**A STATIC SEMANTICS FOR HASKELL UTILIZING THE K-FRAMEWORK**

Bradley Morrell

The Department of Electrical and Computer Engineering

bmorrel2@illinois.edu

Fall 2018

Advisor: Professor Elsa Gunter

**ABSTRACT**

For this thesis, I introduce a static semantics for Haskell by utilizing the K-Framework. This implementation includes support for the module system of Haskell but not for type classes. There are many layers that have to be implemented in K before type inference can be performed. The first part of the implementation is the entire context free syntax of Haskell in K. Since all the syntax is included, any program written in Haskell extended syntax can be parsed into an abstract syntax tree. However, this includes only the Haskell extended syntax

but does not include the syntactic short-cuts such as treating tabs as syntactic sugar for grouping constructs such as curly braces. Programs that include multiple modules can be parsed, but the multiple modules must be written in a single file. This is unlike how the Glasgow Haskell Compiler allows for module imports, where each module must be kept in separate files. The multiple modules are then made as nodes in a directed acyclic graph. A directed edge in the graph represents a module importing another module. This graph is used for importing the user defined types from one module into another module. Context sensitive checks and type inference are then performed on modules. The static semantics specifies that, at each node in the graph, assuming all child modules are already checked and inferred, the user defined types of each of the child modules are imported into the module at the given node. All rules of the Haskell type system must take mutual recursion into account. There is repeated layering of inferences in Haskell. My semantics is mathematically precise and executable. I utilized the executability of the semantics to test both positive inferences and exceptional inferences. This is part of a larger project to give a formal semantics to Haskell.