Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 307 (2014) 17–31

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Computing Invariants with Transformers: Experimental Scalability and Accuracy

Vivien Maisonneuve[1], Olivier Hermant[2] and François Irigoin[3]

MINES ParisTech, France

Abstract

Using abstract interpretation, invariants are usually obtained by solving iteratively a system of equations linking preconditions according to program statements. However, it is also possible to abstract first the statements as transformers, and then propagate the preconditions using the transformers. The second approach is modular because procedures and loops can be abstracted once and for all, avoiding an iterative resolution over the call graph and all the control flow graphs.
However, the transformer approach based on polyhedral abstract domains encurs two penalties: some invariant accuracy may be lost when computing transformers, and the execution time may increase exponentially because the dimension of a transformer is twice the dimension of a precondition.
The purposes of this article are 1) to measure the benefits of the modular approach and its drawbacks in terms of execution time and accuracy using significant examples and a newly developed benchmark for loop invariant analysis, ALICe, 2) to present a new technique designed to reduce the accuracy loss when computing transformers, 3) to evaluate experimentally the accuracy gains this new technique and other previously discussed ones provide with ALICe test cases and 4) to compare the executions times and accuracies of different tools, ASPIC, ISL, PAGAI and PIPS.
Our results suggest that the transformer-based approach used in PIPS, once improved with transformer lists, is as accurate as the other tools when dealing with the ALICe benchmark. Its modularity nevertheless leads to shorter execution times when dealing with nested loops and procedure calls found in real applications.

Keywords: model checking, abstract interpretation, static program analysis, linear relation analysis, automatic invariant detection, loop invariant, transformer, benchmark

1 Introduction

Using abstract interpretation, invariants are usually obtained by solving iteratively a system of equations linking preconditions according to program statements. However, it is also possible to abstract first the statements as state transformers, and then propagate the preconditions using these transformers. The second approach is

[1] Email: vivien.maisonneuve@cri.mines-paristech.fr
[2] Email: olivier.hermant@cri.mines-paristech.fr
[3] Email: francois.irigoin@cri.mines-paristech.fr

http://dx.doi.org/10.1016/j.entcs.2014.08.003

modular because procedures and loops can be abstracted once and for all, avoiding an iterative resolution over the call graph and all control flow graphs.

However, the transformer approach, based on polyhedral abstract domains [14,2], encurs two possible penalties: some invariant accuracy may be lost when computing transformers, and the execution time may increase exponentially because the dimension of a transformer is twice the dimension of a precondition. Polyhedral operators have a worst-case exponential complexity and transformers must deal with two values per variable, the value in the precondition, *i.e.* the past value, and the value in the postcondition, *i.e.* the new value, whereas preconditions require only the current value.

The purposes of this article are 1) to measure the benefits of the modular approach and its drawbacks in terms of execution time and accuracy using parametric examples and a newly developed benchmark suite for loop invariant analysis, ALICe [18], 2) to present a new technique developed to reduce the accuracy loss when computing loop invariants, namely control path transformers, 3) to evaluate the accuracy gains this new technique and older ones, previously discussed in [2] but not implemented, provide with ALICe test cases and 4) to compare the execution times and accuracies of different tools using either precondition propagation or transformer computation. Namely, we compare ASPIC [8], which is a standard abtract interpretation (AI) tool based on widening and acceleration, PAGAI [13], a SMT-based AI tool, the Integer Set Library, ISL [24], which is a library including a transitive closure for Presburger relations and PIPS [15,14], which is a compilation framework using polyhedral sets to abstract transformers and preconditions. The comparisons are difficult because the tools have different input and output languages, but this is dealt with by the ALICe framework.

The techniques considered in this paper to improve the accuracy of the transformer approach are the computation of transformers along different control paths to postpone convex hull operations until the precondition propagation phase, the iterative computation of new transformers based on previously computed preconditions [2], the exploitation of idempotent transformers [2] and the control simplification that can be obtained by splitting and specializing control nodes [17]. Although we strived to make this paper self-contained, it might be useful to read [2] first or when encountering difficulties.

The outline of the paper is the following. Since the transformer-based approach is unusual, we introduce it briefly in Section 2. We then present a first set of experimental results to explore the accuracy and execution time issues existing with the techniques presented in [2] (Section 3). Accuracy issues encountered can be traced back to early convex hull operations, and several techniques designed to postpone them as much as possible are detailed in Sections 4 and 5. Finally, we measure the impact of these improvements with ALICe (Section 6) and conclude.