Master’s Thesis

Work Stealing in the Hotspot VM

Student:	Johannes Scheibl (00956055)
Advisor:	Prof. Mössenböck, DI Thomas Schatzl
Start date:	01.04.2019

o.Univ.-Prof. Dr. Dr.h.c.
Hanspeter Mössenböck
Institute for System Software

P +43 732 2468 4340
F +43 732 2468 4345
hanspeter.moessenboeck@jku.at

Secretary:
Birgit Kranzl
Ext. 4341
birgit.kranzl@jku.at

Linz, March 20, 2019

The Hotspot virtual machine (VM) is a core component of the OpenJDK project [1]. It provides automatic dynamic memory management for its hosted applications using different garbage collection algorithms (e.g. [2][3]).

Parallel execution of these garbage collection algorithms provides the necessary speedup for achieving good performance. Balancing the workloads among threads is vital to ensure optimal use of resources. The main technique Hotspot uses to provide these properties is work stealing.

Aurora-Blumofe-Plaxton (ABP) deques [4][5] as shared buffers backed up by per-thread private queues (described e.g. by Horie et al. [6]) are the main data structures of the work stealing mechanism. This lock-free algorithm has been used for more than 12 years: however changes in the typical number of threads available and increased parallelism in the VM let the relative cost of memory barriers increase significantly. There has also been significant research in the area of work stealing and in the area of work stealing in a VM in particular (e.g. [6][7][8]) during this time.

There is one particular interesting idea that could be applied successfully to work stealing in garbage collection: Idempotent Work Stealing (IWS) [9]. It weakens one of the principles of work stealing, by allowing that a given work unit may be executed more than once instead of only once. This allows additional overhead reducing optimizations at the cost of some re-work.

The desired goal of this thesis is to retire and replace the existing ABP deque as main datastructure by implementing and comparing two to three algorithms from the literature in the Hotspot VM with ABP. If applicable, loosen the guarantees of one or more algorithms according to the idempotent work stealing principle and measure any additional gains.

The goals of this thesis are:

- Perform literature research into existing work stealing techniques and memory models on contemporary microprocessors.
- Implement two to three different work stealing algorithms selected in conjunction with your advisor in the HotSpot VM.
- Try to apply the IWS principle on one of the algorithms, i.e. trade overhead for starting to do work multiple times against memory barrier overhead.
- Measure and compare the impact of these implementations on benchmarks used in the literature on x86-64 and ARM64 as representatives of strongly and weakly ordered memory architectures. This should help validating the correctness of the implementations.

The result of the thesis should be contributed to the OpenJDK open source project on success. This requires the student to understand and sign an Oracle Contributors Agreement [10] at the start of the thesis work.

The progress of the project should be discussed at least every two weeks with the advisor. A time schedule and a milestone plan must be set up within the first 3 weeks. It should be continuously refined and monitored to make sure that the thesis will be completed in time.

The final version of the thesis should be submitted not later than 31.03.2020.

[1] OpenJDK project, http://openjdk.java.net/

[2] David Detlefs, Christine Flood, Steve Heller, and Tony Printezis. 2004. Garbage-first garbage collection. In Proceedings of the 4th international symposium on Memory management (ISMM '04). ACM, New York, NY, USA, 37-48. DOI=http://dx.doi.org/10.1145/1029873.1029879

[3] Tony Printezis and David Detlefs. 2000. A generational mostly-concurrent garbage collector. SIGPLAN Not. 36, 1 (October 2000), 143-154. DOI: https://doi.org/10.1145/362426.362480

[4] Nimar S. Arora, Robert D. Blumofe, and C. Greg Plaxton. 1998. Thread scheduling for multiprogrammed multiprocessors. In Proceedings of the tenth annual ACM symposium on Parallel algorithms and architectures (SPAA '98). ACM, New York, NY, USA, 119-129. DOI= http://dx.doi.org/10.1145/277651.277678

[5] Nhat Minh Lê, Antoniu Pop, Albert Cohen, and Francesco Zappa Nardelli. 2013. Correct and efficient work-stealing for weak memory models. In Proceedings of the 18th ACM SIGPLAN symposium on Principles and practice of parallel programming (PPoPP '13). ACM, New York, NY, USA, 69-80. DOI=http://dx.doi.org/10.1145/2442516.2442524

[6] Michihiro Horie, Hiroshi Horii, Kazunori Ogata, and Tamiya Onodera. 2018. Balanced double queues for GC work-stealing on weak memory models. In Proceedings of the 2018 ACM SIGPLAN International Symposium on Memory Management (ISMM 2018). ACM, New York, NY, USA, 109-119. DOI: https://doi.org/10.1145/3210563.3210570

[7] M. Wu and X.-F. Li. Task-pushing: a scalable parallel gc marking algorithm without synchronization operations. In IPDPS, pages 1–10, 2007.

[8] Junjie Qian, Witawas Srisa-an, Du Li, Hong Jiang, Sharad Seth, and Yaodong Yang. 2015. SmartStealing: Analysis and Optimization of Work Stealing in Parallel Garbage Collection for Java VM. In Proceedings of the Principles and Practices of Programming on The Java Platform (PPPJ'15). ACM, New York, NY, USA, 170-181. DOI: https://doi.org/10.1145/2807426.2807441

[9] Maged M. Michael, Martin T. Vechev, and Vijay A. Saraswat. 2009. Idempotent work stealing. In Proceedings of the 14th ACM SIGPLAN symposium on Principles and practice of parallel programming (PpoPP '09). ACM, New York, NY, USA, 45-54. DOI=http://dx.doi.org/10.1145/1504176.1504186

[10] Oracle Contributor Agreement (OCA) and Frequently Asked Questions, http://openjdk.java.net/legal/