Proof Transformation and Expansion with a Parameterizable Inference Machine

Christoph Benzmüller **Andreas Meier** **Martin Pollet** **Volker Sorge**
Fachbereich Informatik, Universität des Saarlandes, 66041 Saarbrücken, Germany
{chris|ameier|pollet|sorge}@ags.uni-sb.de

1 Motivation

Tactical theorem provers or proof planners, such as CLAM [4], HOL[3], or ΩMEGA [2] provide proof tactics and/or proof methods to support abstract level proof construction and to improve the communication with the user. Tactics and methods typically abbreviate frequently recurring low level proof patterns. A system like ΩMEGA [2] is capable of an explicit maintenance of abstract level proof objects and support a stepwise expansion of them in pure calculus level proofs while in other systems, e.g. HOL, the application of a tactic immediately introduces a calculus level derivation. The common problem is to describe and operationalize the mapping of the abstract level inferences in terms of a sequence of less abstract inferences. A similar problem arises when different deduction systems are integrated and complex inference steps of one system have to be expressed in terms of inferences of the other system. In ΩMEGA this expansion or transformation knowledge is currently encoded in form of precise programs associated with each tactic/method. This approach clearly suffers on the problem that the programs have to be adapted whenever one of the inference systems is modified – even in case of minor modifications like the permutation of the premise order in one single inference rule.

In order to overcome these problems we propose to use a parameterizable inference machine as proof expansion and transformation mechanism. The parameters are the set of target inference rules whose composition suffices to justify the source inference. Thus the execution of the transformation/expansion program is replaced by controlled proof search over a restricted set of target inferences. Thereby we gain a more flexible and stable transformation/expansion mechanism as well as shorter and better maintainable specifications. As a concrete parameterizable inference machine for ΩMEGA we suggest to employ the agent based inference mechanism Ω-ANTS.

2 Expansion/Transformation in ΩMEGA

We briefly sketch some examples for proof transformation and expansion in the context of ΩMEGA:

E1: The ΩMEGA tactic modus tollens (mt): The expansion of mt is precisely defined in program P employing the tactic contrapos (cp) and the rule modus ponens (mp) in the following way:

$$\frac{a \Rightarrow b \quad \neg b}{\neg a}\ mt \quad \xrightarrow{\text{P}} \quad \frac{\dfrac{a \Rightarrow b}{\neg b \Rightarrow \neg a}\ cp \qquad \neg b}{\neg a}\ mp$$

Tactic cp is associated with further expansion information (where rule mp, for instance, is employed again) such that in a stepwise fashion ΩMEGAs pure natural deduction calculus level is reached in the end. Modifying the order of the premises in rule mp, for instance, unfortunately requires superfluous adaptations of the expansion programs of cp and mt, as they are sensitive wrt. to respective information. mt is clearly a very trivial example for a proof tactic and many far more complicate proof methods are provided by ΩMEGA whose adaptations to modifications can thus be extremely laborious.

E2: Transformation of TPS to ΩMEGA: ΩMEGA can call TPS to close a goal automatically. To support a transparent and stepwise proof transformation from the natural deduction calculus used in TPS to the variant used in ΩMEGA each TPS rule is mirrored as a tactic in ΩMEGA. The mirrored tactics are associated as sketched above with respective expansion programs mapping them into ΩMEGA derivations. Some of the transformation programs simply map the mirrored TPS rule to a corresponding ΩMEGA rule or tactic, whereas others consists in more complicate constructions. Hence the problem sketched above also applies to this proof transformation scenario.

E3: The ΩMEGA method Assertion-Appl: Assertion-Appl is a method used in ΩMEGA's proof planning. It closes a goal by the application of a theorem. When the method is applied, the theorem is represented as a tree structure. This representation provides a very simple possibility to determine if and how the theorem can be applied and what the new goals are. However, the expansion of an assertion application cannot be expressed schematically but depends both on the structure of the theorem and its particular application. In particular, the same theorem can be used in very different ways in an assertion application. For example, the definition of subset $\forall S_1 . \forall S_2 . S_1 \subset S_2 \equiv \forall x . x \in S_1 \Rightarrow x \in S_2$ could be employed in various ways, for instance as:

$$\frac{a \in U \quad U \subset F}{a \in F}\ \subset\!Def \qquad \frac{a \notin F \quad U \subset F}{a \notin U}\ \subset\!Def$$

Although the inference steps needed during the expansion are simple and limited, the actual expansion program is quite complicated.

3 Use of a Parameterizable Prover

Situations as explained in the last section occur often during the expansions/transformations in ΩMEGA: the needed mappings are quite easy and consist only of a restricted number of inference applications. But the mappings cannot be written schematically since they depend on the particular structures of the involved formulas. Therefore, the programs realizing those mappings are quite complicated, often employ large case distinctions over the formula structures, and contain very detailed information.

Instead of fixed and detailed programs, we propose to employ proof search in restricted calls to a parameterized inference machine PARINF. For instance, the transformation program P associated with mt in example E1 can be replaced by a call PARINF(R), where R=$\{cp,mp\}$ specifies the set of allowed inference rules. Expanding mt is thus handled as a proof search problem in PARINF with just these two inference rules $\{cp,mp\}$ and creates the respective expansion derivation on the fly. Thereby the assumptions are the original premises and the proof goal is the original conclusion. The difference to the situation before is that only crucial and adequate expansion information is provided in the sense of "Construct a derivation over inference rules mt and mp". More fine-grained information like the order of premises, structures of formulas, etc. are not relevant anymore but automatically handled within the parameterized inference machine. Changing the premise order in mp, for instance, does not anymore require the adaptation of the expansion information of mt.

At first sight the approach looks quite general, however, its practical applications are clearly limited by the power and possibilities of PARINF. As we cannot expect that a generic parameterisable inference machine is very powerful it seems to be useful that additional control and context information given in the source inference can be employed. This can even be crucial as the following example illustrates: ΩMEGA's tactic =subst* gets as additional parameter a position list $pl = (p_1, \ldots, p_n)$. An application of this tactic, =subst*(pl), realizes the simultaneous application of a premise equation $l = t$ to a premise term A at the positions $p_1, \ldots, p_n$. Its expansion is defined in terms of a recursive program employing =subst* (now with additional parameter list $pl' = rest(pl)$) and the one step equality substitution tactic[1] =subst with the additional single position parameter $p = first(pl)$. The expansion program terminates after n recursive calls. When simply replacing the recursive expansion program of =subst* by a call PARINF({=subst,=subst*}) it is however not guaranteed that the expansion will ever terminate. This is because =subst* could be employed in the original way again at target inference level. The problem is that apart from the superfluous details some important control information in the recursive expansion program has been abstracted away as well.

Motivated by this example we extend our approach. Instead of being parameterizable over the set of target inference rules only, the inference machine PARINF now accepts and employs also further control information. An expansion mapping for =subst* with parameter list pl for $n \geq 1$ thus would be PARINF({=subst with $first(pl)$,=subst* with $rest(pl)$}), i.e. we require that particular parameter instantiations can be described schematically in the expansion mappings in dependence of available (schematic) context information. In particular expansion situations these schematic parameters will be determined and thus guide the search in PARINF.

As a candidate for a parameterizable inference machine in this sense we propose the agent based inference mechanism Ω-ANTS [1]. Ω-ANTS associates each proof rule/tactic with a set of parameter agents looking for suitable parameter instantiations in a given proof context. These parameter agents cooperate by exchanging results via the Ω-ANTS' suggestion blackboards and thereby finally produce a complete set of instantiation suggestions for each rule/tactic. One of the characteristics of the Ω-ANTS blackboard mechanism is that the agents computations can be restricted wrt. a priori selected instantiation constraints by respectively initializing the suggestion blackboards. Hence, Ω-ANTS can handle the =subst* example.

4 Conclusion

We suggest a proof expansion/transformation approach that can potentially reduce the adaptation overhead of recent approaches by avoiding irrelevant detail information typically contained in expansion programs. The new proof expansion and transformation mappings are also better understandable and user friendly. Most of the tactic and method expansions in ΩMEGA are quite simple and we currently suppose that about 90% of them could already be done with Ω-ANTS as parameterizable inference machine. Currently we investigate this question more closely and look for solutions to more challenging expansion szenarios.

References

[1] C. Benzmüller and V. Sorge. Ω-ANTS – an open approach at combining interactive and automated theorem proving. In *Proc. of the Calculemus Symposium 2000*. A.K.Peters.

[2] C. Benzmüller *et al.* ΩMEGA: Towards a mathematical assistant. In *Proc. of the CADE-14*, pages 252–255, 1997.

[3] M. Gordon. HOL: a machine oriented formulation of higher-order logic. Technical Report 68, Univ. of Cambridge, July 1985.

[4] F. van Harmelen *et al.* The clam proof planner, user manual and programmers manual. Technical report, University of Edinburgh, 1993.

[1] One step equality substitution $\bar{=}$subst itself expands over Leibniz equality.