Hochschule Karlsruhe
University of
Applied Sciences
HKA

Dynamische Programmanalysen für nebenläufige Programme - Data Race Prediction mit TSan V2

Seminararbeit

Student: Frank Ling

Matrikelnummer: 79496

Universität: Hochschule Karlsruhe – Technik und Wirtschaft

Studiengang: Informatik, Master

Semester: Sommersemester 2023

Dozent: Prof. Martin Sulzmann

Bearbeitet am: June 5, 2023

Contents

1 Introduction

Nowadays concurrent programs are very common in order to make use of 'hyper-threading and multi-core architectures'[1, p. 14]. 'Due to the highly non-deterministic behavior of concurrent programs' [2, p. 1] data races may arise but can also be hard to find, as they also 'may only arise under a specific schedule' [2, p. 1]. This seminar work shows the motivation and background for the data prediction tool ThreadSanitizer (TSan) V2, which differentiates itself from the first version by utilizing happens-before methods instead of the lockset method. Afterwards the concepts of the FastTrack [4] algorithm will be shown as TSan uses a slightly modified version of the FastTrack algorithm. Examples showing the limits of FastTrack and TSan follow, as they are both incomplete and thus do not find every data race. In the following the same notation as in [2] will be used for traces and events.

2 Motivation and Examples

As stated in the introduction concurrent programs are commonly used and are inherently prone to data races. The following example shows a program written in the programming language C++, which exhibits a data race:

Figure 1: PLACEHOLDER: show example in C showing data race, see slide #2

3 Background

3.1 Data Race

Sulzmann and Stadtmüller [2, p. 1] define data races as follows: 'A data race arises if two unprotected, conflicting read/write operations from different threads happen at the same time.' Further they state that a data race can be described as: '[...] two read/write events on the same variable where at least one of them is a write event and both events result from different threads.'

3.2 Happens-Before Relation

The example shown in figure 1 can be represented by the following trace:

	1#	2#
1.	w(x)	
2.	acq(y)	
3.	rel(y)	
4.		acq(y)
5.		w(x)
6.		rel(y)

Table 1: Trace 1

3.3 Vector-Clock

3.4 Epoch

4 FastTrack + TSan

5 Conclusion

List of Literature

[1] A. R. Molla, G. Sharma, P. Kumar, and S. Rawat, Eds., *Distributed computing and intelligent technology : 19th international conference, icdcit 2023, bhubaneswar, india, january 18–22, 2023, proceedings*, Cham, 2023. [Online]. Available: `https://link.springer.com/book/10.1007/978-3-031-24848-1`.

[2] M. Sulzmann and K. Stadtmüller, "Efficient, near complete and often sound hybrid dynamic data race prediction (extended version)," *CoRR*, vol. abs/2004.06969, 2020. arXiv: `2004.06969`. [Online]. Available: `https://arxiv.org/abs/2004.06969`.

[3] D. S. Fava and M. Steffen, "Ready, set, go!: Data-race detection and the go language," *Science of Computer Programming*, vol. 195, p. 102 473, 2020, ISSN: 0167-6423. DOI: `https://doi.org/10.1016/j.scico.2020.102473`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0167642320300836`.

[4] C. Flanagan and S. Freund, "Fasttrack: Efficient and precise dynamic race detection," vol. 53, Jun. 2009, pp. 121–133. DOI: `10.1145/1542476.1542490`.

List of Figures

List of Tables