NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCE
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

POSTGRADUATE STUDIES
“COMPUTER SYSTEMS: SOFTWARE AND HARDWARE”

MASTER THESIS

Datalog Based Symbolic Program Reasoning for Java

Christos V. Vrachas

Supervisor: **Yannis Smaragdakis,** Professor NKUA

ATHENS

June 2019

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΜΕΤΑΠΤΥΧΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
“ΥΠΟΛΟΓΙΣΤΙΚΑ ΣΥΣΤΗΜΑΤΑ: ΛΟΓΙΣΜΙΚΟ ΚΑΙ ΥΛΙΚΟ”

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Συμβολική Συλλογιστική Προγραμμάτων για Java, βασισμένη σε Datalog

Χρίστος Β. Βραχάς

Επιβλέπων: **Γιάννης Σμαραγδάκης,** Καθηγητής ΕΚΠΑ

ΑΘΗΝΑ

Ιούνιος 2019

MASTER THESIS

Datalog Based Symbolic Program Reasoning for Java

Christos V. Vrachas **R.N.:** M1608

SUPERVISOR: **Yannis Smaragdakis,** Professor NKUA

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Συμβολική Συλλογιστική Προγραμμάτων για Java, βασισμένη σε Datalog

Χρίστος Β. Βραχάς **Α.Μ.:** M1608

ΕΠΙΒΛΕΠΩΝ: Γιάννης Σμαραγδάκης, Καθηγητής ΕΚΠΑ

ABSTRACT

abstract english

SUBJECT AREA: Static Program Analysis, Symbolic Reasoning, Propositional Logic

KEYWORDS: flow-sensitivity, path-sensitivity, inference rules and logic proofs

ΠΕΡΙΛΗΨΗ

abstract greek

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Στατική Ανάλυση Προγραμμάτων, Συμβολική Συλλογιστική, Προτασιακή Λογική

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: ”ευαισθησία”-ροής, ”ευαισθησία”-μονοπατιού, κανόνες συμπερασμού και αποδείξεις προτασιακής λογικής

This master thesis is dedicated to ...

ACKNOWLEDGEMENTS

I would like to thank ....

June 2019

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF SOURCE CODES

1. INTRODUCTION

FooBar

1.1 Thesis Structure

BarBar

2. BACKGROUND

Several program analysis techniques have been proposed in the literature, in order to aid developers and users discover interesting program properties within their software. For example, one might be interested in finding out whether there are memory leaks or whether some piece of code is reachable.

Such techniques come in different flavors, either static or dynamic, and are usually based on strong mathematical concepts. In this section we provide a brief background on these concepts and the frameworks utilized towards the development of program analysis tools.

2.1 Static Program Analysis

Static program analysis is a program analysis technique that aims to reason about a program's behaviors without actually executing it. It has been heavily utilized by optimizing compilers since their early stages and it has also found several other applications, among the areas of softare security, software correctness [1]. The question of whether a program is correct or may terminate for all possible inputs is in general undecidable. However, static analysis techniques are able to tackle undecidability by overapproximating or underapproximating the initial problem, attempting to reason over a simplified version of it.

There is a plethora of different static analysis algorithms in the literature, each one met in several domains. For example, one may utilize analyses such as *liveness* and/or *pointer* analyses in an optimizing compiler with the intention of eliminating dead code regions or performing a constant propagation/folding optimization. A *reachability* analysis that determines whether a specific program point is reachable could be used for example by a software correctness tool to make sure that an erroneous state is actually never reached.

There is a variety of design choices that may prove essential towards the *scalability* and *precision* of a static program analysis algorithm.

2.1.1 Whole-Program vs. Partial Analyses

2.1.2 Flow-Sensitivity

2.1.3 Path-Sensitivity

2.2 Symbolic Execution

Foo

2.3 Theorem Provers

Lalal

2.4 Datalog

Datalog stuff

3. STATIC DECLARATIVE SYMBOLIC REASONING

our approach

3.1 Schema Relations and Program Expression Trees

input facts and Reasoning

3.2 Path Expressions

pathhz

3.3 Boolean Symbolic Reasoning

reasoning logic

3.3.1 Propositional Logic Axioms

pure axioms

3.3.2 Propositional Logic Inference Rules

inferencer

4. EVALUATIONS

foolala

5. RELATED WORK

foorelated

6. CONCLUSIONS AND FUTURE WORK

FooConclusions

ABBREVIATIONS - ACRONYMS

TODO	to do

REFERENCES

[1] Anders Møller and Michael I. Schwartzbach. Static program analysis, October 2018. Department of Computer Science, Aarhus University, http://cs.au.dk/~amoeller/spa/.