COMPARING GARBAGE COLLECTORS & ALLOCATION

PRESENTED BY YARDEN MARTON  
18.11.14

For a start, we saw that comparing between different GCs is basically a complicated task, and one's conclusion regarding his preferred GC may differ according to his application's characteristics and purpose.

A garbage collector has lots of qualities which varies between one another – Throughput, pause times, space utilization, implementation and more. A collector's performance is also affected by the environment it runs on – heap size, amount of available heap space during its run and application dependency.

We saw a graph that shows how important are low pause times – how each processor's throughput slows the more processors on a system with a stop-the-world collector.

To summarize this sub-chapter, we discussed how can a developer choose a collector wisely – measure its application's behavior, track and define the objects it uses and experiment with the collectors on offer and their many configurations.

Next, we presented an abstract framework for a wide variety of collectors.

This method describes high-level garbage collecting algorithms, which can be implemented and maintained in many ways. Those algorithms highlight both similarity and difference between the collectors we learnt.

The second chapter talked about allocation technics and issues.

We presented the different allocation methods and their improvements – sequential allocation, free-list allocation (first-fit, next-fit and best-fit), segregated-fits allocation, and saw a practice and useful one like the buddy system.

We discussed how each method is different in matter of speed, efficiency, locality and fragmentation.

For conclusion we raised additional considerations regarding allocation – alignment, size constrains, boundary tags, and heap parsability.