University of Technology Sydney
School of Computer Science
Faculty of Engineering and Information Technology

Dissertation submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

under principal supervisor **Dr. Yulei Sui** and co-supervisor **Dr. Shiping Chen**

Data Structures for Points-To Analysis

Mohamad Barbar

October 2022

Certificate of Original Authorship

I, Mohamad Barbar, declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the School of Computer Science, Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Production Note:
Signature: Signature removed prior to publication.
Date: 06/10/2022

Preface

With the end of my candidature, more than 4 years later, it is time to collect my work in one place. I hope it is of use to some.

Firstly, major thanks is due to my family who were patient with me during my candidature. 4 years is not a short period of time.

I also thank my principal supervisor, Dr. Yulei Sui, for the many long hours of discussion on both the big picture and the smallest of details. Certainly, working with Yulei got me started with static analysis, and it is an area I would like to continue hacking on. The opportunity to work on SVF has also been great as my first prolonged exposure to a large codebase where performance really matters. I also thank my co-supervisor, Dr. Shiping Chen, and CSIRO's Data61 as a whole, for supporting my research with the Data61 scholarship.

My thanks also goes to my peers at UTS who kept me company, particularly before moving to work-from-home at the start of the pandemic; Omar, Mingshan, Ayman, Joakim (who also inspired some of the work in Chapter 4), Ibraheem, Yahya, and Akram. I also thank the Program Analysis Group, especially Yanxin and Yuxiang with whom I started around the same time, though it was unfortunate meetings had to move online. I must also say that all my interactions with administration at UTS has been pleasant, and express my thanks in that direction. At CSIRO, I also thank Ejaz for organising seminars allowing for some nice exchange of ideas. Unfortunately, that too was cut short by the pandemic.

Finally, most of all, I give thanks to the All-Wise, the All-Knowing and hope for His forgiveness for my many shortcoming during this candidature.

Mohamad Barbar
Sydney, Australia
October 2022

Abstract

In this dissertation, we present improvements to data structures, and the algorithms upon them, for points-to analysis. Our focus is mainly on flow-sensitive analysis but our techniques can either be applied to other analyses or used in analyses which combine flow-sensitivity with other sensitivities. For staged flow-sensitive analysis (SFS), we introduce a pre-analysis (meld versioning) where we determine when it is possible to reuse the points-to sets of individual address-taken variables at different program points, then perform the main analysis using this information. Meld versioning is also amenable to parallelisation with minimal effort. For points-to sets, we introduce an improved bit-vector stripping both leading and trailing zero-words, then use that to aid in improving the object-to-identifier mapping required to use bit-vectors as points-to sets. We frame this as an integer programming problem, yielding an optimal solution but with impractical performance, and so we develop a more approximate (yet extremely fast) method based on hierarchical clustering. We also describe hash consing and memoisation, along with some optimisations which would otherwise be impractical, for points-to sets. We have implemented our techniques in open source points-to analysis framework SVF, and upon evaluating with 12 open source programs, we find, on average, a speedup of almost $6\times$ and a reduction in memory usage of more than $3.97\times$ when compared with a baseline SFS.

Publications

Mohamad Barbar, Yulei Sui, and Shiping Chen. 2020. Flow-Sensitive Type-Based Heap Cloning. In 34th European Conference on Object-Oriented Programming (ECOOP '18, Vol. 166). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Germany, 24:1–24:26. https://doi.org/10.4230/LIPIcs.ECOOP.2020.24

Mohamad Barbar, Yulei Sui, and Shiping Chen. 2021. Object Versioning for Flow-Sensitive Pointer Analysis. In 2021 IEEE/ACM International Symposium on Code Generation and Optimization (CGO '21). IEEE Computer Society, USA, 222–235. https://doi.org/10.1109/CGO51591.2021.9370334

Mohamad Barbar and Yulei Sui. 2021. Hash Consed Points-To Sets. In International Static Analysis Symposium (SAS '21). Springer, Germany, 25–48. https://doi.org/10.1007/978-3-030-88806-0_2

Mohamad Barbar and Yulei Sui. 2021. Compacting Points-to Sets through Object Clustering. Proceedings of the ACM on Programming Languages 5, OOPSLA, Article 159 (Oct. 2021), 27 pages. https://doi.org/10.1145/3485547

Colophon

This dissertation was created using XeLaTeX. The Maggi Memoir Thesis template, originally by Federico Maggi and later modified by Vel from `LaTeXTemplates.com`, is used as a base after some slight modifications. The body is set 10pt with Linux Libertine O primarily. Linux Biolinum O, Latin Modern Math, and Latin Modern Mono make occasional appearances. Graphs, in the graph theoretical sense, are drawn with TikZ/PGF, tables use the `booktabs` package, and the single line graph in this dissertation is drawn with Matplotlib (DejaVu Sans on the axes).

Contents

List of Figures

List of Tables

List of Acronyms

BDD	binary decision diagram	47
DUG	def-use graph	15
GLLVM	Whole Program LLVM in Go	18
ICFG	interprocedural control-flow graph	14
IP	integer programming	57
IR	intermediate representation	5
ISA	instruction set architecture	47
OOM	out of memory	20
PAG	pointer assignment graph	18
SFS	staged flow-sensitive analysis	xvi
SSA	static single assignment	6
STL	Standard Template Library	18
SVFG	sparse value-flow graph	xvi
VSFS	versioned staged flow-sensitive analysis	3