G1GC

How Things are Connected

Jenny Zhang
JVM Platform, Oracle
April, 2017

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

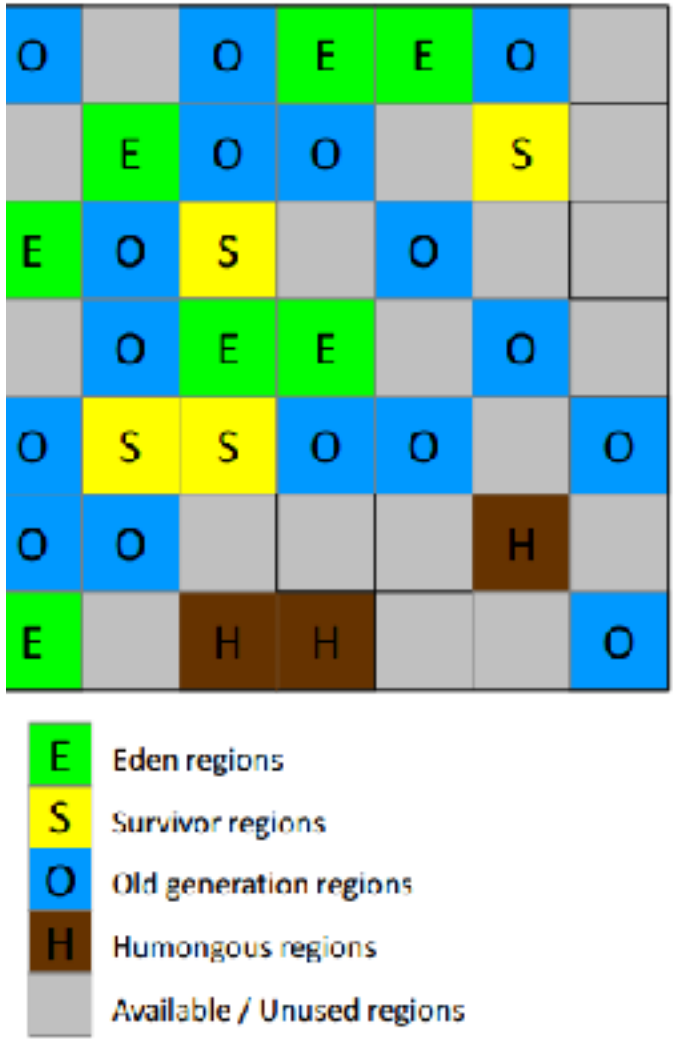
Program Agenda

1. Concepts to Understand G1GC
2. GC Logs: How to Print GC logs
3. G1GC: Analysis and Tuning
4. Other Considerations
5. Q/A

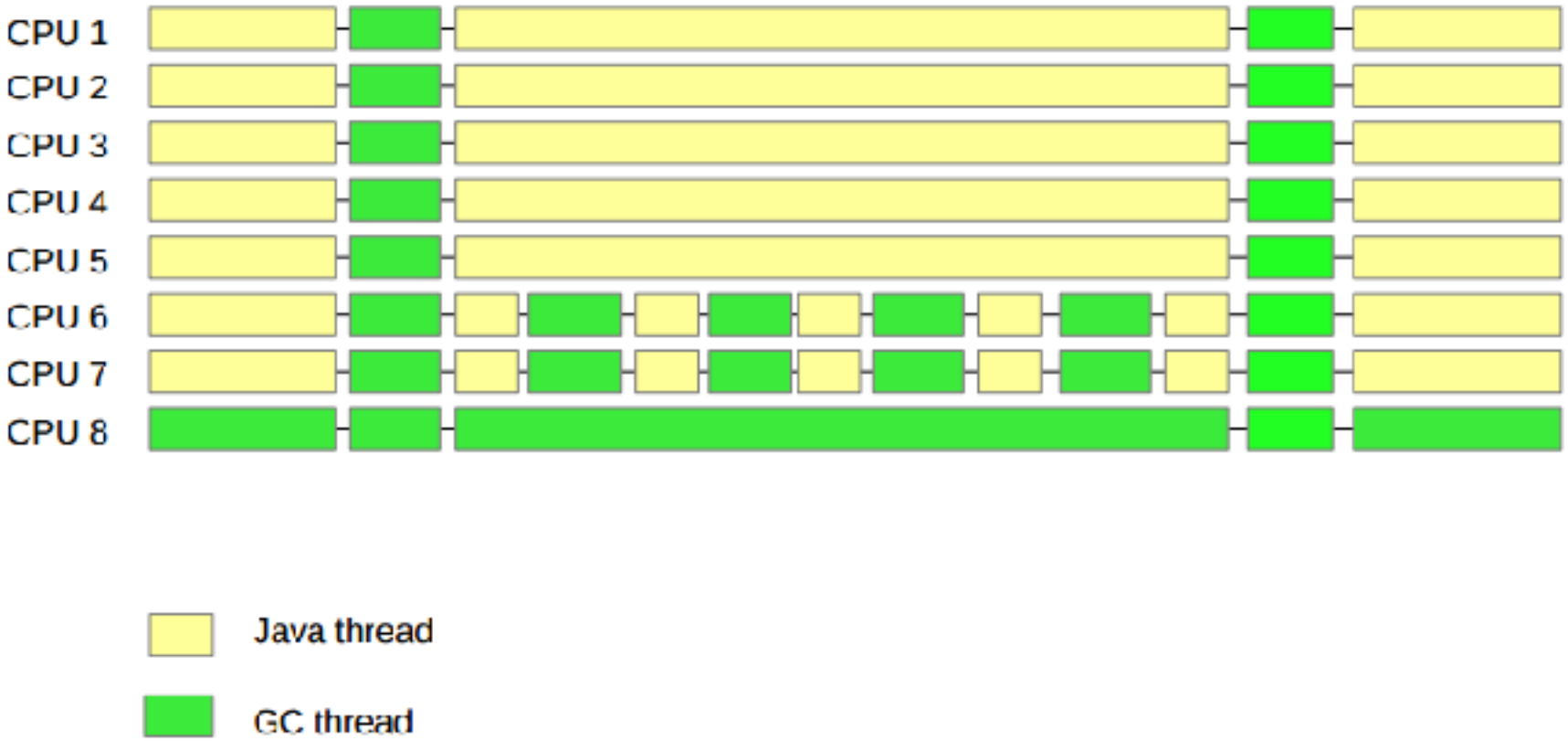
Program Agenda

1. Concepts to Understand G1GC
2. GC Logs: How to Print GC logs
3. G1GC: Analysis and Tuning
4. Other Considerations
5. Q/A

Regions



Concurrent Generational GC



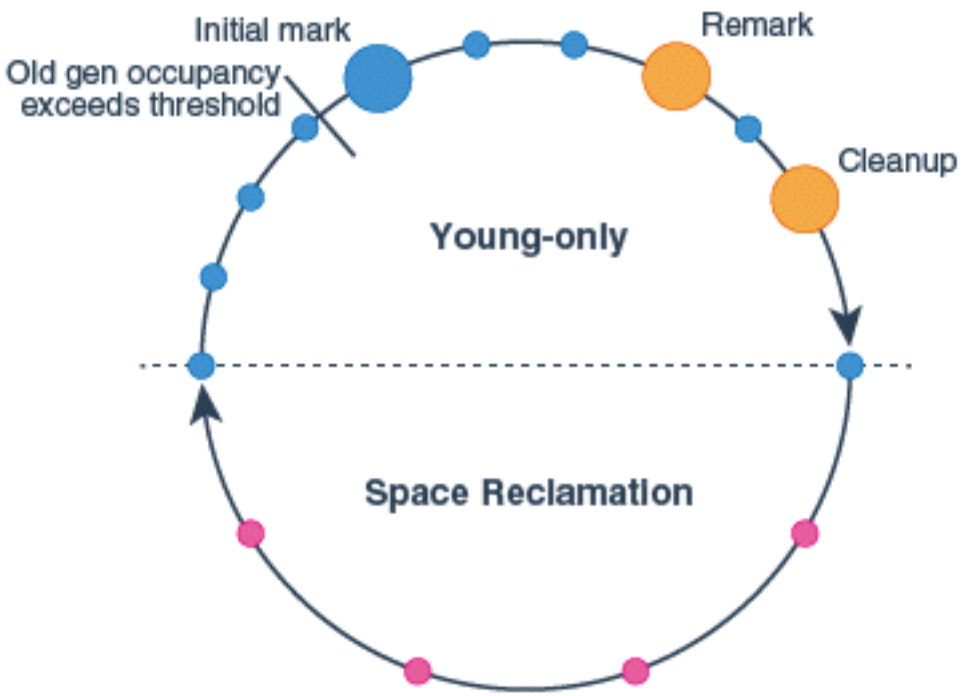
G1GC Concepts: GC Threads (may change)

Name	Parameters	Function	Default
ParallelGCThreads	-XX:ParallelGCThreads	Threads for parallel work during a gc pause	# of cpus (up to 8) 8+(#processors-8)(5/8)
Parallel Marking Threads	-XX:ConcGCThreads	Threads for concurrent marking	Max((parallelGCThreads+2)/4,1)
G1 Main Concurrent Mark Thread		Controller of concurrent marking work	1
G1 Concurrent Refinement Threads	-XX:G1ConcRefinementThreads	Update Remember Set concurrently with the application	Max: parallelgGCThreads+1

G1GC Concepts

- Collection Set(CSet)
- Card Table
- Remember Set
  - Track outside references that point into a heap region
  - One per heap region
- SATB
  - Logical snapshot of the heap, live objects at the beginning of the marking cycle
- Humongous Objects
  - Object Size > 1/2 of Region Size
  - Allocated to old gen
  - Primitive type can be collected during young gc.
  - Other type collected at the end of marking, or Full GC
- Dynamic IHOP(Initial Heap Occupancy Percentage)

G1 Collection Cycle



Program Agenda

1. Concepts to Understand G1GC
2. GC Logs: How to Print GC logs
3. G1GC: Analysis and Tuning
4. Other Considerations
5. Q/A

JDK8	JDK9	Description
-XX:+PrintGC -Xloggc:<filename>	-Xlog:gc:<filename>	Log messages tagged with ‘gc’ using ‘info’ level to <filename>, with default decorations.
-XX:+PrintGCDetails	-Xlog:gc*=info	In addition to single line summary, prints more details, like phases, heap size, cpu usage
-XX: +PrintAdaptiveSizePolicy	-Xlog:gc=info,gc+ergo*=debug	Print out the information about how G1 ergonomic decision is make
-XX:PrintReferenceGC	-Xlog:gc=info,gc+ref=debug	Print out the information about reference processing: number of references, processing time, etc
-XX:+PrintFlagsFinal	-Xlog:help	commands to get help on gc log flags/tags

GC Log Format

- Format:

```
[timestamp][level][tag] GC(id) Event
[0.395s][info ][gc,phases      ] GC(2)   Pre Evacuate Collection Set: 0.0ms
```

- Event(Reason)
  - Pause Young (G1 Evacuation Pause)
  - Pause Full (System.gc())
  - Pause Initial Mark (G1 Humongous Allocation)
  - Pause Mixed (G1 Evacuation Pause)
  - Pause Remark
  - Pause Cleanup
  - Concurrent …

Program Agenda

1. Concepts to Understand G1GC
2. GC Logs: How to Print GC logs
3. G1GC: Analysis and Tuning
4. Other Considerations
5. Q/A

Behavior-Based Tuning

- Maximum Pause-Time Goal
  - -XX:MaxGCPauseMillis=<200>
- Throughput Goal
  - -XX:GCTimeRatio=<12>
- Try to avoid using -Xmn, -XX:NewRatio
  - -XX:G1NewSizePercent=<5>
  - -XX:G1MaxNewSizePercent=<60>

Case Study: Dynamic IHOP and Humongous Objects

```
[120.265s][debug][gc,ergo        ] GC(67) Initiate concurrent cycle (concurrent
cycle initiation requested)
[120.265s][info ][gc,start       ] GC(67) Pause Initial Mark (G1 Humongous
Allocation)
[120.265s][info ][gc,task        ] GC(67) Using 18 workers of 18 for evacuation
[120.266s][debug][gc,ergo        ] GC(67) Running G1 Clear Card Table Task using
1 workers for 1 units of work for 11 regions.
[120.266s][debug][gc,ergo        ] GC(67) Running G1 Free Collection Set using 1
workers for collection set length 11
[120.267s][info ][gc,phases      ] GC(67)   Pre Evacuate Collection Set: 0.1ms
[120.267s][debug][gc,phases      ] GC(67)     Choose Collection Set: 0.0ms
[120.267s][debug][gc,phases      ] GC(67)     Humongous Register: 0.1ms
```

Case Study: Dynamic IHOP and Humongous Objects

```
[120.267s][info ][gc,phases      ] GC(67)   Evacuate Collection Set: 0.8ms
[120.267s][debug][gc,phases      ] GC(67)     Ext Root Scanning (ms):   Min:  0.2, Avg:  0.2, Max:
0.3, Diff:  0.1, Sum:  4.0, Workers: 18
[120.267s][debug][gc,phases      ] GC(67)     Update RS (ms):           Min:  0.0, Avg:  0.0, Max:
0.0, Diff:  0.0, Sum:  0.1, Workers: 18
[120.267s][debug][gc,phases      ] GC(67)       Processed Buffers:        Min: 0, Avg:  0.6, Max: 3,
Diff: 3, Sum: 10, Workers: 18
[120.267s][debug][gc,phases      ] GC(67)     Scan RS (ms):             Min:  0.0, Avg:  0.0, Max:
0.0, Diff:  0.0, Sum:  0.0, Workers: 18
[120.267s][debug][gc,phases      ] GC(67)     Code Root Scanning (ms):  Min:  0.0, Avg:  0.0, Max:
0.0, Diff:  0.0, Sum:  0.0, Workers: 18
[120.267s][debug][gc,phases      ] GC(67)     AOT Root Scanning (ms):   skipped
[120.267s][debug][gc,phases      ] GC(67)     Object Copy (ms):         Min:  0.1, Avg:  0.2, Max:
0.4, Diff:  0.4, Sum:  3.0, Workers: 18
[120.267s][debug][gc,phases      ] GC(67)     Termination (ms):         Min:  0.0, Avg:  0.3, Max:
0.4, Diff:  0.4, Sum:  5.1, Workers: 18
[120.267s][debug][gc,phases      ] GC(67)       Termination Attempts:     Min: 1, Avg:  1.4, Max: 3,
Diff: 2, Sum: 25, Workers: 18
[120.267s][debug][gc,phases      ] GC(67)     GC Worker Other (ms):     Min:  0.0, Avg:  0.0, Max:
0.0, Diff:  0.0, Sum:  0.2, Workers: 18
[120.267s][debug][gc,phases      ] GC(67)     GC Worker Total (ms):     Min:  0.7, Avg:  0.7, Max:
0.7, Diff:  0.1, Sum: 12.4, Workers: 18
```

Case Study: Dynamic IHOP and Humongous Objects

```
[120.267s][info ][gc,phases      ] GC(67)   Post Evacuate Collection Set: 0.7ms
[120.267s][debug][gc,phases      ] GC(67)     Code Roots Fixup: 0.0ms
[120.267s][debug][gc,phases      ] GC(67)     Preserve CM Refs: 0.0ms
[120.267s][debug][gc,phases      ] GC(67)     Reference Processing: 0.0ms
[120.267s][debug][gc,phases      ] GC(67)     Clear Card Table: 0.0ms
[120.267s][debug][gc,phases      ] GC(67)     Reference Enqueuing: 0.0ms
[120.267s][debug][gc,phases      ] GC(67)     Merge Per-Thread State: 0.0ms
[120.267s][debug][gc,phases      ] GC(67)     Code Roots Purge: 0.0ms
[120.267s][debug][gc,phases      ] GC(67)     Redirty Cards: 0.0ms
[120.267s][debug][gc,phases      ] GC(67)     Clear Claimed Marks: 0.0ms
[120.267s][debug][gc,phases      ] GC(67)     Free Collection Set: 0.0ms
[120.267s][debug][gc,phases      ] GC(67)     Humongous Reclaim: 0.6ms
[120.267s][debug][gc,phases      ] GC(67)     Expand Heap After Collection: 0.0ms
[120.267s][info ][gc,phases      ] GC(67)   Other: 0.4ms
```

Case Study: Dynamic IHOP and Humongous Objects

```
120.267s][info ][gc,heap          ] GC(67) Eden regions: 7->0(275)
[120.267s][info ][gc,heap          ] GC(67) Survivor regions: 4->2(14)
[120.267s][info ][gc,heap          ] GC(67) Old regions: 13->13
[120.267s][info ][gc,heap          ] GC(67) Humongous regions: 655->291
[120.267s][info ][gc,metaspace     ] GC(67) Metaspace: 7576K->7576K(1056768K)
[120.267s][info ][gc               ] GC(67) Pause Initial Mark (G1 Humongous
Allocation) 1355M->610M(1448M) 2.479ms
[120.267s][info ][gc,cpu           ] GC(67) User=0.02s Sys=0.00s Real=0.00s
```

Issues: Demo

G1GC Tuning Parameters

- -XX:-G1UseAdaptiveIHOP -XX:InitiatingHeapOccupancyPercent=90
- -XX:G1HeapRegionSize=32m

Tuning	Score	Number of Marking Cycles
	68.4	62
• -XX:-G1UseAdaptiveIHOP -XX:InitiatingHeapOccupancyPercent=90	72.68	26
• -XX:G1HeapRegionSize=32m	76.09	8

Issues: Demo

- Termination Time - ParallelGCThreads
- Remember Set Processing - G1HeapRegionSize
- Expensive Old Regions for Mixed GC - G1MixedGCCountTarget
- Old Region Allocation and Reclamation - G1UseAdaptiveIHOP and InitiatingHeapOccupancyPercent

Other G1GC Tuning Parameters (gc+ergo*=debug)

Parameter	Description
-XX:G1HeapWastePercent=<5>	Allows this percent of heap to be wasted. G1 uses this to decide when to start/stop mixed gc
-XX:G1MixedGCCountTarget=<8>	The expected number of mixed gc to reclaim the candidate old regions for this marking cycle. Increasing this number could results in more mixed gas, but with shorter pause time for each mixed gc
-XX:G1MixedGCLiveThresholdPercent=85	If the live object occupancy in the old regions is higher than this threshold, g1 will not consider it as a candidate for collection set.

Other G1GC Tuning Parameters (gc+remset=trace -XX:G1SummarizeRSetStatsPeriod=<n>)

Parameter	Description
-XX:G1RSetRegionsEntries=<ergo>	Max number of entries in the fine table
-XX:G1ConcRefinmentThreads=<n>	number of threads that updates the remember set

Other G1GC Tuning Parameters

Parameter	Description
-XX:-ParallelRefProcEnabled	Determines if the processing of java.lang.Ref.* is done in parallel by multiple threads
-XX:+UseLargePages	

Program Agenda

1. Concepts to Understand G1GC
2. GC Logs: How to Print GC logs
3. G1GC: Analysis and Tuning
4. Other Considerations
5. Q/A

Other Considerations

- Decide the optimal number of gc threads
  - On System with a lot of cpu threads, the default ParallelGCThreads might be too much
  - Long Termination Time
  - [3.547s][info][gc,cpu] GC(16) User=0.34s Sys=0.02s Real=0.02s
- SafePoint: PrintApplicationStoppedTime/safepoint=debug
- High System Time
  - Transparent Huge Page on Linux
  - Page Fault
  - GC logging too slow

Additional Info (1 of 2)

- http://docs.oracle.com/javase/9/gctuning
- http://performance-related.blogspot.com/2017/04/how-to-convert-jdk8-gc-log-flags-to.html
- http://performance-related.blogspot.com/2017/04/g1gc-performance-tuning-parameters.html

Additional Info (2 of 2)

- Java Performance book
  - http://www.informit.com/store/java-performance-9780137142521
  - http://techbus.safaribooksonline.com/book/programming/java/9780137001040 (PDF or e-Book — free to Oracle employees)
- Java Performance Companion book
  - http://www.informit.com/store/java-performance-companion-9780133796827
  - http://techbus.safaribooksonline.com/book/programming/java/9780133796896 (PDF or e-Book — free to Oracle employees)
- Java Performance Live Lessons (Video)
  - http://techbus.safaribooksonline.com/video/programming/java/9780133443561 (video training — free to Oracle employees)

Q/A