

Exporting and Reflecting Abstract Metamathematics

Robert L. Constable
rc@cs.cornell.edu

Department of Computer Science, Cornell University, Ithaca, NY 14853, USA

The automated reasoning community has achieved remarkable results over the past 50 years, and we see clearly how to carry on substantially further. On the way, we will circumvent various technical obstacles, and we will discover new truths. I want to suggest a way of dealing with two important technical obstacles. The suggested approach will also lead us to a deeper understanding of a fundamental mechanism of language — *reflection*. The first problem is how to safely extend the automatic capabilities of a theorem prover, and the second is how to guarantee the correctness of the prover in the first place.

In tactic-oriented provers, the programming of new refinement tactics is a safe way to extend the prover. It is safe because refinement tactics reduce new inference steps to primitive rules. While this is safe, it is increasingly expressive as tactics do more work. Moreover, not all reasoning steps proceed by such reductions, e. g. decision procedures work differently as do transformation tactics.

I propose that we prove formal metamathematical results which are useful in building and extending provers, and then invoke these results via reflection rules. Moreover, we should develop the results abstractly so that they can be exported and specialized to a number of logics. In this way the community can collectively assist in the accumulation and use of relevant metaknowledge.

In this paper, I illustrate this method by considering the simple example of a propositional calculus *decision procedure*. I sketch its development for abstract logic, definable in Nuprl; then I show how to apply this to Nuprl via its reflection rule. A similar idea is developed in the paper *Using Reflected Decision Procedures* co-authored with William Aitken and Judith Underwood.