INF-2990 Bachelor's Thesis in Informatics

Verification of correctness of a concurrent hash map with TLA+

Åsmund Aqissiaq Arild Kløvstad - Advisor: Håvard D. Johansen

I. Introduction

As early as 1977 Leslie Lamport noted the difficulty and importance of writing correct programs in a concurrent setting[1]. Since then concurrency has become increasingly important due to the power wall and the high level of parallelism in modern CPUs[2].

It is difficult for programmers to reason about concurrent execution and this leads to bugs which are often subtle and difficult to reproduce[3]. It is useful, then, to be able to prove the correctness of such programs. The main approaches to this are

- exhaustive testing,
- formal proofs, and
- model checking.

Exhaustive testing is often time consuming and may miss rare or unexpected errors. Formal proofs prove correctness for all cases, but may be outside the expertise of many programmers. A model checker is used to check the specification of a program for correctness by viewing it as a finite state machine and searching the possible state space[4]. It is of interest whether such an approach can be as good as a formal proof while still being practical for programmers.

II. Project Description

This project will use the TLA specification language and the TLC model checker to verify the correctness of a concurrent data structure. The model checker can then be used to check the data structure for other properties.

The data structure in question is a lock-free concurrent hashmap[5]. This structure is formally proven to be correct – specifically Shalev et.al show that the keys are sorted, that each key points to the correct bucket, and that the basic operations *insert*, *find* and *delete* behave correctly – so using a model checker to confirm will strengthen both trust in the formal proof and in model checkers' usefulness.

Additionally, once the specification is written the model can be used to check other properties of the data structure such as liveness or type correctness.

III. Suggested Reading List

The following works are recommended for completing this course.

[4] Edmund M. Clarke, E. Allen Emerson, and Joseph Sifakis. "Model Checking: Algorithmic Verification and Debugging". In: *Commun. ACM* 52.11 (Nov. 2009), pp. 74–84. ISSN: 0001-0782. DOI: 10.1145/1592761.1592781. URL: https://doi.org/10.1145/1592761.1592781.

[5] Ori Shalev and Nir Shavit. "Split-Ordered Lists: Lock-Free Extensible Hash Tables". In: *J. ACM* 53.3 (2006), pp. 379–405. ISSN: 0004-5411. DOI: 10.1145/1147954.1147958. URL: https://doi.org/10.1145/1147954.1147958.

[6] Jørgen Aarmo Lund. "Verification of the Chord protocol with TLA+". In: (2019). URL: https://munin.uit.no/bitstream/handle/10037/15613/thesis.pdf?sequence=2%7B%5C&%7DisAllowed=y.

[7] Leslie Lamport et al. "Specifying and Verifying Systems with TLA+". In: *Proceedings of the 10th Workshop on ACM SIGOPS European Workshop*. EW 10. New York, NY, USA: ACM, 2002, pp. 45–48. DOI: 10.1145/1133373.1133382. URL: http://doi.acm.org/10.1145/1133373.1133382.

[8] Chris Newcombe et al. "How Amazon Web Services Uses Formal Methods". In: *Commun. ACM* 58.4 (Mar. 2015), pp. 66–73. ISSN: 0001-0782. DOI: 10.1145/2699417. URL: https://doi.org/10.1145/2699417.

[9] Leslie Lamport. *The TLA+ Video Course*. Online Video. Mar. 2018.

Note that this reading list should not be considered final. The student is expected to research the given topic and expand this list for the final report.

IV. Deliverables

The final hand-in will consist of

- a thesis describing the work done and the results of model checking
- a specification of Shalev et. al's hashmap written in PlusCal

The final deadline is June 1st at 23:59 and the project will be graded on an A-F letter scale. The project is worth 10 credits.

V. Timeline

- Week 7 Project start
- Week 9 Thesis first draft
- Week 12 Specification first draft
- Week 14 Mid-project eval
- Week 20 Final draft hand-in of thesis and specification
- Week 22 Final hand-in

References

[1] L Lamport. "Proving the Correctness of Multiprocess Programs". In: *IEEE Trans. Softw. Eng.* 3.2 (1977), pp. 125–143. ISSN: 0098-5589. DOI: 10.1109/TSE.1977.229904. URL: https://doi.org/10.1109/TSE.1977.229904.

[2] Andrew S. Tanenbaum and Herbert Bos. *Modern Operating Systems*. 4th. USA: Prentice Hall Press, 2014. ISBN: 013359162X.

[3] Ivan Beschastnikh et al. “Debugging Distributed Systems”. In: *Commun. ACM* 59.8 (July 2016), pp. 32–37. ISSN: 0001-0782. DOI: 10.1145/2909480. URL: https://doi.org/10.1145/2909480.

[4] Edmund M. Clarke, E. Allen Emerson, and Joseph Sifakis. “Model Checking: Algorithmic Verification and Debugging”. In: *Commun. ACM* 52.11 (Nov. 2009), pp. 74–84. ISSN: 0001-0782. DOI: 10.1145/1592761.1592781. URL: https://doi.org/10.1145/1592761.1592781.

[5] Ori Shalev and Nir Shavit. “Split-Ordered Lists: Lock-Free Extensible Hash Tables”. In: *J. ACM* 53.3 (2006), pp. 379–405. ISSN: 0004-5411. DOI: 10.1145/1147954.1147958. URL: https://doi.org/10.1145/1147954.1147958.

[6] Jørgen Aarmo Lund. “Verification of the Chord protocol with TLA+”. In: (2019). URL: https://munin.uit.no/bitstream/handle/10037/15613/thesis.pdf?sequence=2%7B%5C&%7DisAllowed=y.

[7] Leslie Lamport et al. “Specifying and Verifying Systems with TLA+”. In: *Proceedings of the 10th Workshop on ACM SIGOPS European Workshop*. EW 10. New York, NY, USA: ACM, 2002, pp. 45–48. DOI: 10.1145/1133373.1133382. URL: http://doi.acm.org/10.1145/1133373.1133382.

[8] Chris Newcombe et al. “How Amazon Web Services Uses Formal Methods”. In: *Commun. ACM* 58.4 (Mar. 2015), pp. 66–73. ISSN: 0001-0782. DOI: 10.1145/2699417. URL: https://doi.org/10.1145/2699417.

[9] Leslie Lamport. *The TLA+ Video Course*. Online Video. Mar. 2018.