Superposition with Simplification as a Decision Procedure for the Monadic Class with Equality*

Leo Bachmair¹, Harald Ganzinger², Uwe Waldmann²

¹ Department of Computer Science, SUNY at Stony Brook, Stony Brook, NY 11794, U.S.A, leo@sbcs.sunysb.edu
² Max-Planck-Institut für Informatik, Im Stadtwald, D-W-6600 Saarbrücken, Germany, {hg,uwe}@mpi-sb.mpg.de

Abstract. We show that superposition, a restricted form of paramodulation, can be combined with specifically designed simplification rules such that it becomes a decision procedure for the monadic class with equality. The completeness of the method follows from a general notion of redundancy for clauses and superposition inferences.

1 Introduction

The monadic class is a fragment of first-order logic the decidability of which, not only for the case with equality but also for quantification over predicates, is known since (Löwenheim 1915). A simpler proof has later been given by Ackermann (1954), whose syntactic method of transforming the given formula into some kind of solved form is surprisingly modern in style and may be usable in practice. Joyner Jr. (1976) has shown that ordered resolution, which is known to be refutationally complete for arbitrary first-order theories, can be specialized to yield a decision method for the monadic class without equality and without second-order quantifiers. (For an overview of more recent results along this line see Fermüller, Leitsch, Tammet, et al. 1992.) Resolution can be combined with an inference rule, called paramodulation, to yield a refutationally complete inference system for first-order clauses with equality. The purpose of this paper is to show that superposition, a restricted form of paramodulation, can be equipped with specific simplification techniques such that it becomes a decision procedure for the monadic class with equality. It should be emphasized that this generalization of inference system and logic is not at all a trivial exercise. In fact it turned out to be technically more involved than we expected.

From a theoretical point of view our result is of interest as it provides evidence of the usefulness of the superposition calculus as a general method of refutational theorem proving for first-order logic with equality. Superposition refines the well-known inference rule of paramodulation in that certain additional

* The research described in this paper was supported in part by the German Science Foundation (Deutsche Forschungsgemeinschaft) under grant Ga 261/4-1, by the German Ministry for Research and Technology (Bundesministerium für Forschung und Technologie) under grant ITS 9102/ITS 9103 and by the ESPRIT Basic Research Working Group 6028 (Construction of Computational Logics).
restrictions formulated in terms of a given well-founded ordering on terms and literals, are imposed on the premises of a respective inference. Based on earlier work (e.g., Bachmair and Ganzinger 1990) the superposition calculus as we use it here has been proposed and proven refutationally complete (for arbitrary first-order clauses) by Bachmair and Ganzinger (1991). It represents an improvement on similar results obtained by Rusinowitch (1989) (also see Rusinowitch 1991) in at least two directions. The ordering restrictions are sharpened and the inference rules are additionally parameterized by selection functions so that a larger class of search strategies can be modelled. More importantly, the calculus provides a general notion of redundancy by which, at any state of the theorem proving process, inferences can be further restricted in terms of global properties of the current database of formulas. Simplification inferences which replace formulas by simpler ones do not affect refutational completeness whenever the simplified formula renders the original formula redundant. In this paper we demonstrate that with an appropriate setting of its three main parameters (ordering, selection function, simplification inferences), the superposition calculus can be tailored to the specifics of the problem at hand.

Typically, a general-purpose inference system such as superposition does not directly yield a decision procedure when applied to a decidable class of formulas. One difficulty is that the application of an inference to premises in the given class, say the monadic class with equality, may yield a conclusion that does not fall into the same class. Another problem is that even if a suitable closure property holds, repeated application of inferences to a finite set of input formulas need not necessarily terminate. The approach we advocate here for transforming a general-purpose inference system into a decision method is ultimately based on the concept of redundancy. We first restrict inferences in a suitable way to guarantee both closure and termination properties. In the case of so-called “flat” clauses—a class of formulas that is actually slightly larger than the set of clauses obtained from monadic formulas—this leads to “flatness-preserving” superposition inferences. The appropriate choice of an ordering (and of a selection function) is a key in proving that sequences of such admissible inferences terminate. However, by disregarding inadmissible inferences we may loose completeness. To compensate, we design simplification inferences that can be used to repeatedly simplify inadmissible premises so that eventually all possible inferences are admissible.

This approach is conceptually simple and, we believe, can be applied in other contexts as well (though dealing with the equality predicate seems to require an extensive technical apparatus). On the practical side we can also implement our decision method simply by taking an implementation of the superposition calculus such as the one described in (Nivela and Nieuwenhuis 1993) and add only what is needed to specify the parameter setting. Our main practical motivation for the work described here, however, stems from the observation that the satisfiability of set constraints with “negative” subterm equality tests can be checked by translating them into the monadic class with equality (Bachmair, Ganzinger, and Waldmann 1992). Set constraints have attracted quite some interest as a means of specifying type inference and flow analysis for programming languages (Heintze and Jaffar 1991, Aiken and Wimmers 1992). Applications of the monadic class to knowledge representation languages have been described by Fermüller, Leitsch, Tammet, et al. (1992).

Although there is quite some body of work on using resolution as a decision