CS421 Sum 20 Project Report

Author: Lu Liu **NetID**: lul8

Overview

This project is an attempt to build an interpreter for Lua programming language in Haskell. The motivation for picking this task is the following:

1. It gives me the opportunity to utilize and fully understand some of the major topics taught in this course including:
   programming in a functional language (Haskell);
   Using monads to simulate stateful computation;
   Understanding formal language syntax and semantics specification, converting them into implementation；
   Building a parser using combinator parsing (parsec).
2. It allows me to learn new things not specifically taught in the course, but are of great interest to me:
   Understanding Lua;
   Building a complete haskell project using the Haskell Tool Stack;
   Learning and Implementing randomized property testing using QuickCheck;
   Learning to use testing framework Tasty.

Implementation

To limit the scope of the project, only a subset of the functionalities of Lua is supported, they are listed below:

- 5 of the 8 basic value types in Lua are supported: nil, boolean, int, string and table. We omitted function, userdata and thread.
- Evaluation of Constant expression, variable expression
- Evaluation of all binary and unary operations of Lua except bitwise operations
- Table construction, table lookup and table elements assignment
- Print statement
- Multiple assignment statement
- Block statement (sequence of statements)

The interpreter also has a lexer, a parser and REPL which supports all the above functionalities.

Error handling is omitted.

Tests

Following the testing framework of MP2, this project implemented randomized property testing for constant expression evaluations. For all other functionalities, unit testing are implemented.

For each test, we provide a line of code (as a String) and its reference execution result (obtained using a reference Lua interpreter). Then the line of code is parsed and executed by our program. The result is compared with the reference result.

Listing

Github: https://github.com/336699github/cs421_project

In directory app/:

- Main.hs: REPL frontend

In directory app/Lua/:

- Core.hs: core language data structures
- Parse.hs: lexer and parser
- Eval.hs: monatic evaluator
- Runtime.hs: primitive operations of the language

In directory test/:

- Spec.hs: front end for testing
- PropertyTests.hs: Code generator and property function for property testing
- UnitTests.hs: all unit test cases.

Citation

UIUC CS421 Programming Language and Compilers Summer 2020 Machine Problems 2 and 5

Cheplyaka, Roman. “Tasty: Modern and Extensible Testing Framework.” Hackage, 2020. https://hackage.haskell.org/package/tasty-1.3.1.

Claessen, Koen, and John Hughes. “QuickCheck.” Proceedings of the fifth ACM SIGPLAN international conference on Functional programming - ICFP '00, 2000. https://doi.org/10.1145/351240.351266.

Ierusalimschy, Roberto, Luiz Henrique de Figueiredo, and Waldemar Celes. “Lua 5.4 Reference Manual,” 2020. http://www.lua.org/manual/5.4/manual.html.

Leijen, Daan, and Paolo Martini. “Parsers.” Text.Parsec, 2007. https://hackage.haskell.org/package/parsec-3.1.14.0/docs/Text-Parsec.html.

Soldevila, Mallku, Beta Ziliani, Bruno Silvestre, Daniel Fridlender, and Fabio Mascarenhas. “Decoding Lua: Formal Semantics for the Developer and the Semanticist.” ACM SIGPLAN Notices 52, no. 11 (2017): 75–86. https://doi.org/10.1145/3170472.3133848.