Verifying Programs with a Vampire

Case studies in verifying Selection Sort and Key-Value Stores

1. Introduction

- Vampire is one of the **fastest** automated theorem provers (ATPs) used for **software verification.**
- I tried to prove the **correctness** of **Selection Sort** and of a **Key-Value Store**.

2. Research Question

How can Vampire be used to verify algorithms, and what are the **capabilities** and **limitations** of Vampire?

3. Background

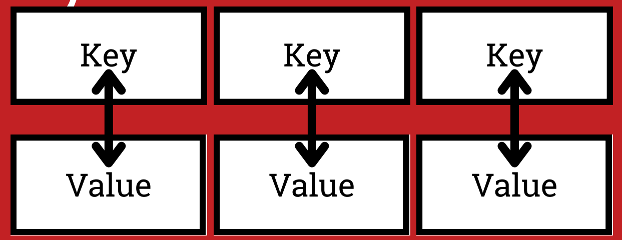
- Vampire uses typed first-order logic and specific programming constructs.
- Vampire reads problems in TPTP, a standard language for ATPs.

$$(\forall x : \mathbb{Z})$$
$$(Even(x) \lor Odd(x))$$

An example of a tautology in typed logic.

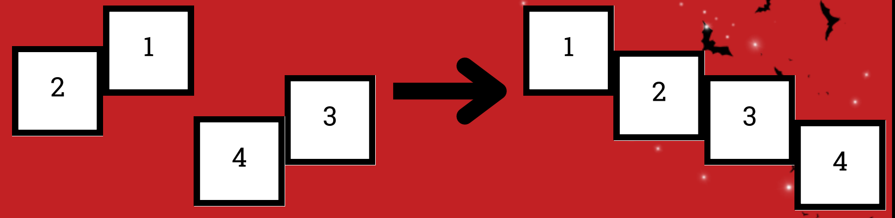
4. Problem Descriptions

Key-Value Store



To Prove:
Properties stating how different methods should work.

Selection Sort



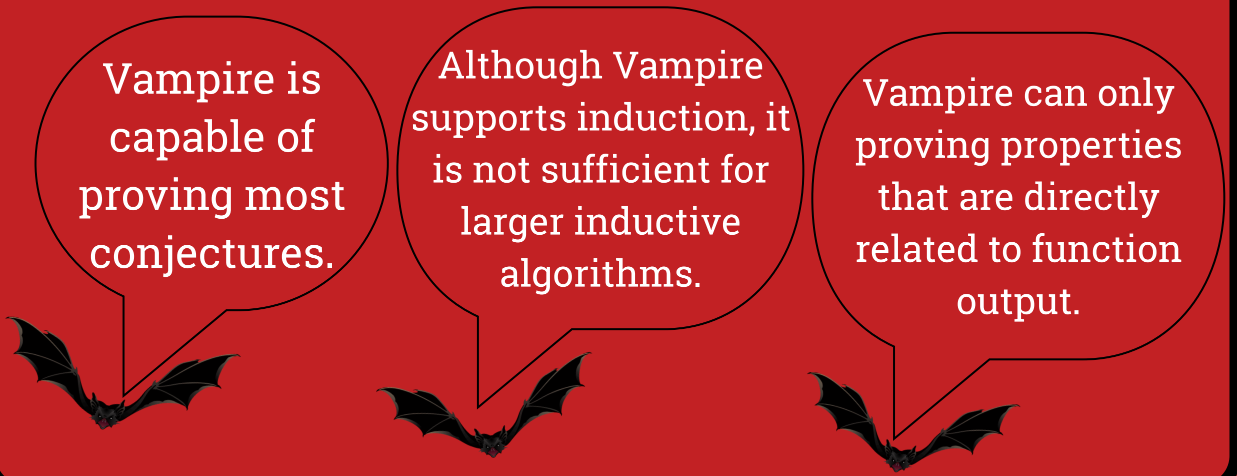
To Prove:

- Sortedness
- Permutation Equivalence

5. Results

- The majority of the Key-Value Store could be verified.
- None of the properties of Selection Sort could be proven.

6. Discussion



Vampire is capable of proving most conjectures.
Although Vampire supports induction, it is not sufficient for larger inductive algorithms.
Vampire can only proving properties that are directly related to function output.

7. Limitations of Vampire

- Difficult to prove multiple properties.
- Bugs with $\LaTeX$ output.
- It does not always inform you if your problem is ill-formed.

8. Conclusion

- Vampire was capable of proving most properties, except those heavily reliant on induction or that were too general.
- Documentation and guides for Vampire could be greatly improved.
- Code can be found on: https://github.com/mbalfakeih/Vampire-Case-Study

Image References

Author: Mohammed Balfakeih[1]
Contact: M.H.M.Balfakeih@student.tudelft.nl

Supervisors: Dr. Benedikt Ahrens[1], Kobe Wullaert[1]
[1]EEMCS, Delft University of Technology, The Netherlands