

A Generic Framework for Interprocedural Analyses of Numerical Properties

Markus Müller-Olm¹ and Helmut Seidl²

¹ Universität Dortmund, Fachbereich Informatik, LS 5
Baroper Str. 301, 44221 Dortmund, Germany
markus.mueller-olm@cs.uni-dortmund.de

² TU München, Institut für Informatik, I2
80333 München, Germany
seidl@in.tum.de

Abstract. Relations among program variables like $1 + 3 \cdot \mathbf{x}_1 + 5 \cdot \mathbf{x}_2 \equiv 0$ [224] have been called *linear congruence relations*. Such a relation is *valid* at a program point iff it is satisfied by all reaching program states. Knowledge about non-trivial valid congruence relations is crucial for various aggressive program transformations. It can also form the backbone of a program correctness proof.

In his seminal paper [1], Philippe Granger presents an intraprocedural analysis which is able to infer linear congruence relations between integer variables. For affine programs, i.e., programs where all assignments are affine expressions and branching is non-deterministic, Granger's analysis is *complete*, i.e., infers *all* valid congruence relations between variables. No upper bound, though, has been proven for Granger's algorithm. Here, we present a variation of Granger's analysis which runs in polynomial time. Moreover, we provide an interprocedural extension of this algorithm. The polynomial algorithm as well as its interprocedural extension are obtained by means of multiple instances of a general framework for constructing interprocedural analyses of numerical properties. This framework can be used for different numerical domains such as fields or modular rings and thus also covers the interprocedural analyses of [2,3] where valid affine relations are inferred.

We also indicate how the base technique can be extended to deal with equality guards in the interprocedural setting.

References

1. P. Granger. Static Analysis of Linear Congruence Equalities among Variables of a Program. In *Int. Joint Conf. on Theory and Practice of Software Development (TAPSOFT)*, pages 169–192. LNCS 493, Springer-Verlag, 1991.
2. M. Müller-Olm and H. Seidl. Precise Interprocedural Analysis through Linear Algebra. In *31st ACM Symp. on Principles of Programming Languages (POPL)*, pages 330–341, 2004.
3. M. Müller-Olm and H. Seidl. Analysis of Modular Arithmetic. In *European Symposium on Programming (ESOP)*. Springer Verlag, 2005. To appear.