Analysis Report

Consecutive Full GC

Our analysis tells that Full GCs are consecutively running in your application. It might cause intermittent OutOfMemoryErrors or degradation in response time or high CPU consumption or even make application unresponsive.

Read our recommendations to resolve consecutive Full GCs

JVM Heap Size

Allocated Size: : 512 mb

Peak Size: : 511 mb

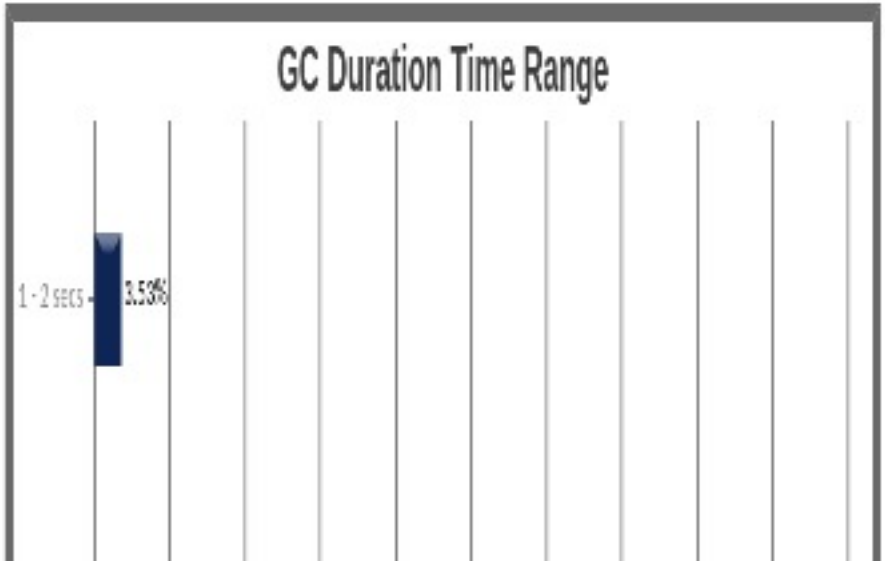
Key Performance Indicators

Important section of the report. To learn more about KPIs, click here!

1 Throughput: : 95.468%

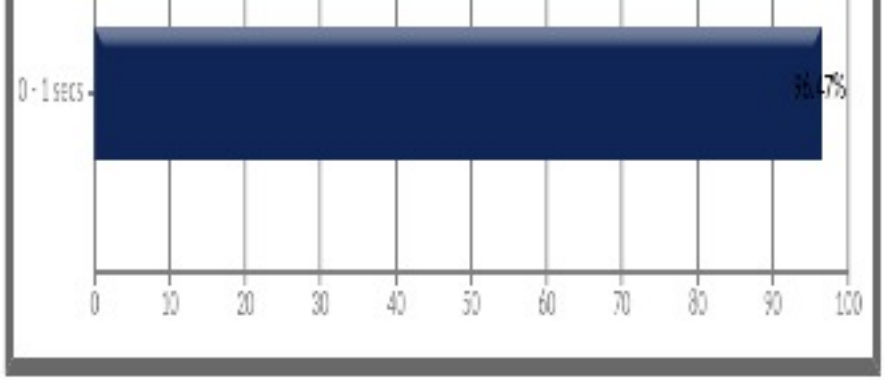
2 Latency:

Avg Pause GC Time	159 ms
Max Pause GC Time	1 sec 92 ms



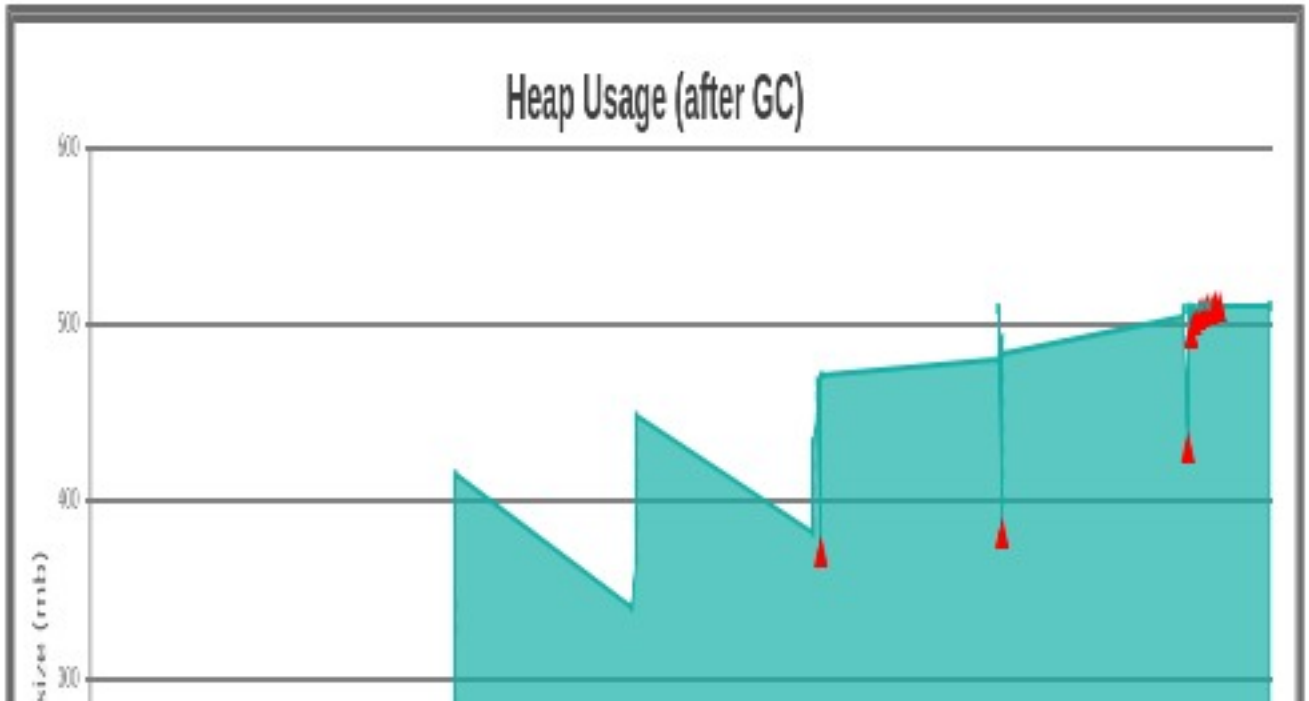
GC Pause Duration Time Range

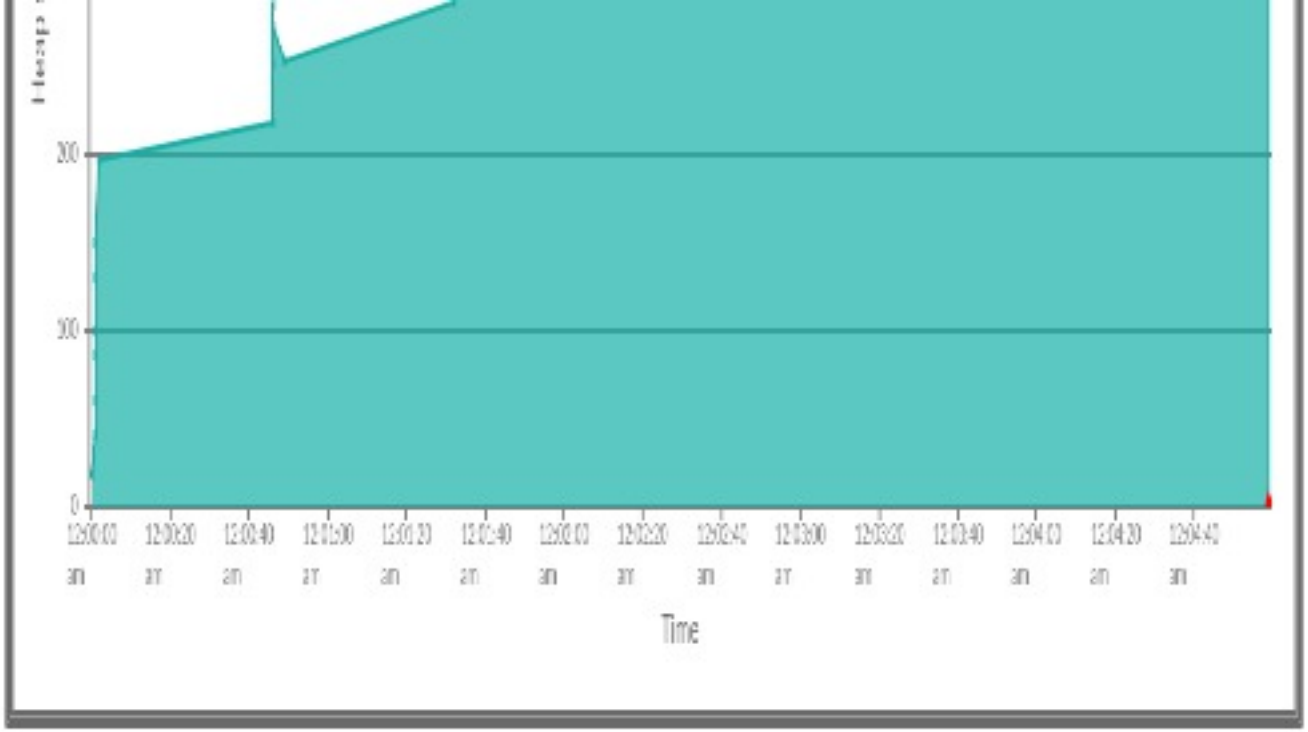
Duration (secs)	No. of GCs	Percentage
0 - 1	82	96.47%
1 - 2	3	100.0%

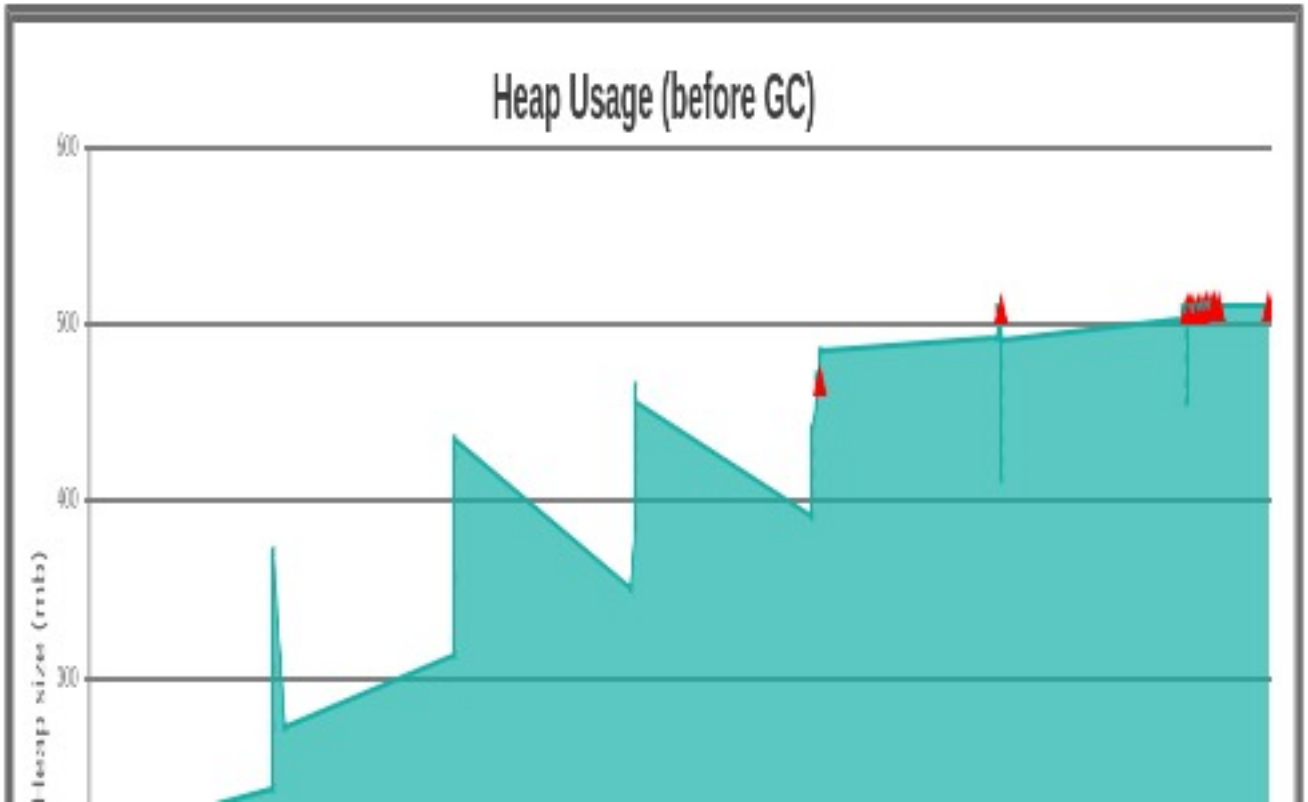


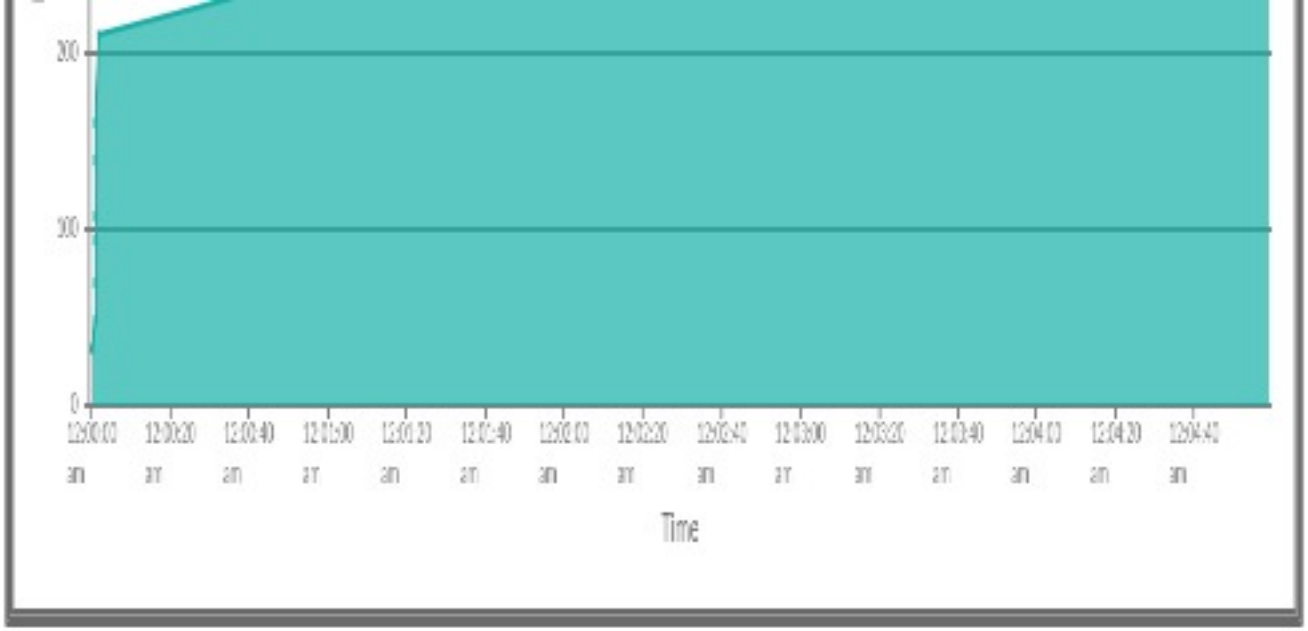
Interactive Graphs

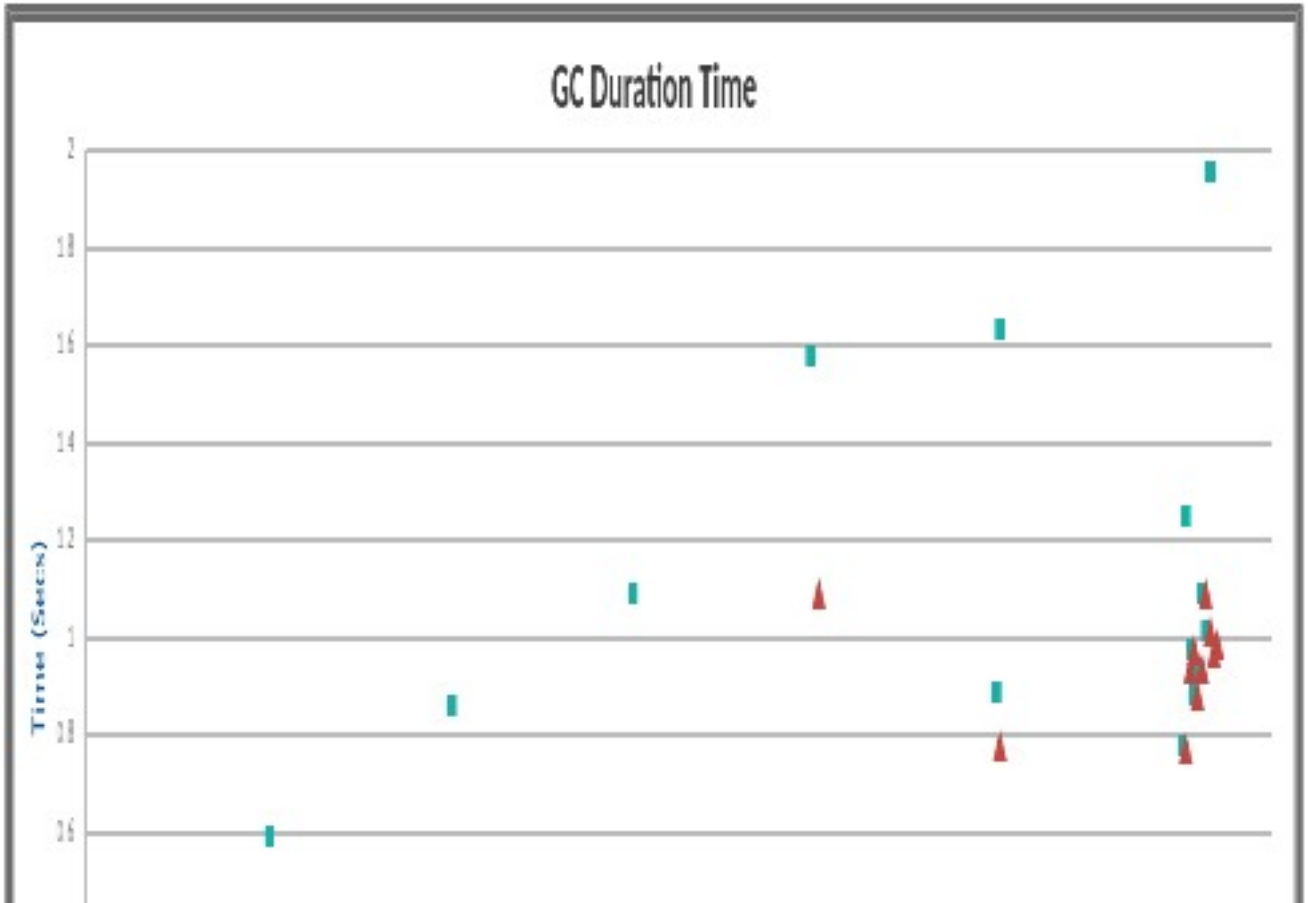
(All graphs are zoomable)

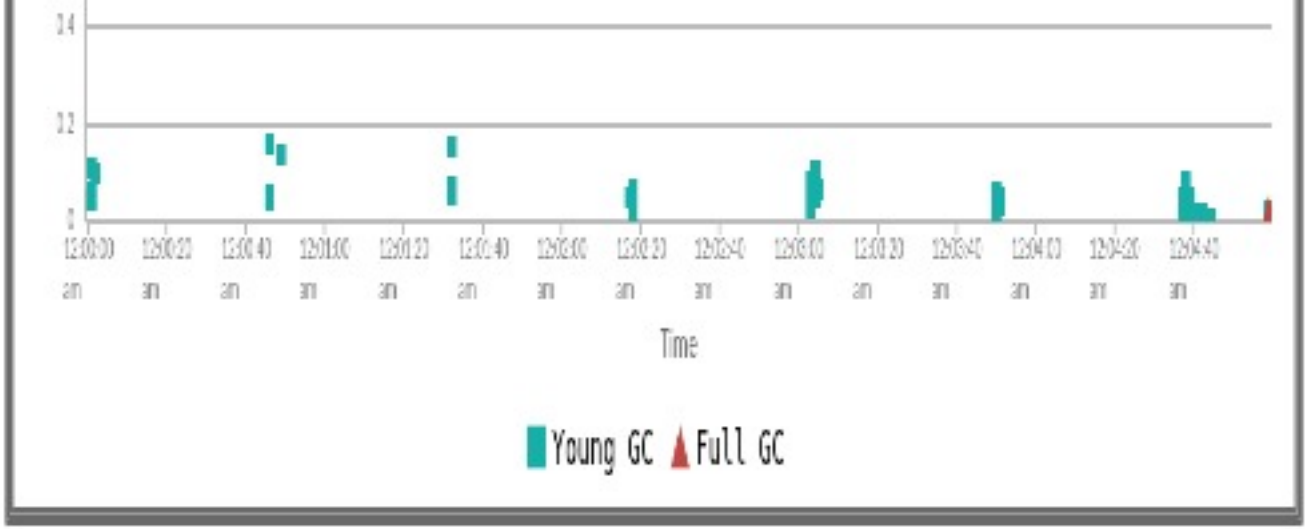


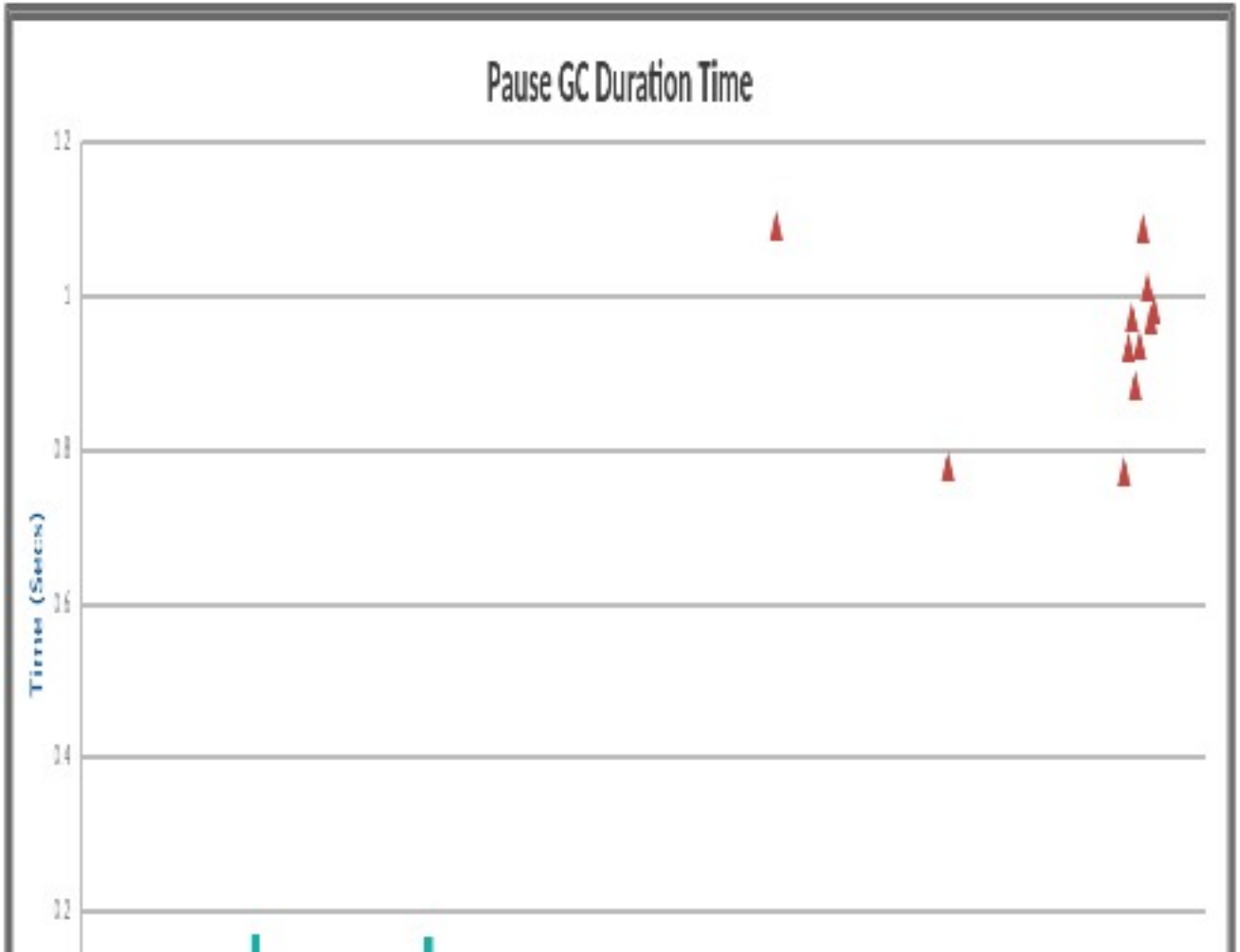

200
100
0
Time

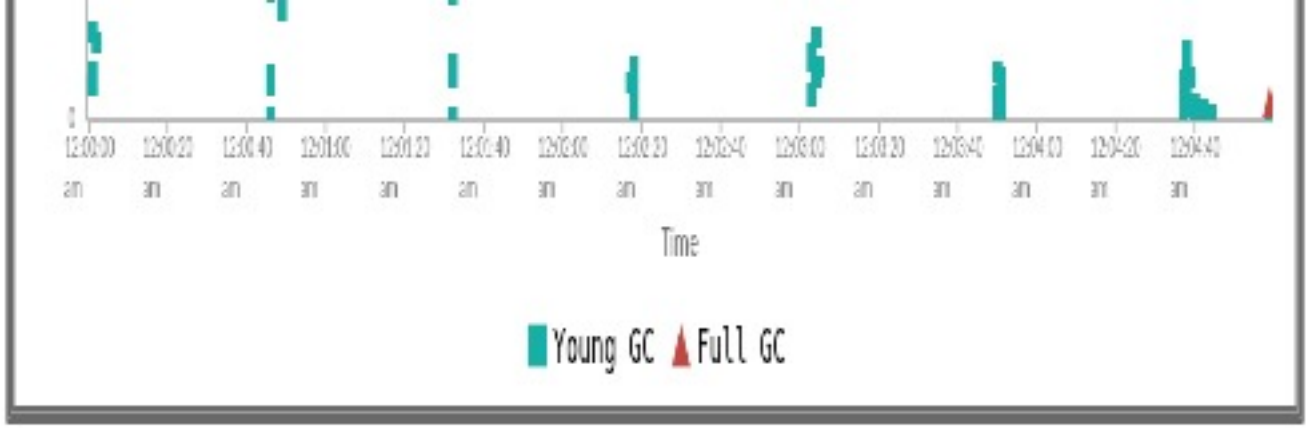

Heap Usage (before GC)
900
500
400
300

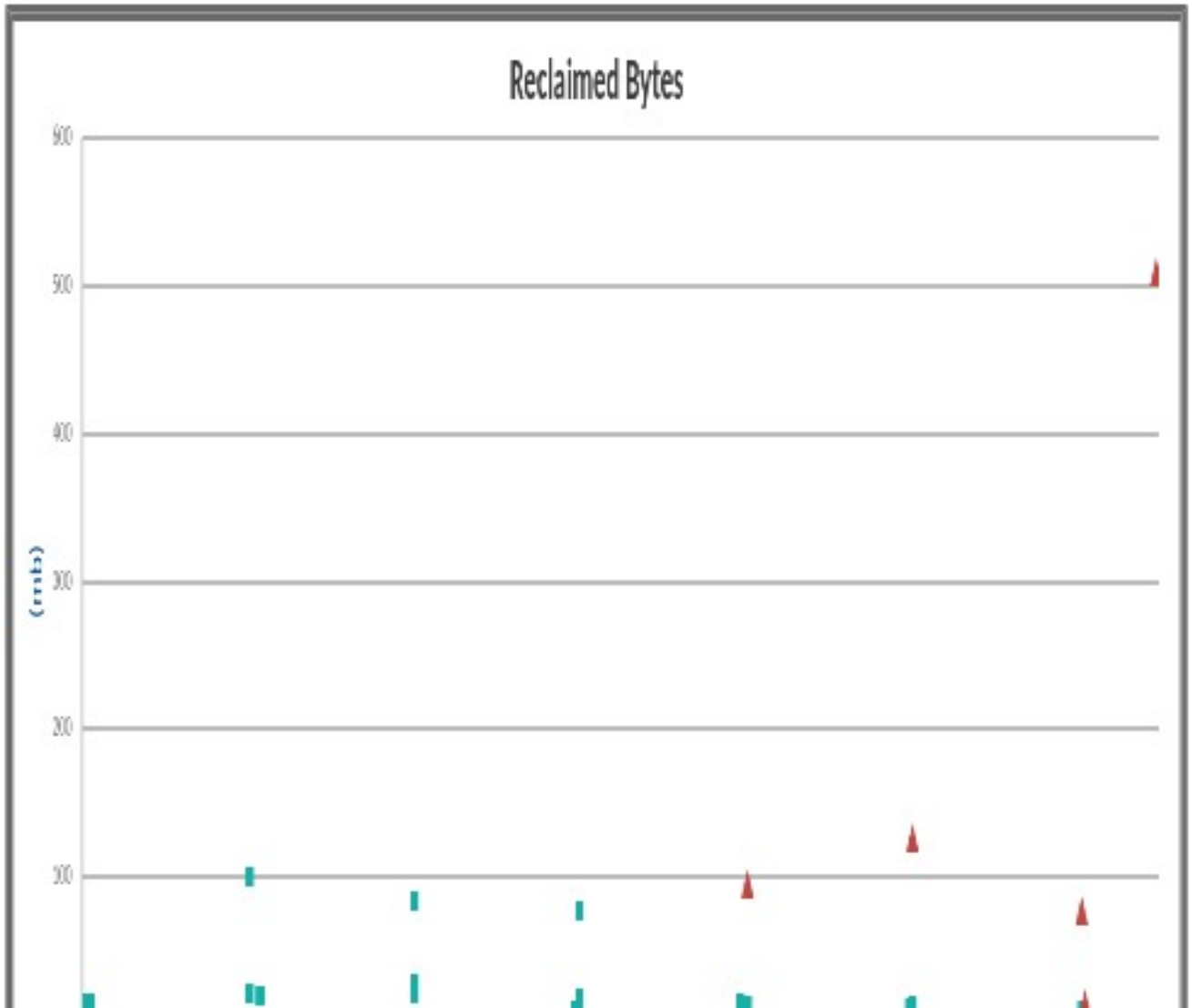

200
100
0
Time


GC Duration Time
2
1.8
1.6
1.4
1.2
1
0.8
0.6
Time (Secs)

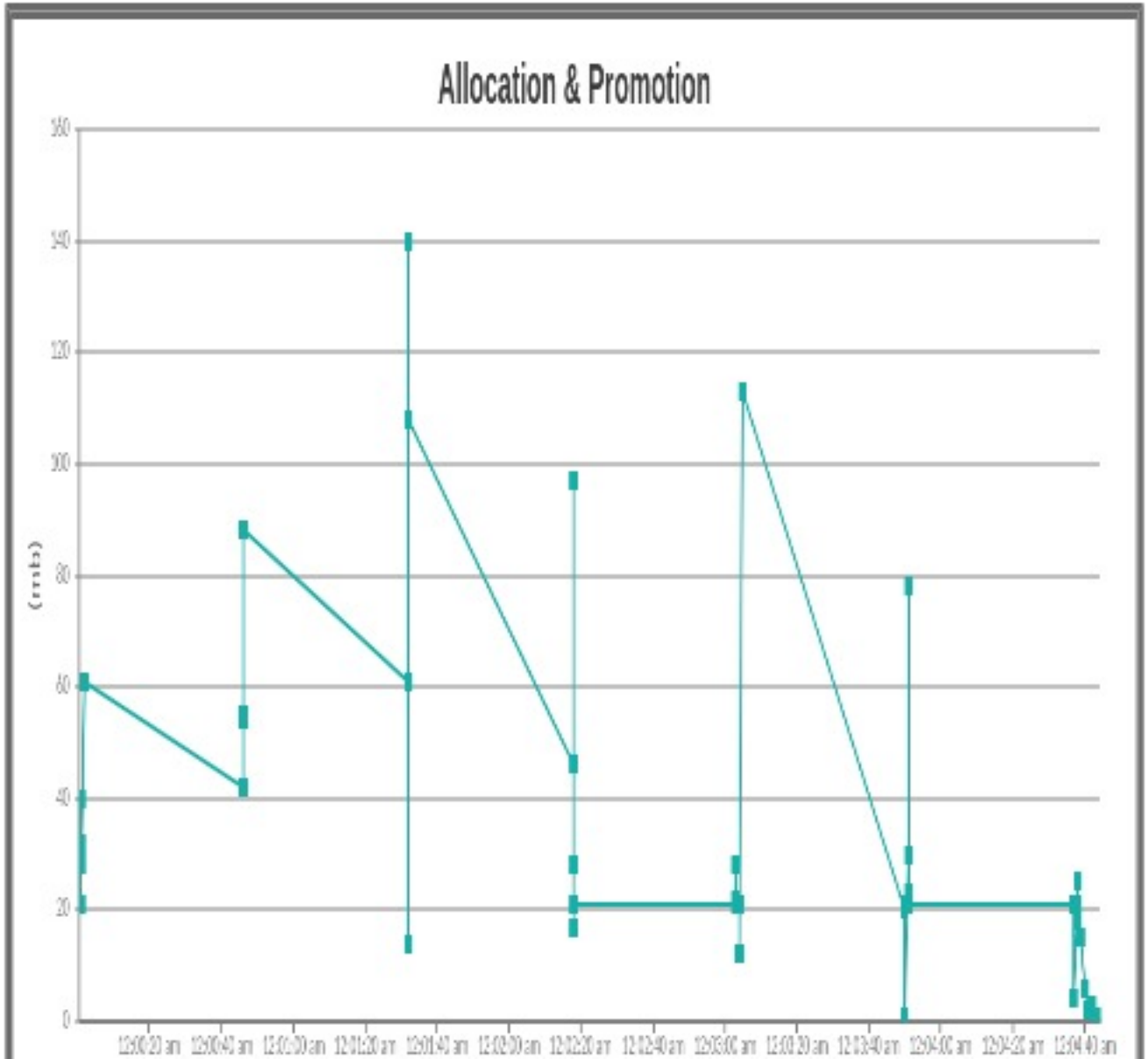

0.4
0.2
0
Time
Young GC
Full GC


Pause GC Duration Time
1.2
1
0.8
0.6
0.4
0.2
Time (Secs)

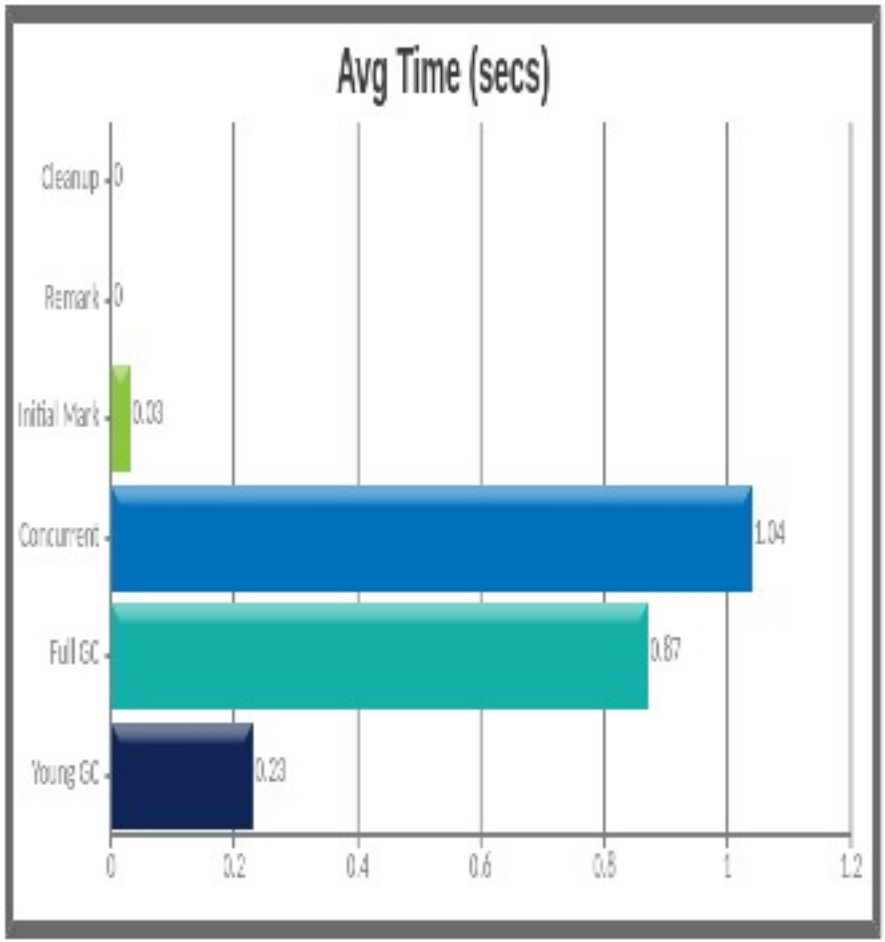

Time
Young GC
Full GC

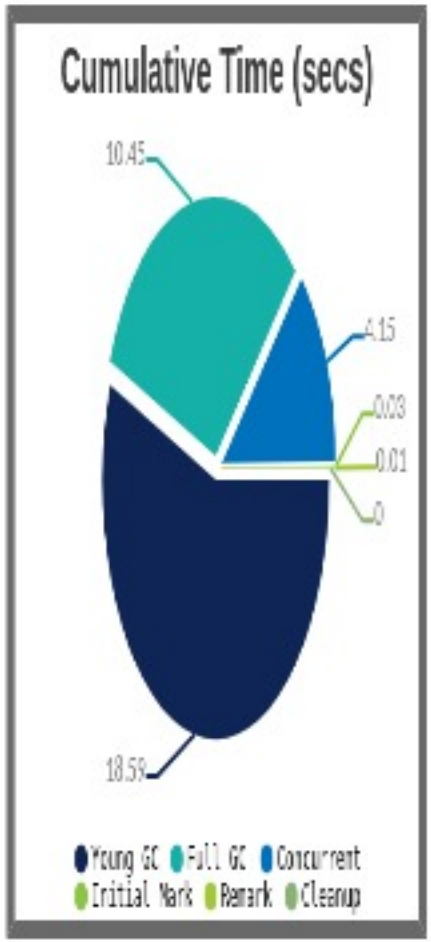

Reclaimed Bytes
(mb)
900
500
400
300
200
100


Time
Young GC
Full GC


Allocation & Promotion
(mb)
160
140
120
100
80
60
40
20
0

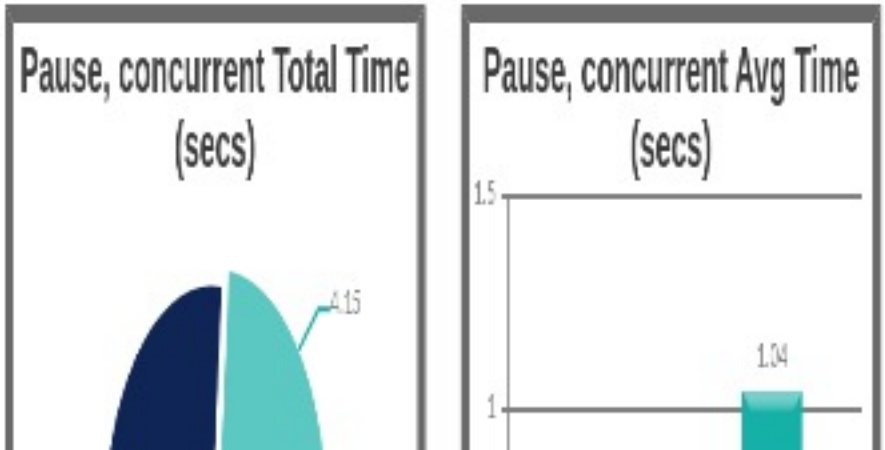
G1 Collection Phases Statistics

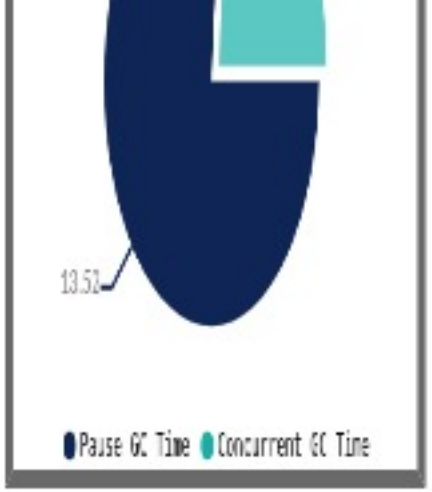


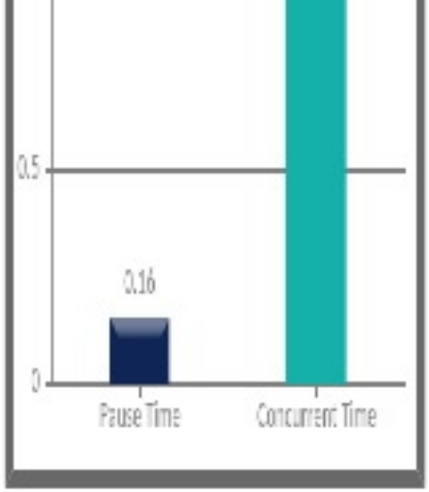


	Young GC	Full GC	Concurrent	Initial Mark	Remark	Cleanup
Total Time	18 sec 588 ms	10 sec 450 ms	4 sec 155 ms	27 ms	13 ms	3 ms
Avg Time	232 ms	871 ms	1 sec 39 ms	27 ms	3 ms	1 ms
Std Dev Time	432 ms	273 ms	382 ms	0	1 ms	0
Min Time	3 ms	22 ms	588 ms	27 ms	3 ms	1 ms
Max Time	1 sec 955 ms	1 sec 92 ms	1 sec 625 ms	27 ms	4 ms	1 ms
Count	80	12	4	1	4	4

G1 GC Time







Pause Time

Total Time	13 sec 515 ms
Avg Time	159 ms
Std Dev Time	308 ms
Min Time	1 ms
Max Time	1 sec 92 ms

Concurrent Time

Total Time	4 sec 155 ms
Avg Time	1 sec 39 ms
Std Dev Time	382 ms
Min Time	588 ms
Max Time	1 sec 625 ms

Object Stats

(These are perfect micro-metrics to include in your performance reports)

Total created bytes	1.72 gb
Total promoted bytes	n/a
Avg creation rate	5.91 mb/sec
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.)

Long Pause

None.

Safe Point Duration

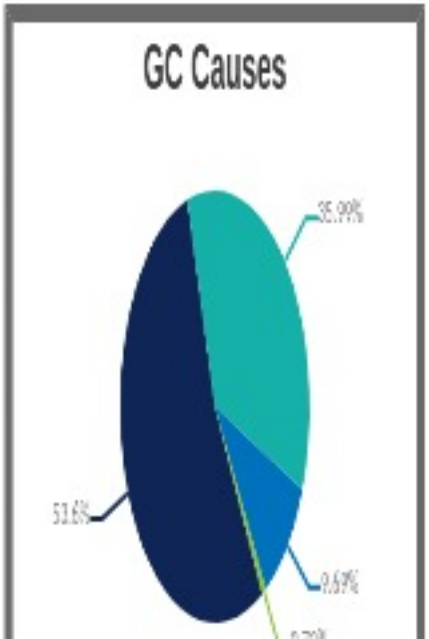
(To learn more about SafePoint duration, click here)

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 %
Others	16	n/a	n/a	15 sec 566 ms	53.61%
Full GC - Allocation Failure	12	871 ms	1 sec 92 ms	10 sec 450 ms	35.99%
G1 Evacuation Pause	62	45 ms	154 ms	2 sec 813 ms	9.69%
G1 Humongous Allocation	2	104 ms	157 ms	209 ms	0.72%
Total	92	n/a	n/a	29 sec 39 ms	100.01%



Tenuring Summary

Not reported in the log.

Command Line Flags

Not reported in the log.