THEOREM PROVING WITH STRUCTURE SHARING AND EFFICIENT UNIFICATION

Alberto Martelli
IEI - CNR
Via S.Maria 46, 1-56100 Pisa, Italy

Ugo Montanari
1st. di Scienze dell'Informazione, Univ. of Pisa
Corso Italia 40, 1-56100 Pisa,Italy

Efficient proof checking

The interest in automatic theorem proving, which was very high in the AI community in the late sixties, has decreased. One of the reasons was the impossibility by now of obtaining a theorem prover of wide applicability. In particular "complete" search strategies based on resolution have been heavily criticized. In fact it is felt that the crucial problem is not to have an efficient prover but to be able to communicate with it and to drive it. It is also felt that efficiency improvements cannot increase significantly the performance.

While this skepticism is partially justified in the case of automatic, theorem proving, we think that a careful complexity analysis may help in improving substantially the performance of proof checkers. We agree with Kowalski that logic formalisms are to be considered as programming languages and proof checkers as interpreters. Thus a powerful logic formalism and an efficient proof checker can allow to prove manually theorems whose automatic proof would be otherwise very far from the state of the art. As a step in this direction, in[Boyer, Moore 72] a clever technique has been developed for storing shared structures in resolution-based proof checkers. However, since the standard unification algorithm does not always take advantage of structure sharing, its performance in this setting can be extremely bad. We show in [Martel.1i,Montanari 77] a simple example where checking a proof of n steps may require 2^2 operations.

Efficient unification

Quasi linear and linear algorithms for unification have been recently proposed (see for instance [Martelli, Montanari 76]). The quasi linear algorithm uses counters to discover when a variable is ready to be bound and is particularly appealing in practice, since it is only slightly more complicated than the standard algorithm and needs only the usual, top-down, LISP-like data structures. However, the time spent for initializations might be quite heavy, especially for small cases, and it could be reduced only through a close integration , of the unification algorithm in the whole theorem prover.

In [Martelli, Montanari 77] we thus suggest to merge our quasi linear unification algorithm with Boyer and Moore technique. While the simple juxtaposition of the two methods reduces the worst case above to 2^n, we propose in addition to modify them in order to make explicit further sharing and to eliminate non accessible variables.

Improving Boyer and Moore technique

We suggest three modifications to Boyer and Moore's storing technique for the purpose of adapting to it our efficient unification algorithm [Martelli, Montanari 77"]:

i) counter contents are stored together with variable bindings to avoid initializations;

ii) bindings are sometimes rewritten to increase sharing for making later unifications faster;

iii) bindings of non accessible variables are detected and not stored, to keep the data base smaller.

Besides avoiding initializations, counter stojring has the alternative advantage of allowing to visit, during unification, only the literals actually resolved in the proof. The standard technique does not use counters, but would need them anyway if it had both to detect non accessible variables and to restrict the visited literals as above.

With respect to point ii), we actually have a storage-time tradeoff. In fact in Boyer and Moore's technique every variable in some theorem T is either free or bound exactly once in T or in some theorem derived earlier, during the proof of T. Here instead we may assign a variable many times, to express structure matchings which take place, or become apparent, during the proof.

The time saving produced by those extra bindings can be exponential with the length of the proof. Furthermore we feel that the sharing expressed by such bindings is always potentially useful, and suggestive for the structure of the theorem proved.

References

Boyer, R.S. and Moore, J.S., The sharing of structure in theorem-proving programs, Machine Intelligence 7, B. Meltzer and D. Michie Ceds.l , Ecinburgh University Press, 1972, 101-116.

Martelli A. and Montanari U., Unification in linear time and space. A structured presentation, Internal Report B76-16, _I tituto di *Elabor*azione dell'Informazione, Pisa, July 1976.

Martelli A. and Montanari U., Theorem proving with structure sharing and efficient unification, Tnternal Report S-77-7, Istituto di Scienze dell'Informazione, University of Pisa, February 1977.