Concurrent Mark Sweep

Based on "The Garbage Collection Handbook" – Chapter 16

By Richard Jones, Antony Hosking and Eliot Moss

Presented by Eyal Dushkin
30/12/14

Summary

The talk is based on concurrent mark sweep algorithms (CMS), which are elaborated within chapter 16 of the book. Following the previous lecture on the general concept of concurrency, this talk reveals for the first time concrete concurrent algorithms, both mostly-concurrent and on-the-fly. During the talk we deep dive into the structure of these algorithms, and emphasize their benefits and drawbacks.

The talk begins with a recap of the previous lesson, underlying specific matters which are directly related to CMS, such as the tricolor abstraction, garbage collector invariants, the lost objects problem and read and write barriers. Following that, we describe two CMS algorithms: mostly-concurrent mark-sweep and on-the-fly mark-sweep.

In mostly-concurrent mark-sweep, the mark phase is usually sub-divided into some variant in which in the initial marking, all mutator's threads are suspended and the gc root objects are marked as alive; During concurrent marking, the marked root objects are traversed concurrently with the mutator's threads. The termination could either requires a stop the world phase, or not – depends on the mutator's color. Examining on-the-fly mark-sweep algorithm, there's no stop the world phase, and hence a non-trivial albeit nice methodology is being introduced in order to sync between mutator's and collector's threads without suspending the application.

The last part sheds light on the general usage of these algorithms, and describes what happens when theory meets practice.

Original Ideas

1. I decided to elaborate on the relation between a black mutator and its barriers. A discussion in class has revealed the necessity of using read barriers, instead of write barriers, in order to preserve the strong invariant. A mutator's thread stack could traverse to white objects through a chain of reads from the root objects, which finally might be resulted in a white root, and thus using a read barrier is a must.[1]
2. An original explanation to Lamport concurrent mark and sweep algorithm (1976) is being introduced.[2]

[1] Slides 36,37,72
[2] Slides 60-68

3. I decided to focus on the practical side of CMS algorithms – I showed the usage of CMS in real-world, followed by metrics and benchmarks comparing between CMS to other gc algorithms. [3]

Discussion

The discussions in the classroom revolved mostly around the tradeoffs of CMS algorithms. The class consensus appointed that CMS algorithm minimize pause time, but at the expense of fragmentations, cpu's performance and the possibility of promotion failures. We then argued about the usage of CMS in applications, and concluded that it's a good idea to use CMS when there were large sets of living objects. The discussion was immediately followed by benchmarks and examples from real-world.

[3] Slides 99-104, http://blog.mgm-tp.com/2013/12/benchmarking-g1-and-other-java-7-garbage-collectors/