The Faustus Programming Language

Summer 2022

Adding formally verified parameterized abstractions to the smart contract language Marlowe.

Problem Statement

Marlowe is a formally verified domain specific language (DSL) for programming financial smart contracts. The only control flow statements in Marlowe are an if-then-else statement and a When statement that waits for valid user input. There are no looping constructs, no recursion, and no parameterized functions or contracts. Marlowe contracts can be executed on the blockchain, it is a good target for compilation; however, programs become hard to read and maintain. Faustus is designed to address these weaknesses.

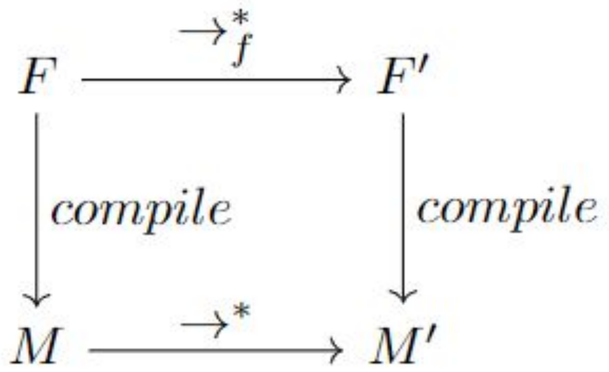
Background

- Isabelle/HOL (Isabelle) is an interactive theorem prover that can be used to prove the correctness of programs and programming languages.
- Marlowe and its evaluator have been formalized in the Isabelle theorem prover. Marlowe contracts will not “lose” money and can run on the blockchain.
- The methods for adding functions and procedures with parameters are cornerstones in general purpose programming languages.

Faustus Team Members †

Kegan McIlwaine - COSC Ph.D. Student
Stone Olguin - COSC M.S. Student
Professor James Caldwell - Advisor

Methods



- Formalize the semantics of Marlowe with respect to the already formalized evaluator. $\to^*$
- Create the Faustus abstract syntax by adding new constructs to Marlowe that will bind parameterize procedures to identifiers.
- Formalize the semantics of Faustus. $\to_f^*$
- Formalize a type system that will prevent undefined behavior in the new semantics.
- Formalize a compiler that maps Faustus contracts to Marlowe contracts. (*compile*)
- Proved the compiler correct with respect to the formalized semantics.

Results

In 4,883 lines of Isabelle definitions and proofs, we have formalizd the Faustus DSL for writing financial contracts. The Isabelle proofs show that the compiler is correct. Faustus allows programmers to write maintainable code by reducing code duplication. An example 5 party multi-signature contract written in 199 lines of Faustus code requires 4,030 lines of Marlowe code.

Challenges & Future Work

Challenges

- Marlowe’s limited features made compilation of Faustus’ complex features quite difficult.
- Formal verification depends on correct formulations of specifications. This is difficult, and errors are only discovered when attempting proofs.

Future Work

- Developing modular methods for extending DSLs.
- Formalizing and adding new features to Faustus. *e.g.* merkleization of contracts compactifies them, this allows them to be stored on the blockchain which is needed due to severe time and space constraints for on-chain code.

† The research presented here was generously supported by a grant from IOG Singapore Pte. Ltd. and funding from the State of Wyoming and the University of Wyoming and the College of Engineering and Physical Sciences. Special thanks to the Marlowe development team at IOG for their biweekly meetings over the last year and a half.

Advisor: Dr. James Caldwell
Group Members:
- Kegan McIlwaine (kmcilwai@uwyo.edu)
- Stone Olguin (aolguin1@uwyo.edu)
- James Caldwell (jlc@uwyo.edu)

University of Wyoming | IOG Advanced Blockchain Research Lab