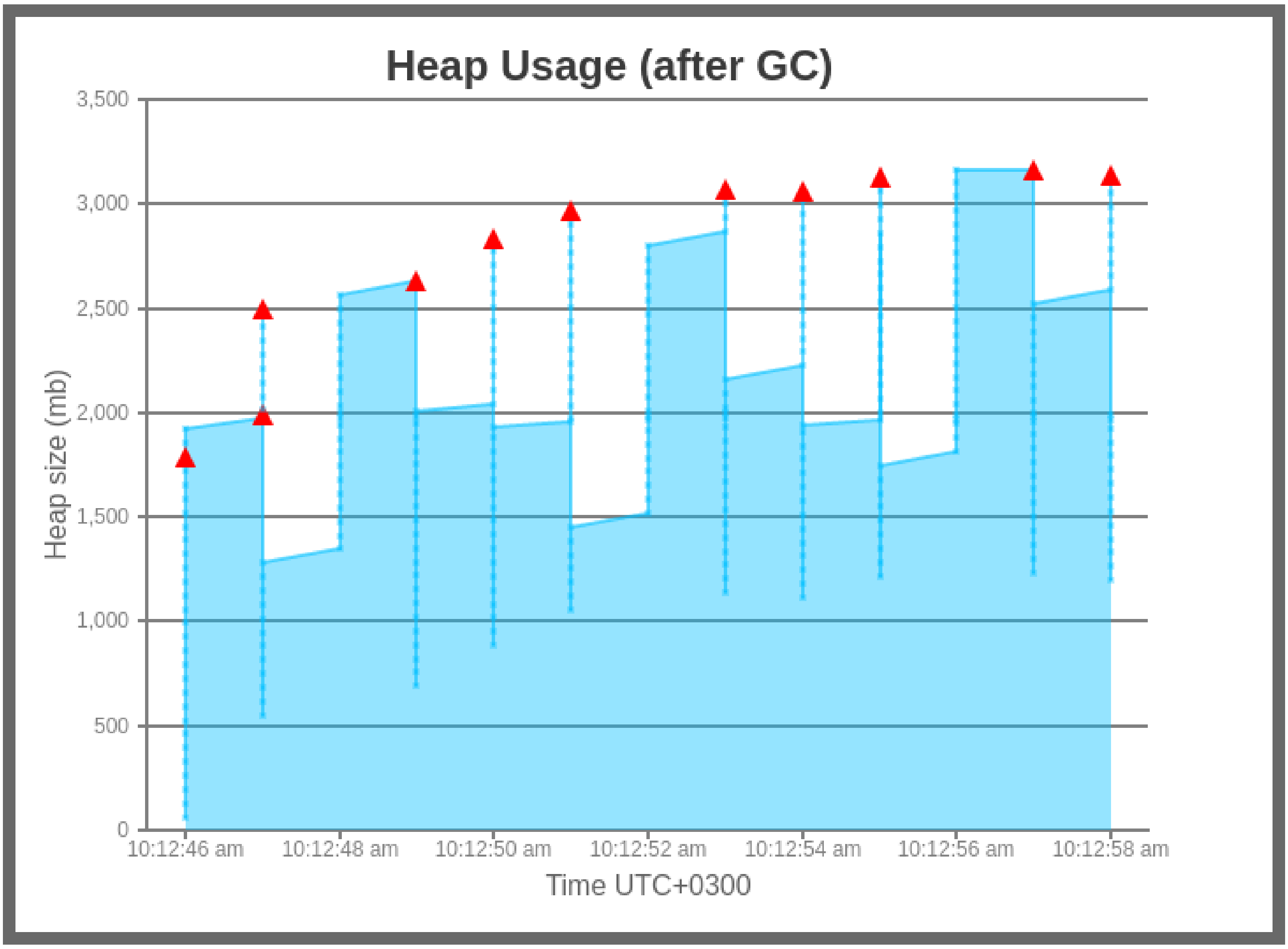
Duration (secs)	No. of GCs	Percentage

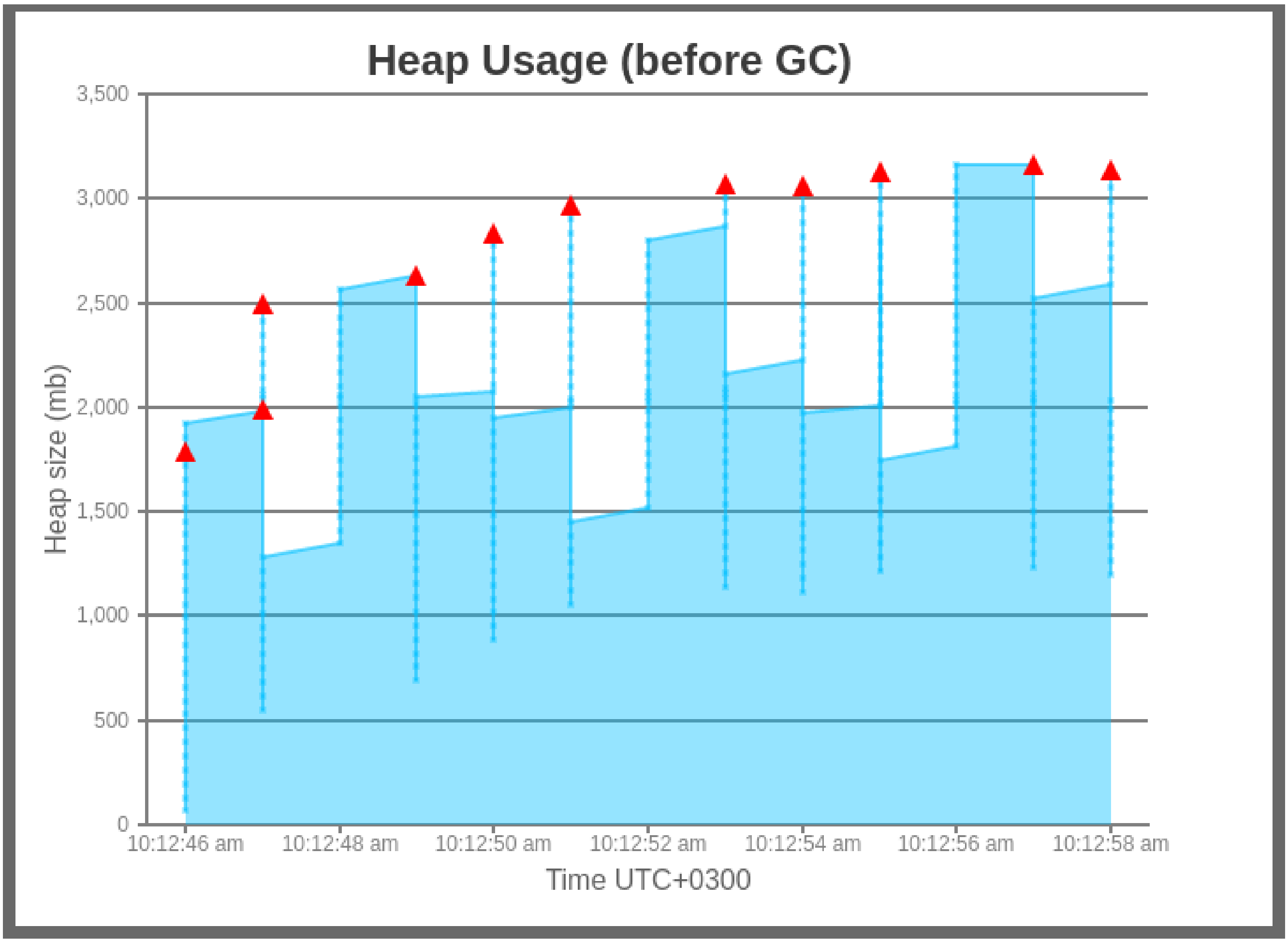
0 - 0.1	333	100.0%

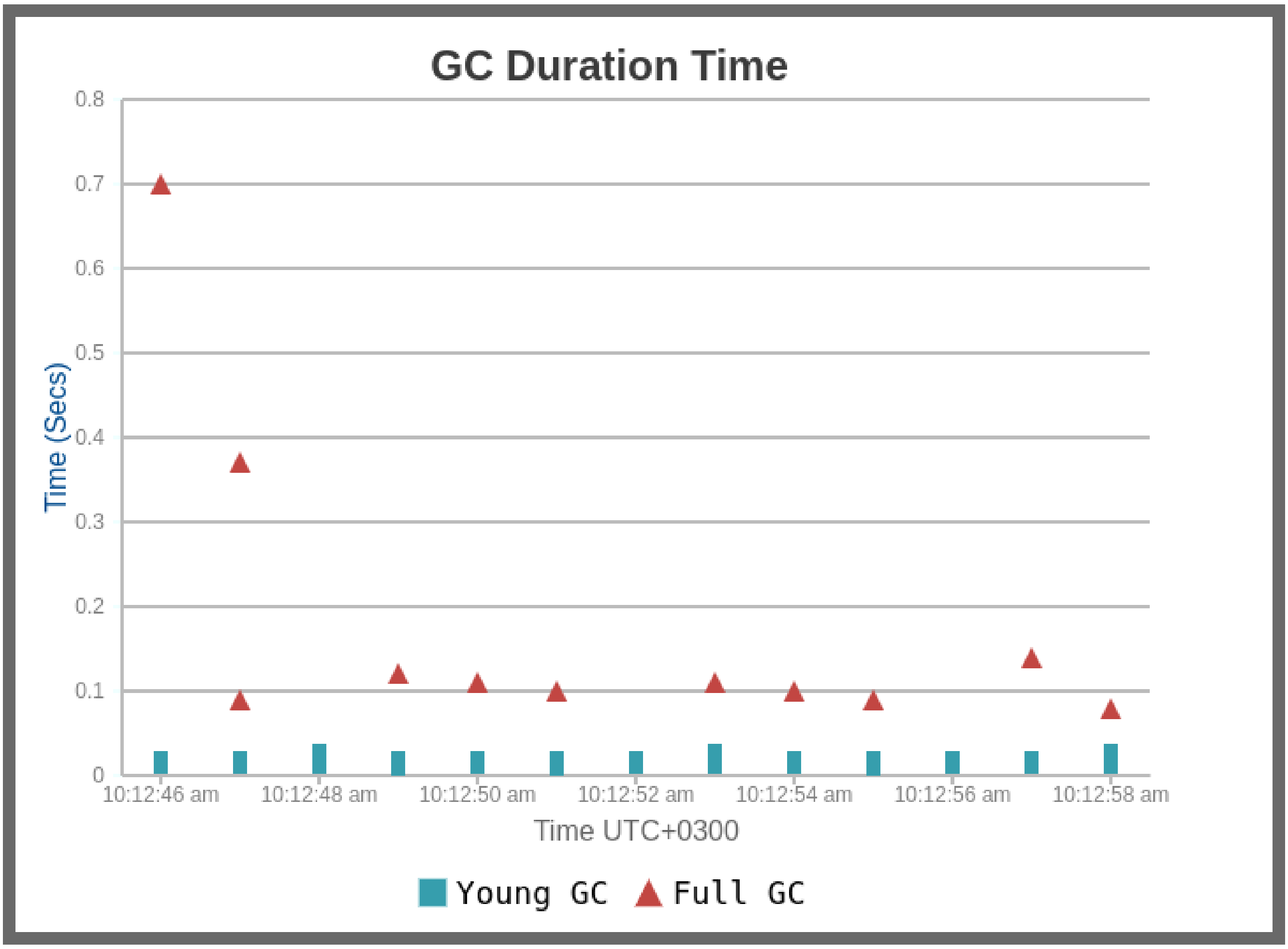


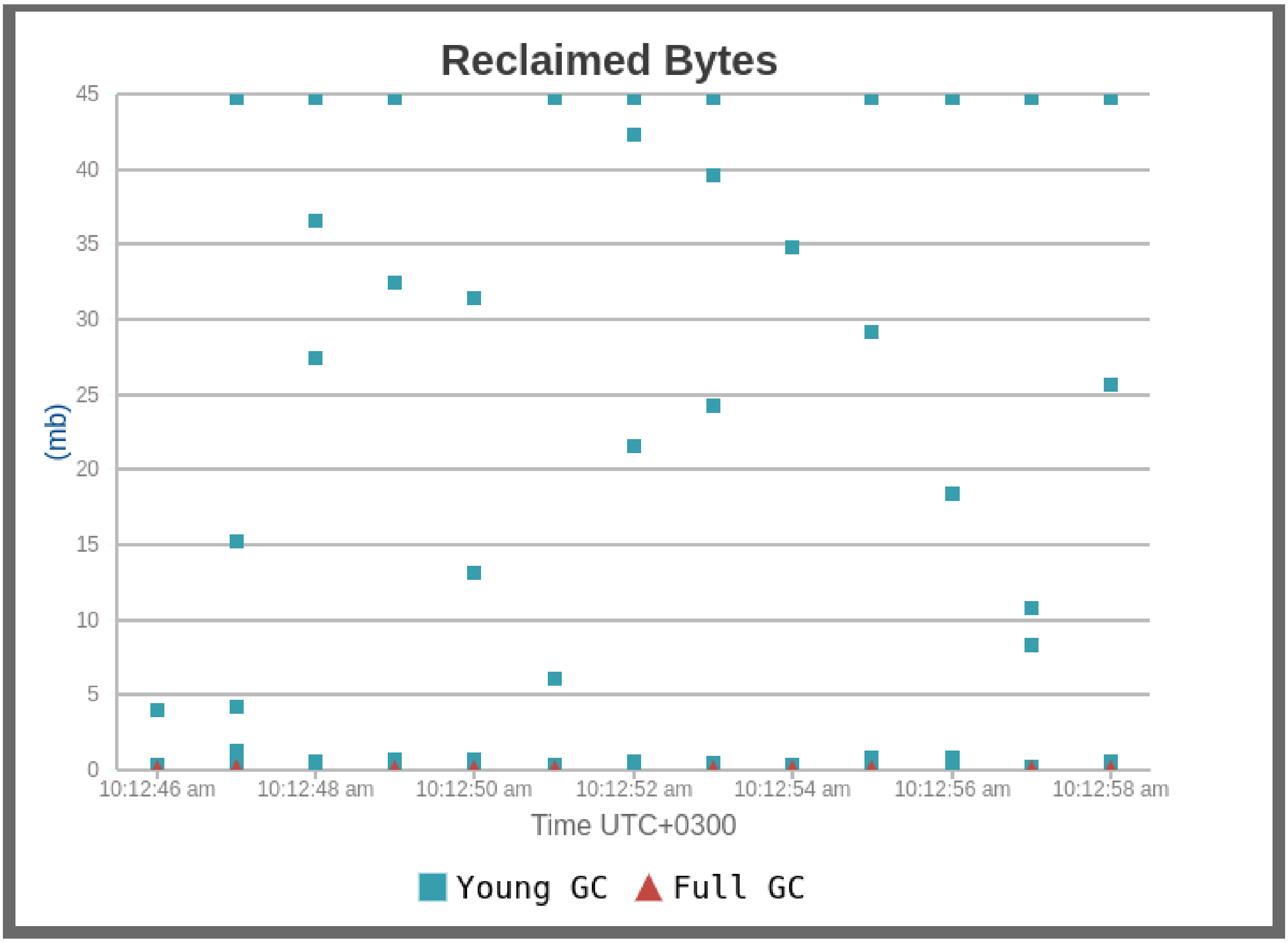
Interactive Graphs

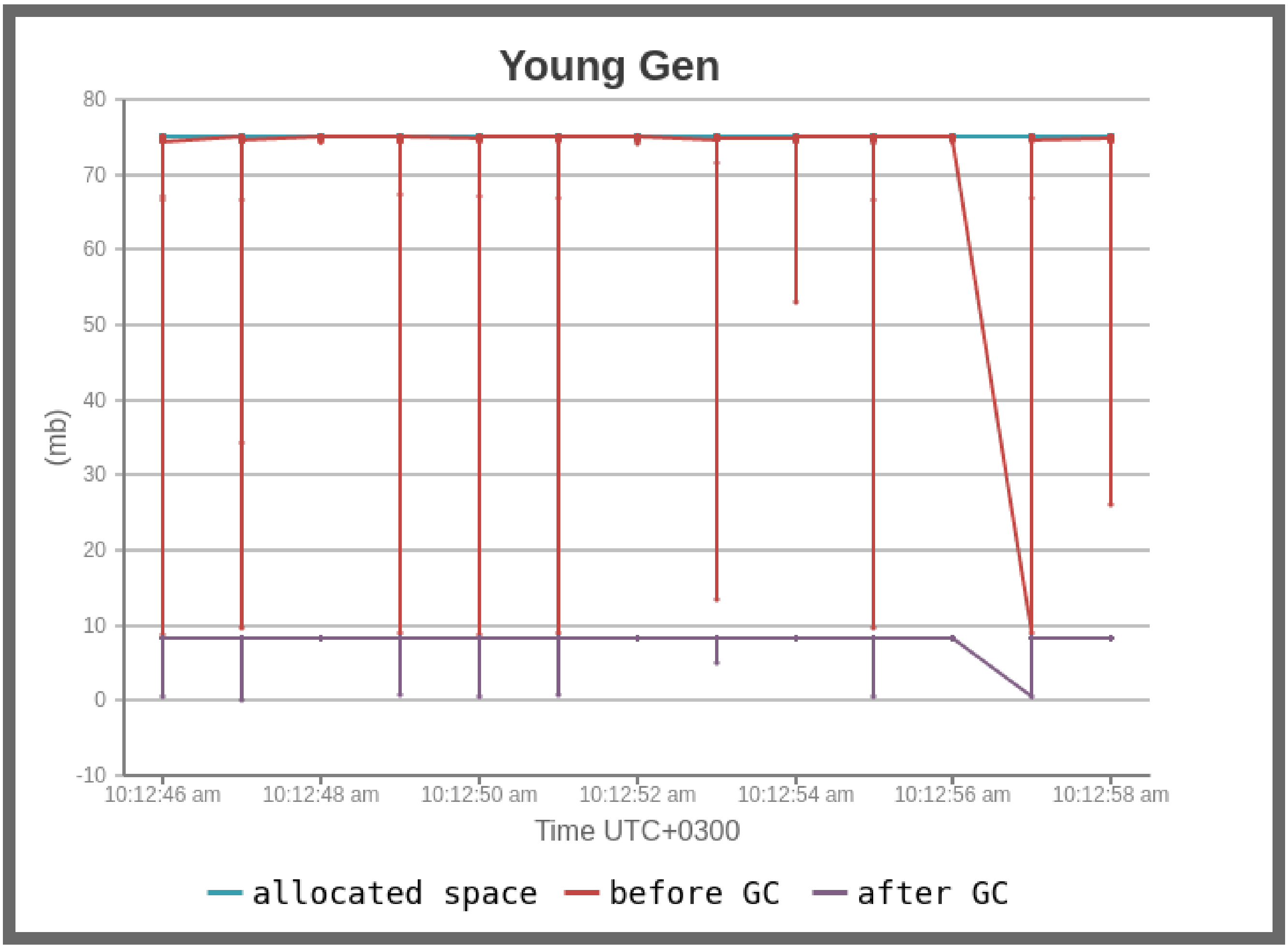
(All graphs are zoomable)

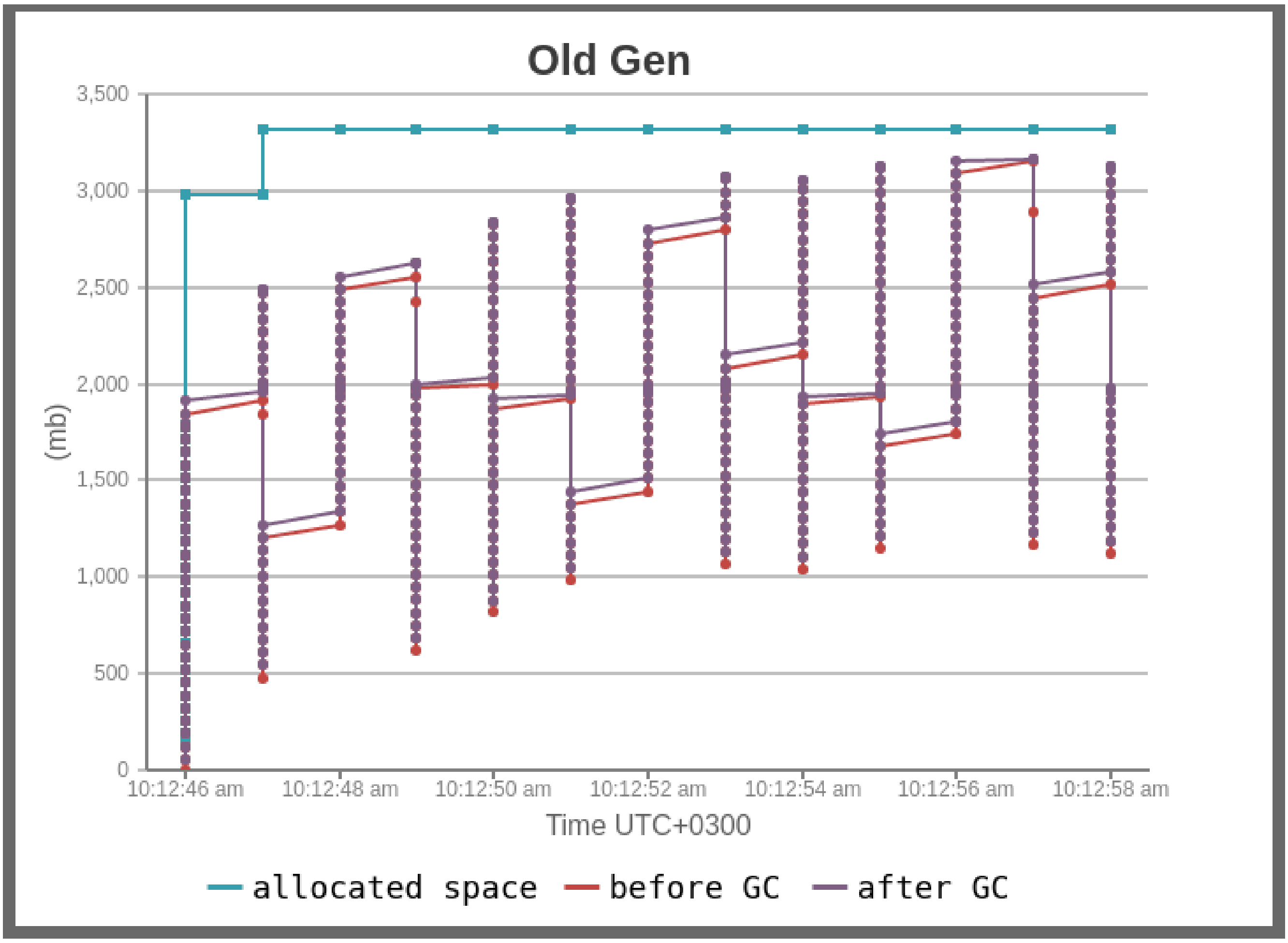


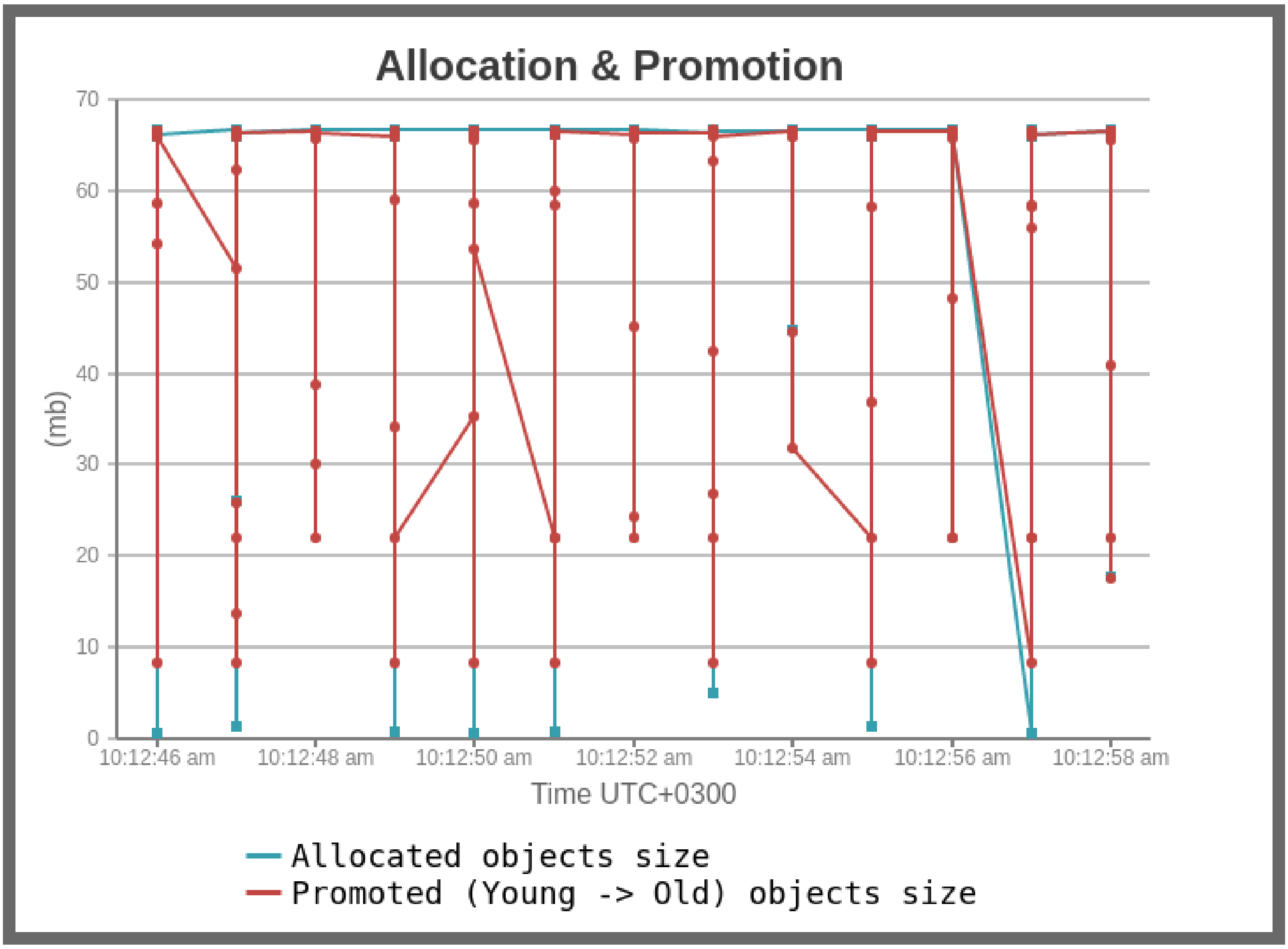

Heap Usage (before GC)
Heap size (mb)
3,500
3,000
2,500
2,000
1,500
1,000
500
0
10:12:46 am
10:12:48 am
10:12:50 am
10:12:52 am
10:12:54 am
10:12:56 am
10:12:58 am
Time UTC+0300


GC Duration Time
Time (Secs)
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
10:12:46 am
10:12:48 am
10:12:50 am
10:12:52 am
10:12:54 am
10:12:56 am
10:12:58 am
Time UTC+0300
Young GC
Full GC

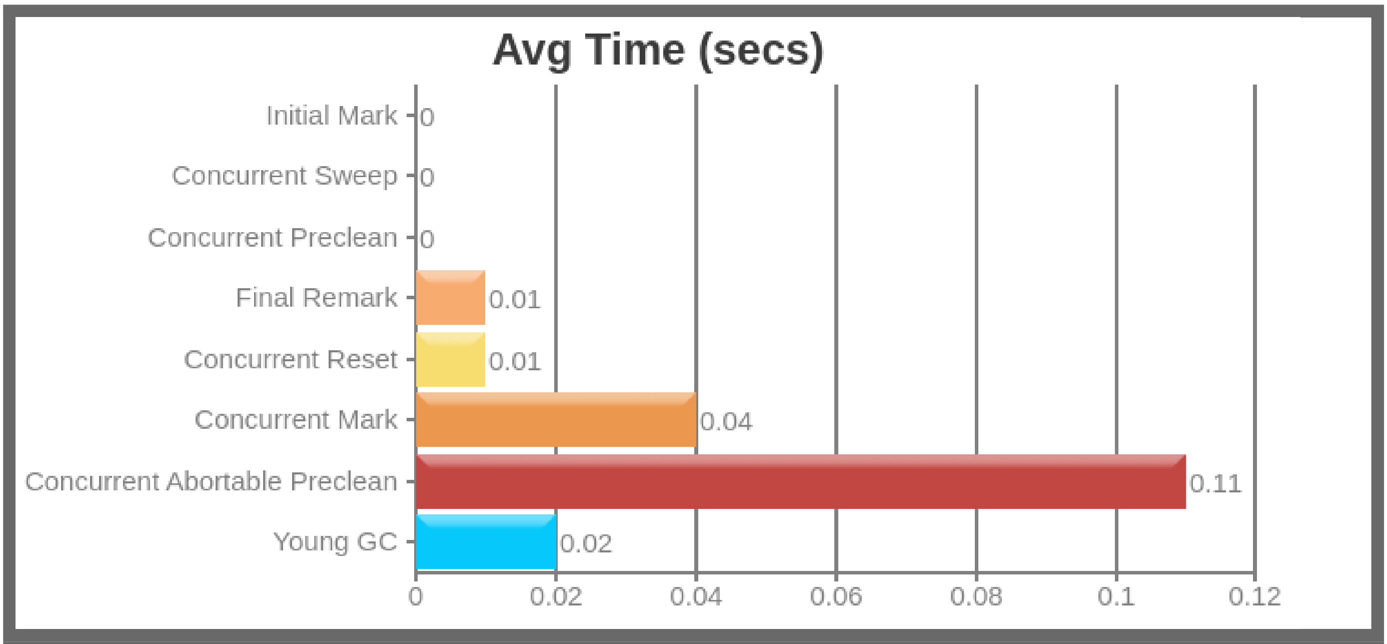

Reclaimed Bytes
45
40
35
30
25
20
15
10
5
0
(mb)
10:12:46 am
10:12:48 am
10:12:50 am
10:12:52 am
10:12:54 am
10:12:56 am
10:12:58 am
Time UTC+0300
Young GC
Full GC

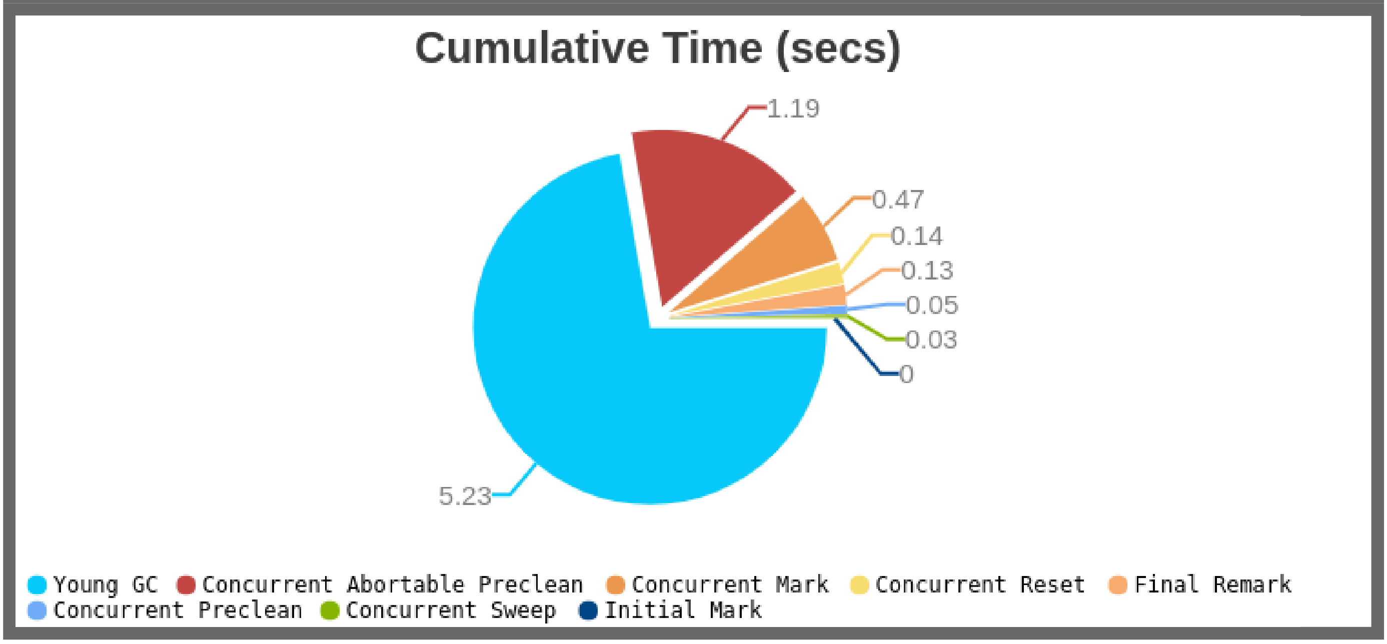

Young Gen
80
70
60
50
40
30
20
10
0
-10
(mb)
10:12:46 am
10:12:48 am
10:12:50 am
10:12:52 am
10:12:54 am
10:12:56 am
10:12:58 am
Time UTC+0300
allocated space
before GC
after GC


Old Gen
3,500
3,000
2,500
2,000
1,500
1,000
500
0
(mb)
10:12:46 am
10:12:48 am
10:12:50 am
10:12:52 am
10:12:54 am
10:12:56 am
10:12:58 am
Time UTC+0300
allocated space
before GC
after GC


Allocation & Promotion
70
60
50
40
30
20
10
0
(mb)
10:12:46 am
10:12:48 am
10:12:50 am
10:12:52 am
10:12:54 am
10:12:56 am
10:12:58 am
Time UTC+0300
Allocated objects size
Promoted (Young -> Old) objects size

CMS Collection Phases Statistics

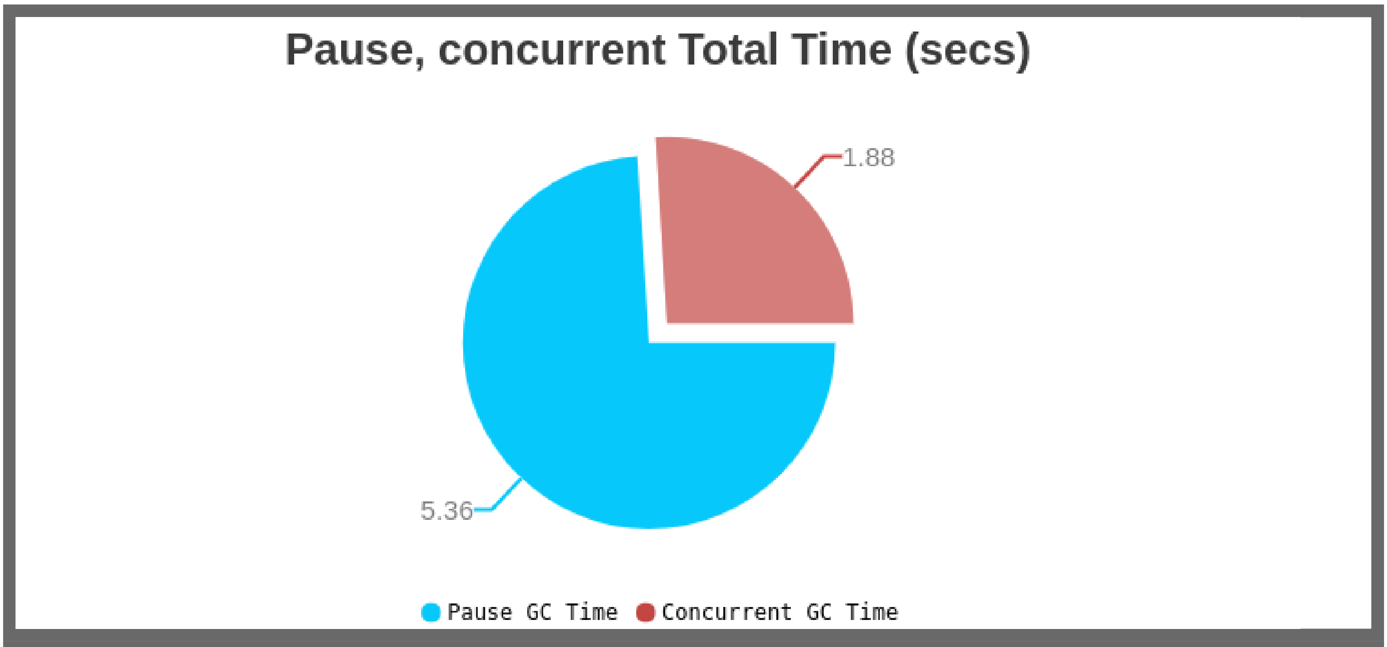




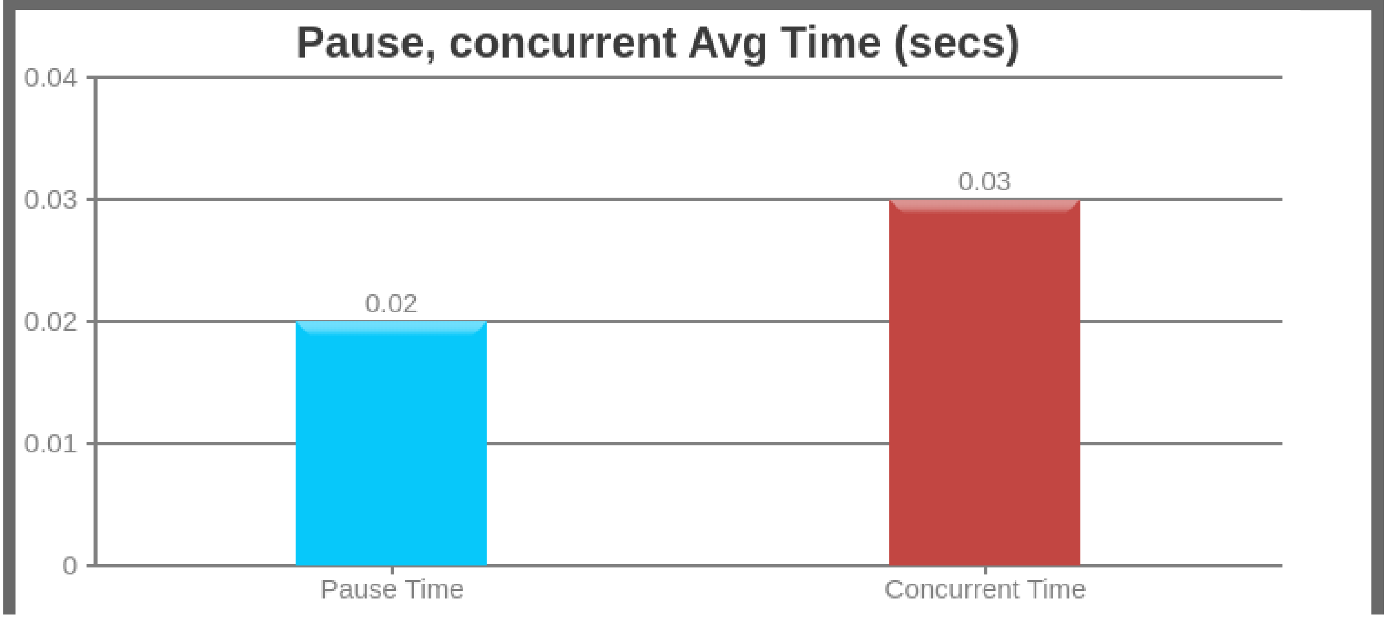
	Young GC ⏺	Concurrent Abortable Preclean	Concurrent Mark	Concurrent Reset	Final Remark ⏺	Concurrent Preclean	Concurrent Sweep	Initial Mark ⏺
Total Time ❓	5 sec 230 ms	1 sec 190 ms	470 ms	140 ms	130 ms	50 ms	30 ms	0
Avg Time ❓	16 ms	108 ms	43 ms	13 ms	12 ms	5 ms	3 ms	0

Std Dev Time	5 ms	189 ms	10 ms	4 ms	7 ms	7 ms	4 ms	0
Min Time ❓	0	10 ms	20 ms	10 ms	0	0	0	0
Max Time ❓	30 ms	660 ms	60 ms	20 ms	20 ms	20 ms	10 ms	0
Count ❓	327	11	11	11	11	11	11	11

CMS GC Time



Pause, concurrent Total Time (secs)



Pause, concurrent Avg Time (secs)

Pause Time

Total Time	5 sec 360 ms
Avg Time	15 ms
Std Dev Time	6 ms
Min Time	0
Max Time	30 ms

Concurrent Time

Total Time	1 sec 880 ms
Avg Time	34 ms
Std Dev Time	94 ms
Min Time	0
Max Time	660 ms

Object Stats

(These are perfect micro-metrics (https://blog.gceasy.io/2017/05/30/improving-your-performance-reports/) to include in your performance reports)

Total created bytes	20.03 gb
Total promoted bytes	19.53 gb
Avg creation rate	1.57 gb/sec
Avg promotion rate	1.53 gb/sec

Memory Leak

No major memory leaks.

(**Note:** there are 8 flavours of OutOfMemoryErrors (https://tier1app.files.wordpress.com/2014/12/outofmemoryerror2.pdf). With GC Logs you can diagnose only 5 flavours of them(Java heap space, GC overhead limit exceeded, Requested array size exceeds VM limit, Permgen space, Metaspace). So in other words, your application could be still suffering from memory leaks, but need other tools to diagnose them, not just GC Logs.)

Consecutive Full GC

None.

Long Pause

None.

Safe Point Duration

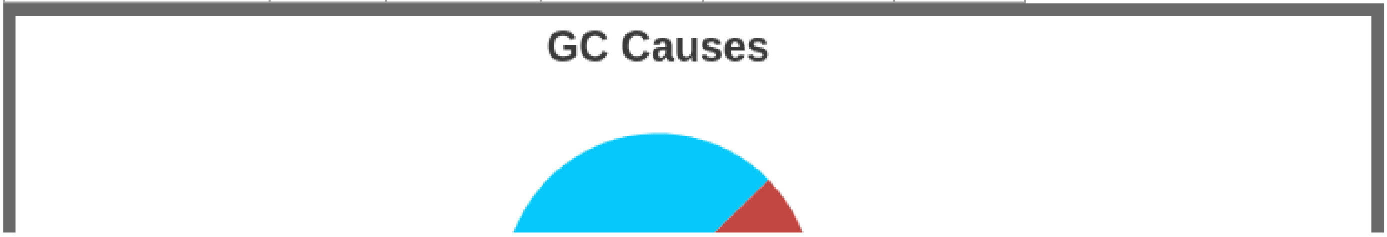
(To learn more about SafePoint duration, click here (https://blog.gceasy.io/2016/12/22/total-time-for-which-application-threads-were-stopped/))

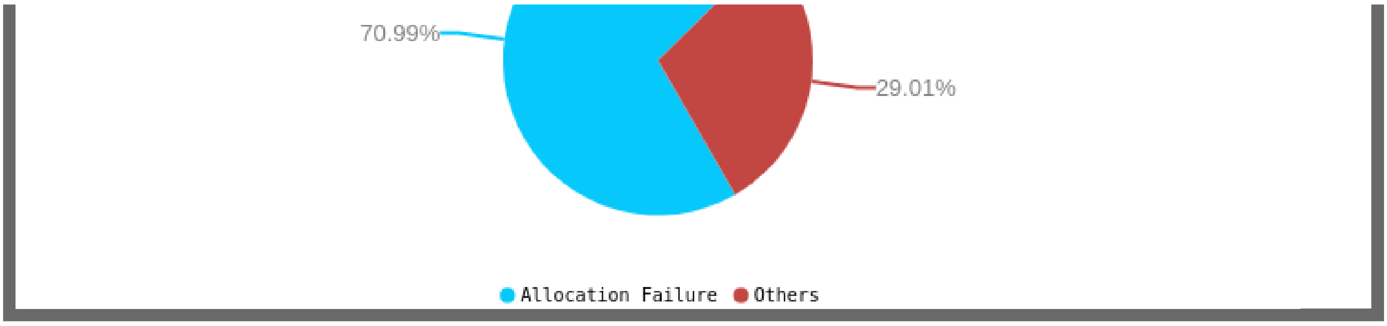
Not Reported in the log.

GC Causes

(What events caused the GCs, how much time it consumed?)

Cause	Count	Avg Time	Max Time	Total Time	Time %
Allocation Failure	316	16 ms	30 ms	5 sec 140 ms	70.99%
Others	22	n/a	n/a	2 sec 100 ms	29.01%
Total	338	n/a	n/a	7 sec 240 ms	100.0%





Tenuring Summary

Not reported in the log.

Command Line Flags

-XX:+CMSScavengeBeforeRemark -XX:InitialHeapSize=260849920 -XX:MaxHeapSize=4173598720 -XX:MaxNewSize=348966912 -XX:MaxTenuringThreshold=6 -XX:OldPLABSize=16 -XX:+PrintGC -XX:+PrintGCDateStamps -XX:+PrintGCDetails -XX:+PrintGCTimeStamps -XX:+UseCompressedClassPointers -XX:+UseCompressedOops -XX:+UseConcMarkSweepGC -XX:+UseParNewGC