Improving User-Defined Permission Models in Viper

Anqi Li
anqili@student.ethz.ch
Practical Work Project Supervised by Thibault Dardinier
Department of Computer Science
ETH Zurich, Switzerland

October 18, 2022

1 Introduction

Viper [5] is a powerful toolchain and infrastructure for program verification. Unlike similar verification software such as Boogie [4] and Why3 [3], Viper provides native support for permission-based reasoning by using separation logic [6], which is useful for verifying concurrent programs that manipulate shared data structures.

In Viper, fractional permissions [1][2], where a permission amount is defined as a rational value in the range of [0; 1], are used to express heap location ownership. More specifically, writing to a heap location requires a permission amount of 1, while reading from a heap location requires a non-zero permission amount. However, such permission model is not suitable for every situation. Consider the example in Listing 1 where a predicate for a tree data structure is formulated. When $p > \frac{1}{2}$, the predicate always describes a tree, while when $p \leqslant \frac{1}{2}$, it can describe a directed acyclic graph (DAG) rather than a tree. Consider a DAG with 4 distinct nodes T, L, R, B shown in Figure 1. If `Tree(T, p)` holds, then by definition of the predicate, `Tree(L, p)`, `Tree(R, p)` and `Tree(B, p+p)` must also hold. However, when $p > \frac{1}{2}$, $p + p > 1$ is an undefined permission amount, which invalidates such DAG. In contrast, when $p \leqslant \frac{1}{2}$, $p + p \leqslant 1$ is a valid permission amount, so this DAG can be mistakenly considered as a tree. Consequently, it is desirable to introduce alternative permission models and allow users to select the permission model dependent on the use case.

```
1  field leftChild: Ref
2  field rightChild: Ref
3  field elem: Int
4
5  predicate Tree(this: Ref, p: Perm) {
6      this != null && p > 0/1 ==>
7          acc(this.elem, p) &&
8          acc(this.leftChild, p) &&
9          acc(this.rightChild, p) &&
10         Tree(this.leftChild, p) &&
11         Tree(this.rightChild, p)
12 }
```

Listing 1: Tree predicate written in Viper

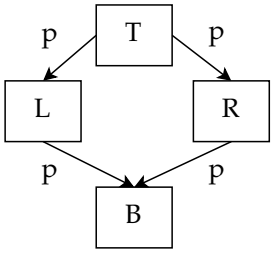

Figure 1: A DAG described by the tree predicate in Listing 1

2 Project Goals and Tasks

The core goal of this project is to improve the experimental feature of allowing user-defined permission models in Viper. The project will build upon the existing work in [7] which documents the theoretical foundation, methodology and implementation details for this feature. The project goal can be accomplished by completing the following tasks:

1. Improve the existing implementation that enables user-defined permission models to make this feature usable for Viper users.

2. Write interesting programs using different permission models to show the usefulness of the tool and also explore its limitations.

3. Add a feature that allows users to specify a permission model per field. In other words, we allow users to customize the permission model of each field.

4. Explore possible designs of a generalized version of predicate multiplication. After the new feature described in 3 is introduced, multiple permission models can appear in a predicate, so it is necessary to generalize predicate multiplication to adapt to such change.

References

[1] Richard Bornat et al. "Permission Accounting in Separation Logic". In: *SIGPLAN Not.* 40.1 (Jan. 2005), pp. 259–270. ISSN: 0362-1340. DOI: 10.1145/1047659.1040327. URL: https://doi.org/10.1145/1047659.1040327.

[2] John Boyland. "Checking Interference with Fractional Permissions". In: *Static Analysis*. Ed. by Radhia Cousot. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 55–72. ISBN: 978-3-540-44898-3.

[3] Jean-Christophe Filliâtre and Andrei Paskevich. "Why3 — Where Programs Meet Provers". In: *Programming Languages and Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 125–128. ISBN: 978-3-642-37036-6.

[4] K. Rustan M. Leino. "This is Boogie 2". June 2008. URL: https://www.microsoft.com/en-us/research/publication/this-is-boogie-2-2/.

[5] Peter Müller, Malte Schwerhoff, and Alexander J. Summers. "Viper: A Verification Infrastructure for Permission-Based Reasoning". In: *Verification, Model Checking, and Abstract Interpretation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 41–62. ISBN: 978-3-662-49122-5.

[6] J.C. Reynolds. "Separation logic: a logic for shared mutable data structures". In: *Proceedings 17th Annual IEEE Symposium on Logic in Computer Science*. 2002, pp. 55–74. DOI: 10.1109/LICS.2002.1029817.

[7] Matthias Roshardt. "Extending the Viper Verification Language with User-Defined Permission Models". MA thesis. 2021.