

GC Intelligence Report

[zgc_3.log](#)

Duration: 4 hrs 24 min 18 sec

Congratulations!

Your application's GC activity is healthy.

Recommendations

(CAUTION: Please do thorough testing before implementing below recommendations.)

- ✓ 0.443 ms of GC pause time is triggered by 'Metadata GC Threshold' event. This type of GC event is triggered under two circumstances:
 1. Configured metaspace size is too small than the actual requirement
 2. There is a classloader leak (very unlikely, but possible).

Solution:

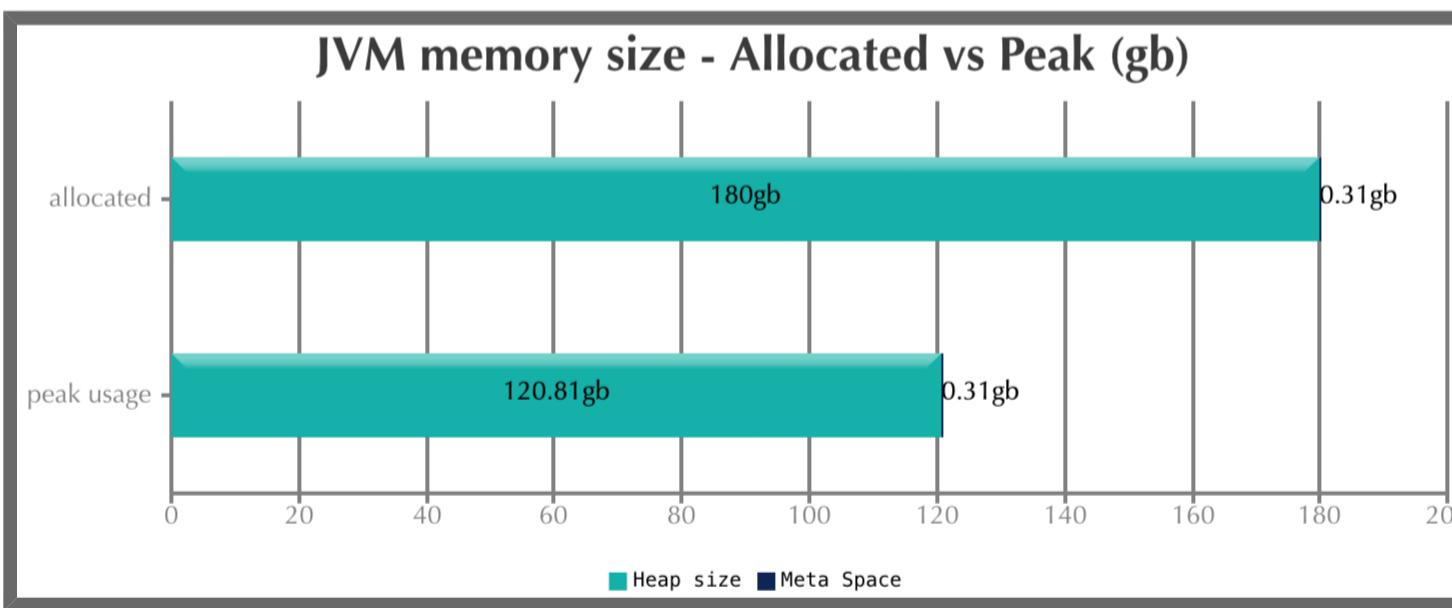
You may consider setting '-XX:MaxMetaspaceSize' to a higher value. If this property is not present already please configure it. Setting these arguments to a higher value will reduce 'Metadata GC Threshold' frequency. If you still continue to see 'Metadata GC Threshold' event reported, then you need to inspect metaspace contents. Learn how to inspect metaspace contents from [this article](#).

- ✓ It looks like you are using ZGC algorithm. If you are running on Java 18 and above, you may consider passing **-XX:+UseStringDeduplication** to your application. It will remove duplicate strings in your application and has potential to improve overall application's performance. You can learn more about this property in [this article](#).
- ✓ This application is using the ZGC algorithm. If you are looking to tune ZGC performance even further, here are the [important ZGC algorithm related JVM arguments](#)

JVM memory size

(To learn about JVM Memory, [click here](#))

Region	Allocated	Peak
Heap	180 gb	120.81 gb
Metaspace	320 mb	314 mb
Total	180.31 gb	121.12 gb



Key Performance Indicators

(Important section of the report. To learn more about KPIs, [click here](#))

1 Throughput : 99.999%

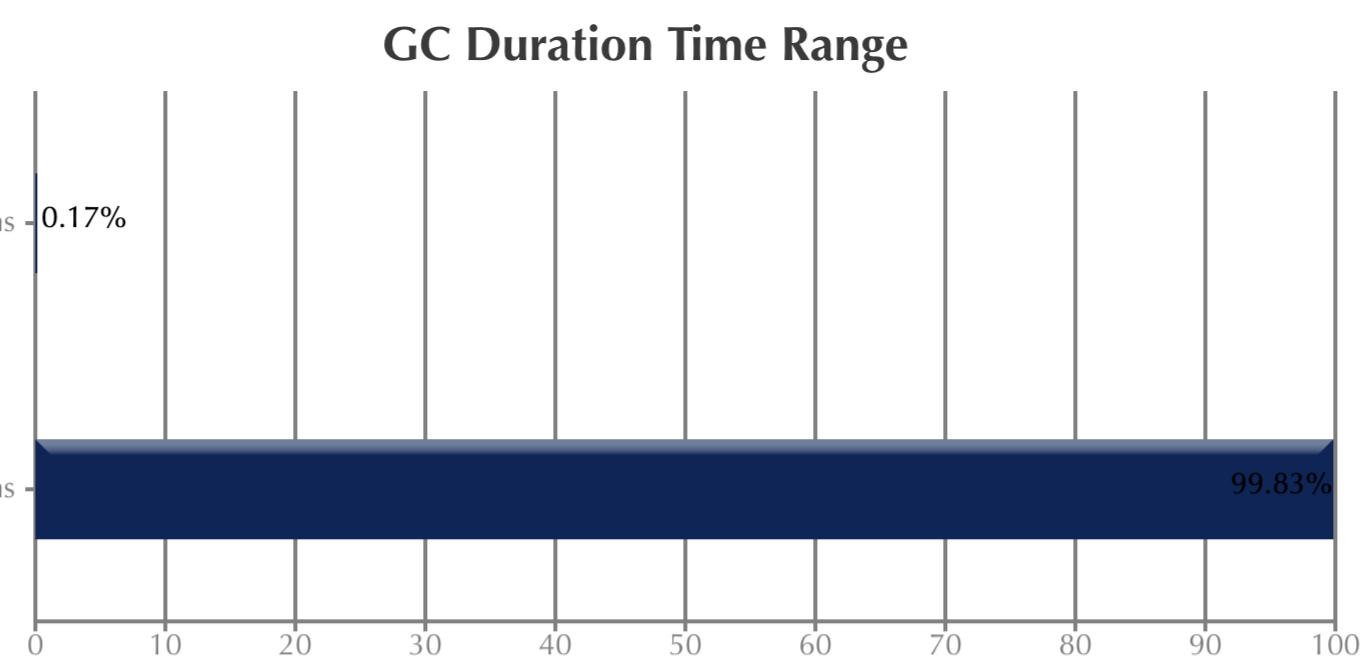
2 CPU Time : n/a

3 Latency:

Avg Pause GC Time <small>?</small>	0.178 ms
Max Pause GC Time <small>?</small>	1.17 ms

GC Pause Duration Time Range ?:

Duration (ms)	No. of GCs	Percentage
1 ms	1147	99.83%
1 - 2	2	0.17%

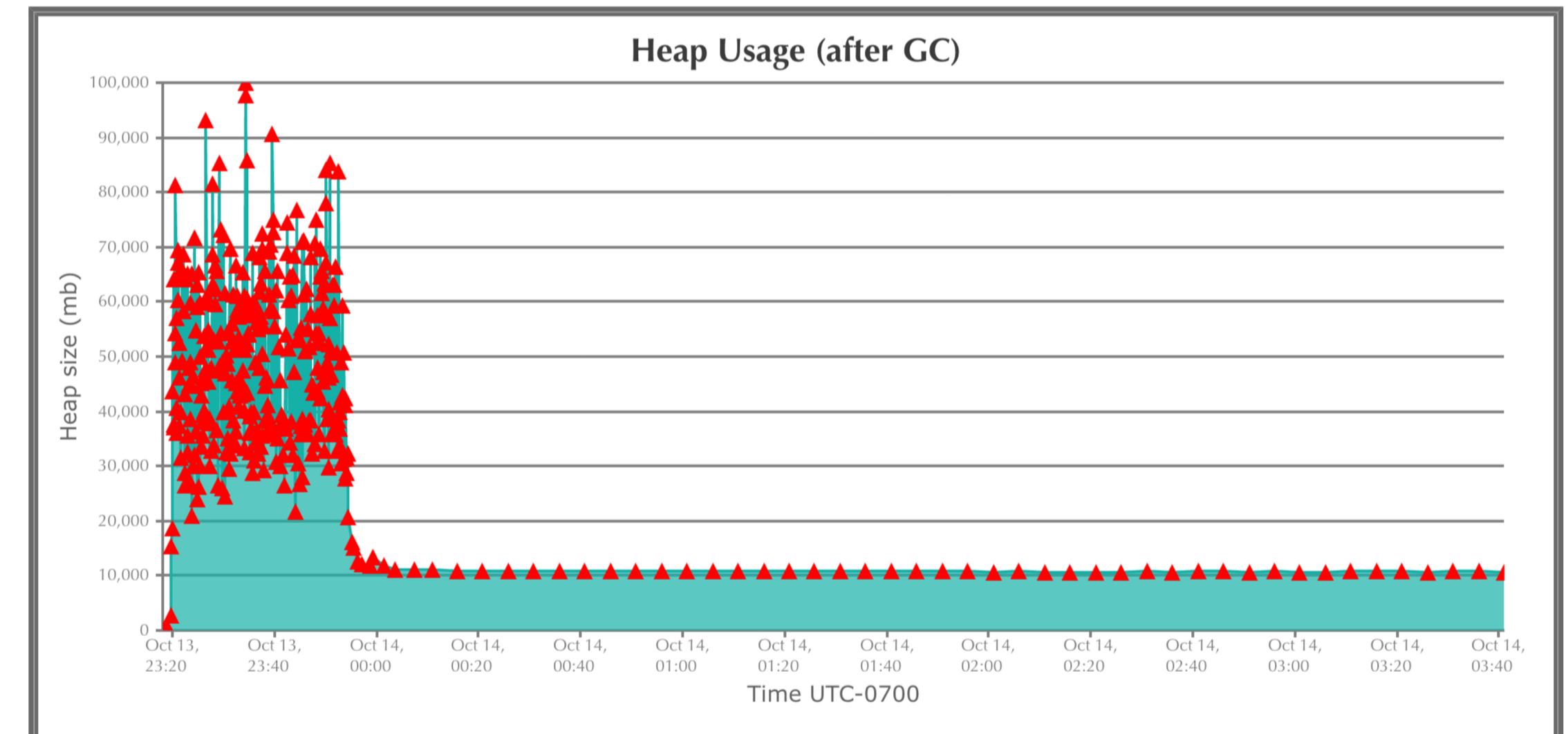


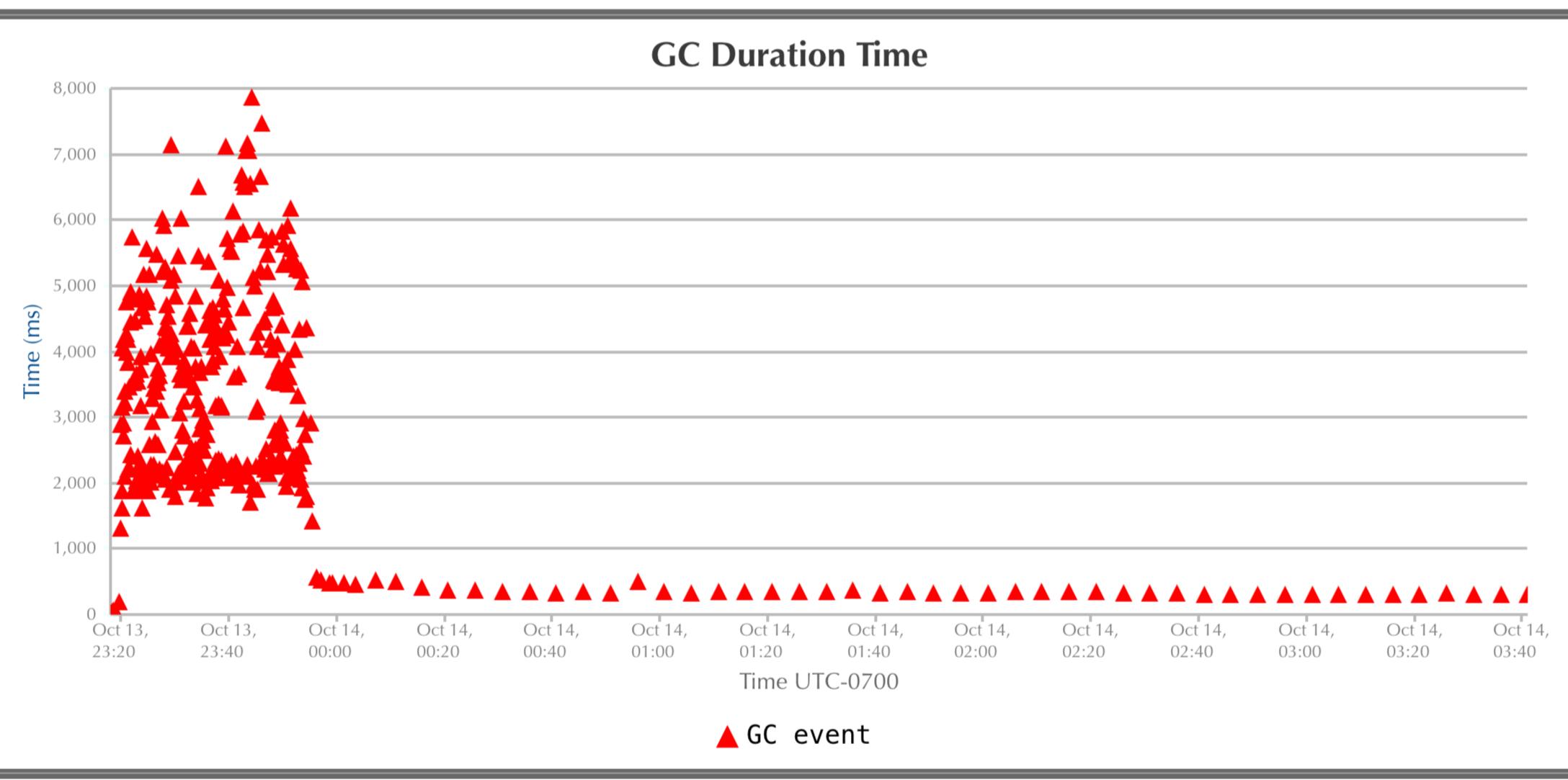
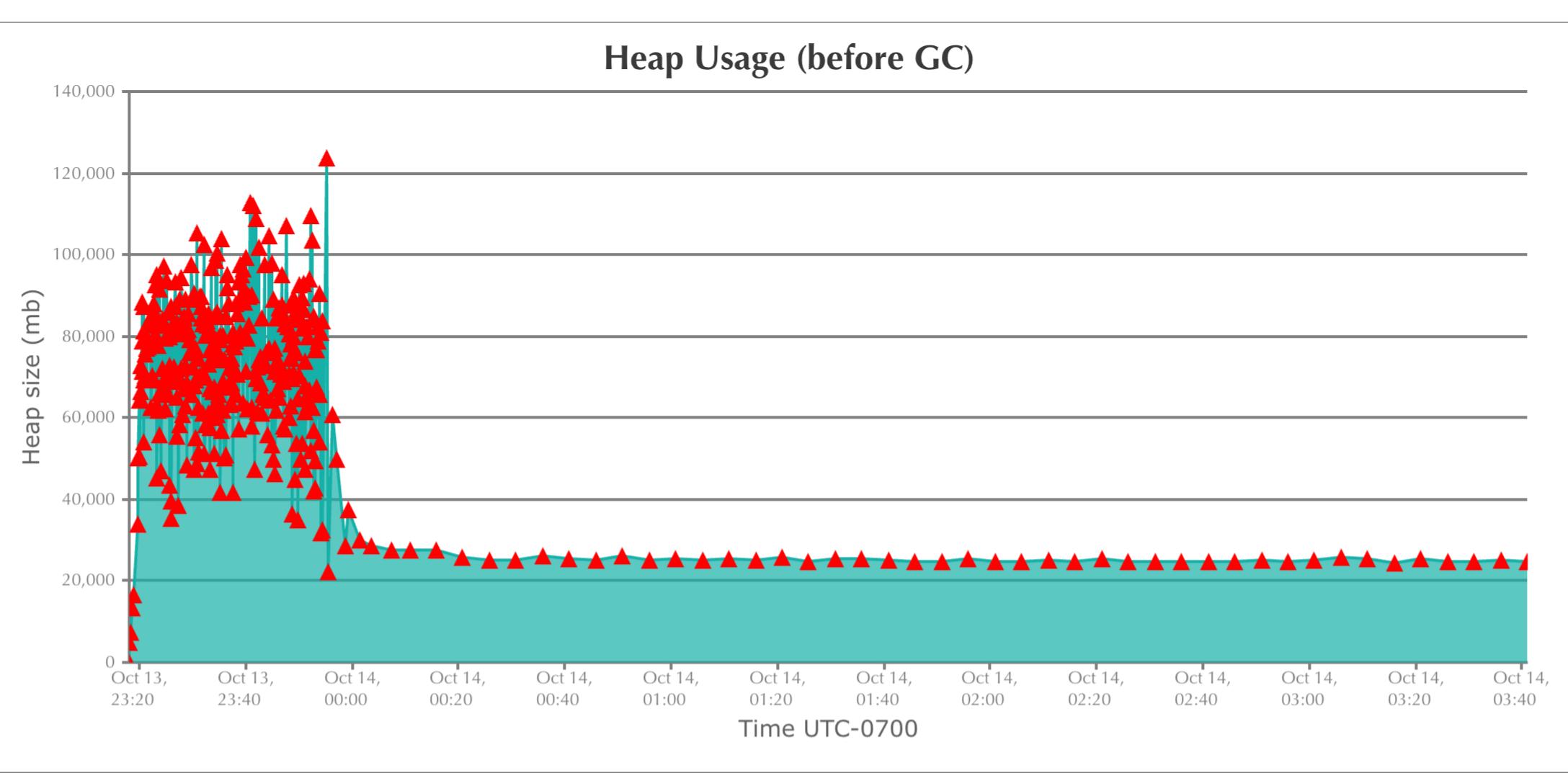
▼ Analyze Specific Time Periods (Beta)

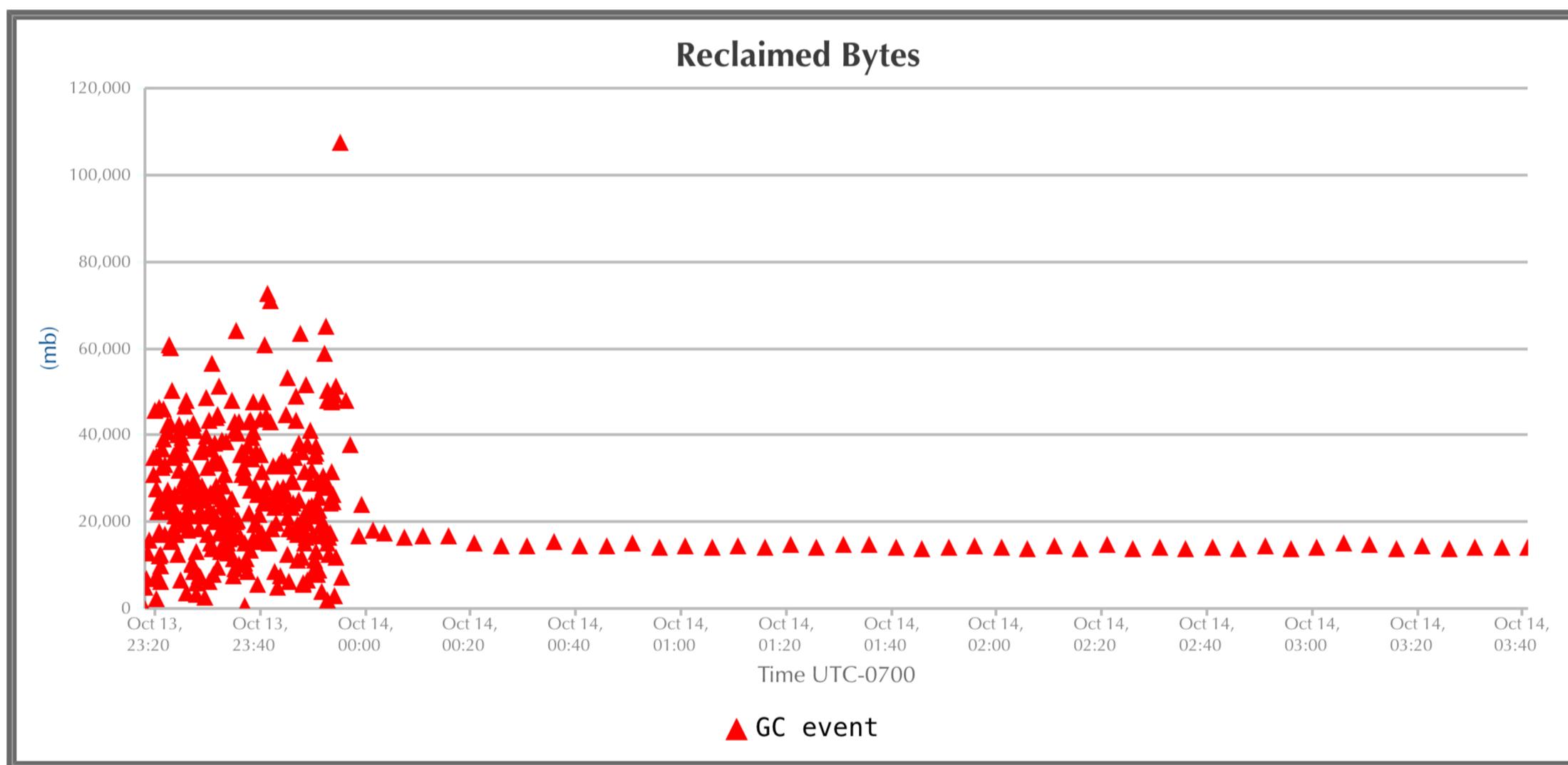
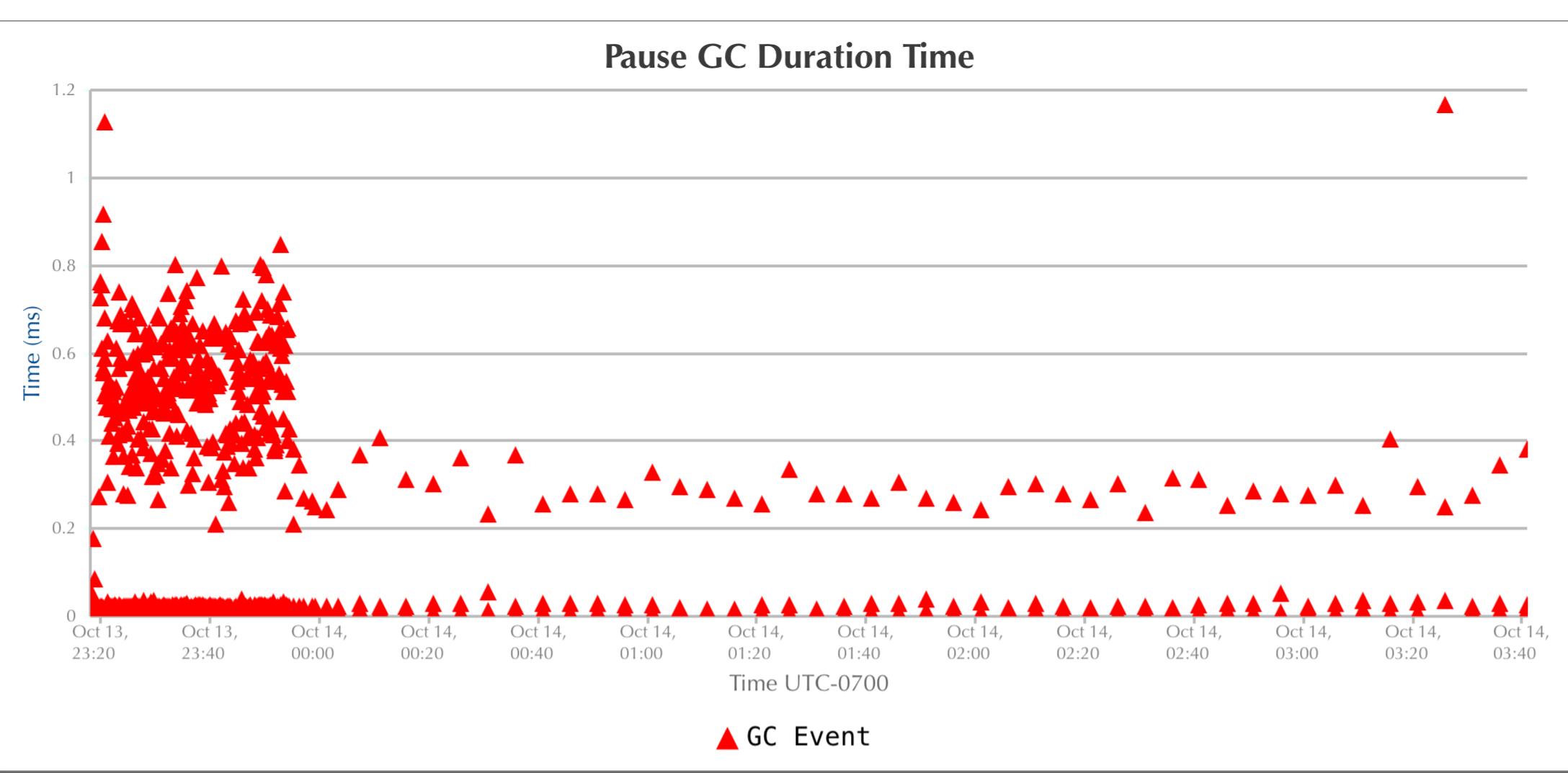
Select time range ?

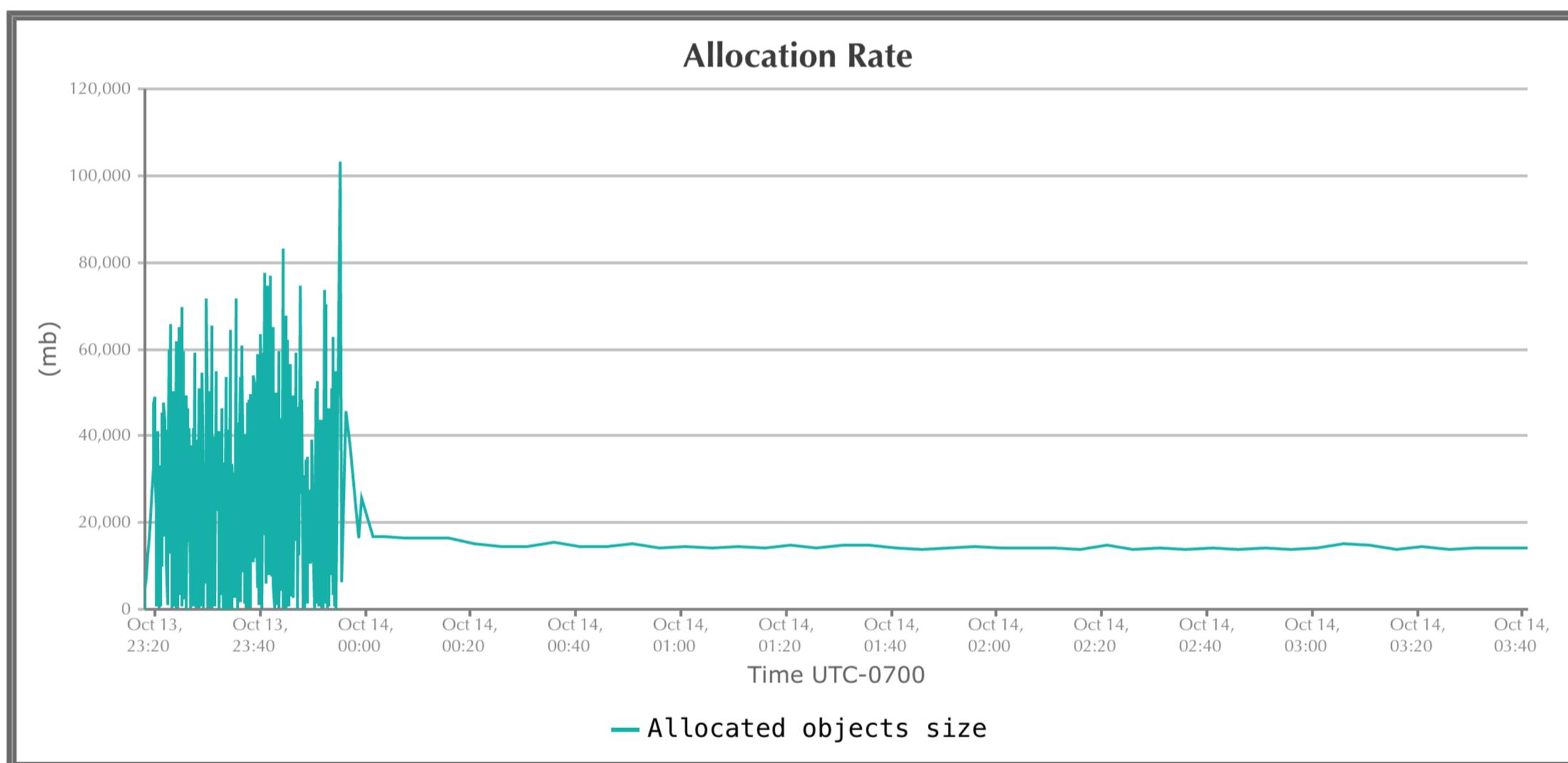
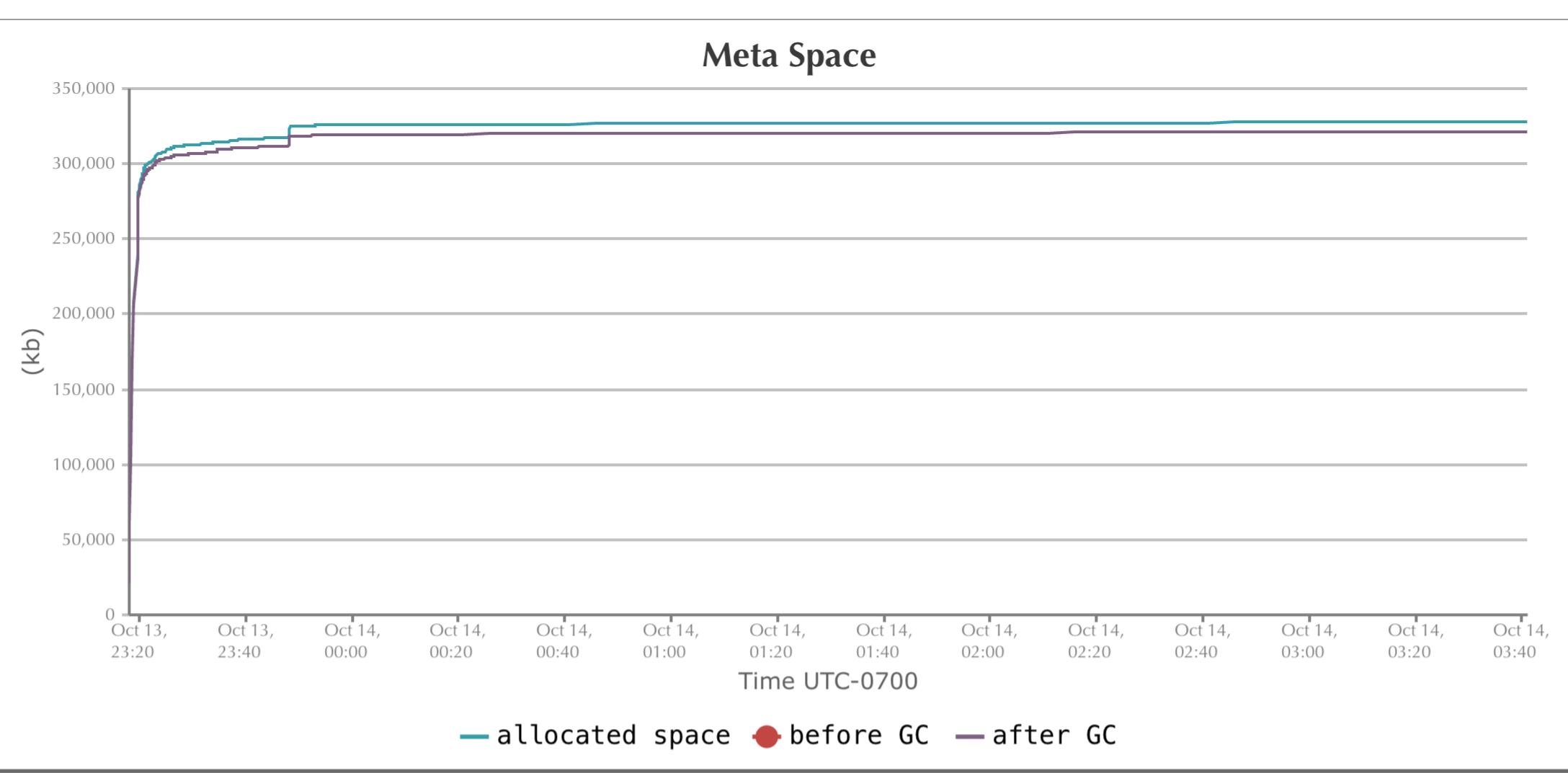
Reset

... Interactive Graphs (How to zoom graphs?)



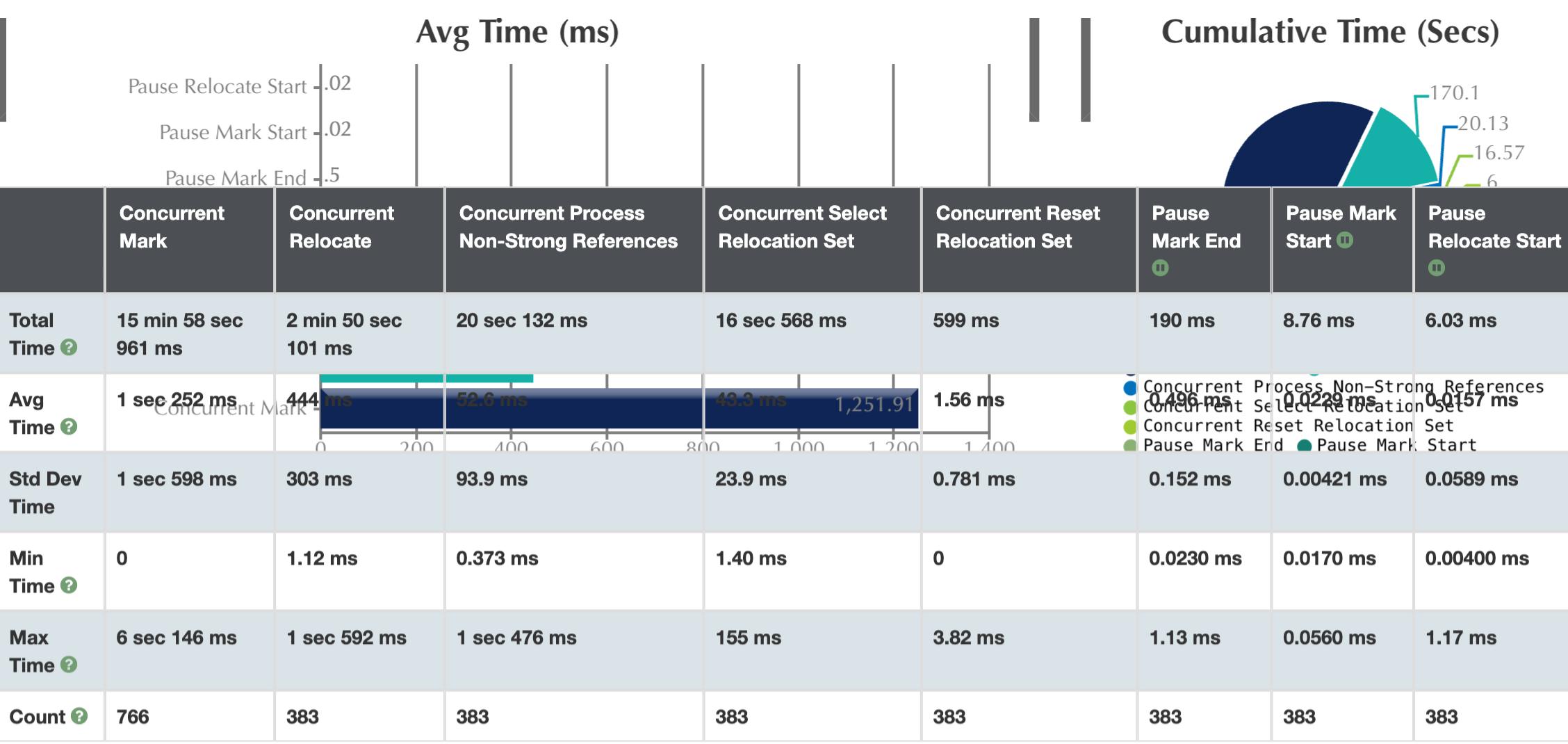




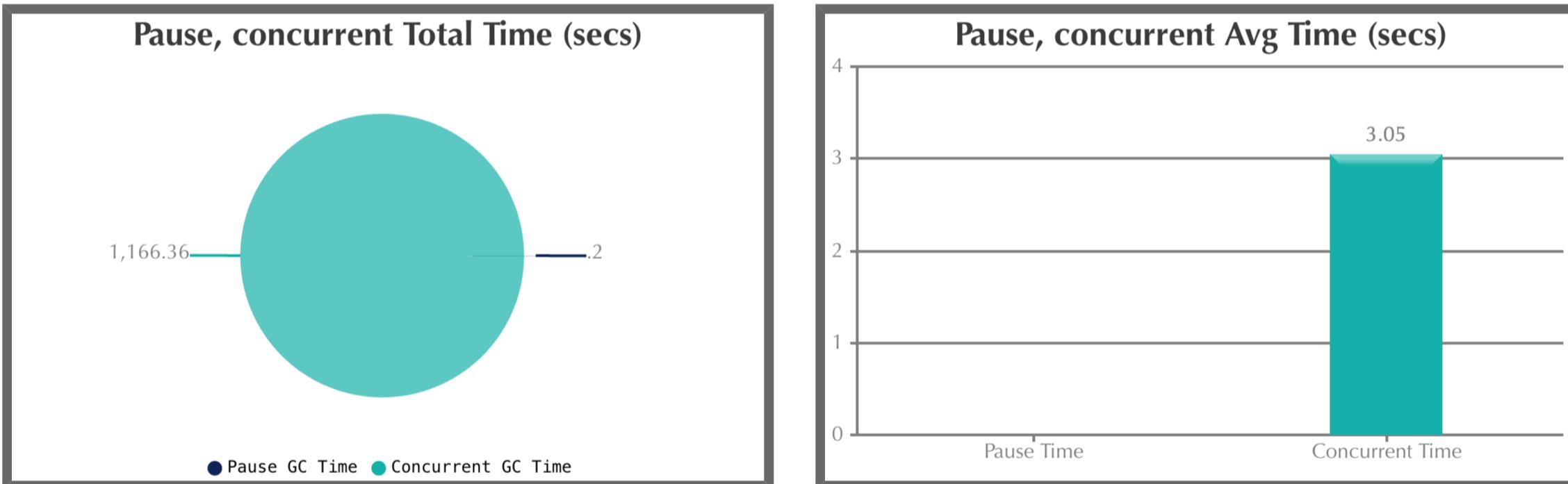


⌚ Z Phases Statistics





⌚ Z GC Time



Pause Time

Total GC Pause Time	205 ms
GC Pause Events	1149.0
Avg GC Pause Time	0.178 ms
Std Dev GC Pause Time	0.244 ms
Min GC Pause Time	0.00400 ms
Max GC Pause Time	1.17 ms

Concurrent Time

Total Concurrent GC Time	19 min 26 sec 362 ms
Concurrent GC Events	383.0
Avg Concurrent GC Time	3 sec 45 ms
Std Dev Concurrent GC Time	1 sec 715 ms
Min Concurrent GC Time	6.64 ms
Max Concurrent GC Time	7 sec 861 ms

⚙ Object Stats

Total created bytes	8.68 tb
---------------------	---------

CPU Stats

CPU Time:	n/a
-----------	-----

(To learn more about CPU stats, [click here](#))

Total promoted bytes ?	n/a	User Time: ?	n/a
Avg creation rate ?	573.81 mb/sec	Sys Time: ?	n/a
Avg promotion rate ?	n/a		

💧 Memory Leak [?](#)

No major memory leaks.

(Note: there are [8 flavours of OutOfMemoryErrors](#). 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](#))

	Total Time	Avg Time	% of total duration
Total time for which app threads were stopped	2.768 secs	0.001 secs	0.017 %
Time taken to stop app threads	7.325 secs	0.002 secs	0.046 %

📦 Threads Affected By Allocation Stalls [?](#)

(To learn more about Allocation Stall, [click here](#))

Not Reported in the log.

📄 String Deduplication Metrics [?](#)

Not Reported in the log.

❓ GC Causes [?](#)

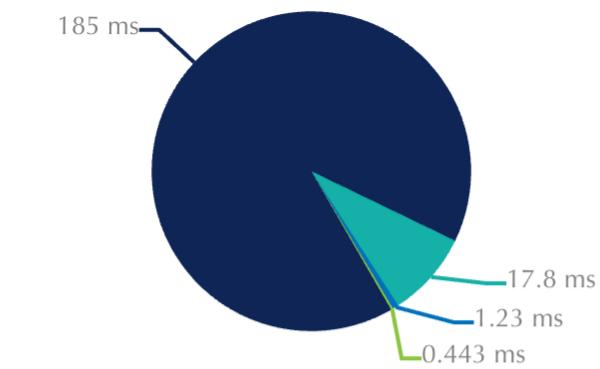
(What events caused GCs & how much time they consumed?)

Cause	Count	Avg Time	Max Time	Total Time
Allocation Rate ?	325	0.570 ms	1.16 ms	185 ms
Proactive ?	50	0.356 ms	1.45 ms	17.8 ms



Warmup	3	0.409 ms	0.797 ms	1.23 ms
Metadata GC Threshold ?	5	0.0886 ms	0.217 ms	0.443 ms

GC Causes (Total Time)



⌚ Tenuring Summary ?

Not reported in the log.

📄 JVM Arguments ?

(To learn about JVM Arguments, [click here](#))

Not reported in the log.

📀 Export Data ?

[Normalized GC Data](#)