Agent-based Proof Search with Indexed Formulas

Malte Hübner[1], Serge Autexier[2], and Christoph Benzmüller[1]

[1] Fachbereich Informatik, Universität des Saarlandes
D-66041 Saarbrücken, Germany
[2] German Research Center for Artificial Intelligence (DFKI)
66123 Saarbrücken, Germany

Abstract. This paper describes work aimed at integrating ΩMEGA's agent-based suggestion mechanism Ω-ANTS and the new theorem proving framework developed by Autexier [Aut01].

1 Introduction

Problems inherent to fully automated theorem provers (such as combinatorial explosion, etc.) have led to growing interest in interactive theorem provers (ITPs). ITPs typically allow the user to construct a proof by subsequently applying inference rules. Examples for inference rules in the ITP ΩMEGA [BCF+97] are the basic rules $\wedge_I$ (conjunction introduction) and $\forall_E$ (universal quantifier elimination) of ΩMEGA's higher-order natural deduction calculus. Other rules are the trivial proof tactic *modus tollens* and the *diagonalization* proof method. The purpose of tactics and methods in ITPs is to abbreviate deductions at the base calculus level. Hence, shorter and more elegant proofs become feasible.

The drawback, however, is the broadening of the range of inference steps which have to be considered at each proof state. With a large number of inference rules available to the user interactive proof search soon becomes intractable, indicating the need for a mechanism which supports the user by suggesting applicable inference rule instances.

For this purpose suggestion mechanisms such as the agent-based Ω-ANTS system [BS00] have been developed. Ω-ANTS differs from related approaches in the following aspects:

- It distributes the computations required to search for parameter instances and to check side conditions to a very fine grained layer of agents.
- It sorts applicable inference rules heuristically and suggests the most promising rules to the user.
- Due to the distribution aspect, Ω-ANTS can handle non-decidable and complex conditions (i.e. it may employ full higher order unification) within its computations without having to restrict them a priori (e.g. by choosing a maximal depth for unification in order to guarantee termination).

- Ω-ANTS supports a uniform modeling of the different kinds of inference rules mentioned above and provides a declarative specification language for this purpose.
- Ω-ANTS can automatically search for proofs by applying inference rules automatically in the order suggested by the heuristics used.

However, even with intelligent support from such suggestion mechanisms, conducting complex proofs in ITPs is still a challenging task. One reason for this is that the calculi employed in most ITPs are aiming at but still do not sufficiently support a human oriented style of reasoning. They have not been designed for practical reasoning but are rather a result of proof theoretic research. Even natural deduction and Gentzen style calculi are unfortunately not human oriented enough to qualify as an ideal basis of an ITP. Despite the availability of powerful proof tactics and methods in ΩMEGA, many trivial, calculus intrinsic proof steps have to be employed which would never occur in human created textbook proofs. An example in ΩMEGA is an explicit decomposition of an assertion to be proved in the essential sub-tasks to be considered. Humans immediately focus on the relevant sub-tasks without employing lengthy but mainly trivial deductions to unwrap the relevant subtasks. See [BMM+01] for further discussion of these aspects in the context of proof planning in ΩMEGA.

Autexier [Aut01] recently proposed a new theorem proving environment that combines ideas of Andrews [?] and Wallen [Wal90] to provide a much better suited basis for ITPs and proof planners like ΩMEGA. The main features of this calculus are:

- It supports focusing on subformulas whereby the proof contexts for these foci in the form of available transformation rules is dynamically computed by the system in the background. It thus replaces the stepwise unwrapping of subtasks in a proof problem by foci relocations and hides calculus dependent aspects within the dynamic computations of the proof contexts. (Here, the context of a subformula is computed from an indexed formula tree as defined in [Wal90]; however, see [?] for a slightly different notion of context.)
- It supports a variety of logics in a uniform way.

The outline of the project described in this paper can be divided into three subtasks. Firstly, we plan to adapt the Ω-ANTS suggestion mechanism to the new proof-planning framework in order to support interactive proof search by suggesting applicable inference steps and focus relocations.

We will then set out to analyze how complete automated proof search can be realized in the new calculus. In the third step we will transfer the theoretical results obtained in the previous step into a realization of automated proof search based on Ω-ANTS.

This paper is outlined as follows: First, we informally describe the new framework advocated by Autexier (Sec. 2). We then briefly sketch first ideas on how to proceed on the theoretical considerations (Sec. 3). Section 4 identifies some of the problems that might arise when trying to adapt Ω-ANTS to the new framework. The last section then describes how automated proof search can be realized with the results gained from the work described in the previous sections.

2 Autexier's new framework

The framework suggested by Autexier [Aut01] aims at providing a more user-friendly environment for conducting proof-search interactively. It differs from classical calculi in that there is no fixed set of inference rules. Instead, a set of transformation rules of the form

$$[\Phi]\ u \to v$$

is dynamically generated for each proof context. Rules of this form allow one to replace a subformula or subterm u of the goal formula by another subexpression v, such that the overall formula is transformed into a refined formula. Φ is an application condition that has to be met in order for the rule to be applicable. In other words, applying a formula of the above form introduces the additional formulas in Φ as new subgoals. In this framework, a formula is proven to be a tautology if it can be transformed to $\top$ by a sequence of transformation rule applications and all conditions are validated.

Proof search basically proceeds as follows: By focusing on a subformula a proof context is selected and a set of transformation rules for that focus is generated dynamically from this context. The user can then apply these inference rules in order to transform the subformula in the current focus.

Instead of giving a formal definition of the framework we will illustrate the basic concept by an example. This also helps to identify the problems posed to a suggestion mechanism and an automated proof search. As an example we give a proof of the theorem: $\forall XY.X \subseteq X \cup Y$

Example:

$$\begin{array}{rl} & \forall XY.X \subseteq X \cup Y \\ \textit{Step 1: Def.}\subseteq\textit{Expand} & \Big\downarrow \\ & \forall e.e \in X \Rightarrow e \in X \cup Y \\ \textit{Step 1: Def.}\cup\textit{Expand} & \Big\downarrow \\ & \forall e.e \in X \Rightarrow \mathbf{e} \in \mathbf{X} \vee e \in Y \\ \textit{Focus to right } e \in X & \Big\downarrow \quad \textit{Rule generated:} e \in X \to \top(1) \\ & e \in X \\ \textit{Apply } e \in X \to \top & \Big\downarrow \\ & \top \end{array}$$

This proof reads as follows: In the first two steps the definitions of $\subseteq$ and $\cup$ are expanded. In the third step, the focus is put on the subformula $e \in X$ of the succedent. This yields the transformation rule $e \in X \to \top$, which is generated from the context of the focus, i.e. the antecedent of the implication. This rule is then applied to the subformula in the focus, which can be rewritten to $\top$. Overall, the sequence of foci choices and rule applications shown above transforms the initial formula to the equivalent formula $\forall e.e \in X \Rightarrow \top \vee e \in Y$ which holds trivially and can be shown by simplification.

This example already points at two key issues that have to be accounted for in every proof search mechanism for this calculus. First, it can be seen that the choice of placing the focus is crucial. Had we put the focus on the subformula $e \in Y$ instead of $e \in X$ the transformation rule (1) could not have been applied. Of course, in automated proof search this problem could be overcome by performing a breadth-first search over all possible choices for focusing. However, this would cause far too much branching during the proof search to be a practical solution.

A second problem lies in the selection of the right inference rule to apply. In the above example, this was trivial as there was only one rule generated from the context. However, with more complex theorems to prove, the number of available transformation rules may become significantly larger. This is partly due to the fact that in general larger formulas lead to dynamic generation of many transformation rules. Furthermore, for many mathematical domains background knowledge such as axioms and proof-methods is available in form of transformation rules as well. For instance, in the above example we implicitly used the rule $X \subseteq Y \rightarrow \forall e.e \in x \Rightarrow e \in y$ which expresses the definition of $\subseteq$.

3 Proof-Theoretic Considerations

We believe that the framework provided by Autexier is more convenient for practical reasoning and interactive proof search than classical calculi (for instance, it releases the user from the burden of unwrapping all the hypothesis and places no constraints on the sequence of quantifier eliminations). However, it is not yet clear whether this framework is also beneficial for automatic proof-search. In particular, a proof search for this framework not only has to apply the inference rules in a systematic fashion, but also to systematically place the focus on the subformulas of the overall theorem. To make things even more complicated, the problems of foci-choice and rule applications are interdependent. That is, the rules available to transform a formula depends on the chosen focus and vice versa. Another issue that has to be accounted for is backtracking.

However, although the above considerations let an automated proof search appear to be difficult it is our strong belief that this calculus allows for employment of strong strategies and heuristics which help to drastically reduce the search space and guide the proof search.

Invention of strong strategies should be possible because for each subformula of a given goal, the information necessary to transform this expression can be extracted immediately from the indexed formula tree of the goal. In other calculi, like the ND and sequent calculi the information in the hypothesis has to be extracted explicitly in order to be able to close a subgoal. Our general claim is that in the framework provided by Autexier, more information is available in every proof state. A challenge is thus posited by the task to exploit this information in order to guide the search either by strategies or heuristics.

There are basically two approaches to find heuristics and strategies as the above. A practical approach would be to try to identify strategies used by humans who work with the system in the interactive mode. One then had to evaluate

whether it is possible to formalize such strategies in order to incorporate them in the system.

Another, more theoretical approach is to look into strategies that are used in related calculi and to examine whether these strategies could be amended to apply them in the new framework. A candidate for related work is Sieg's intercalation calculus [**?**] and the NIC-calculus [Byr99]. Byrnes [Byr99] for instance, describes the *extraction strategy* which allows extraction of hypothesis only in the case they might be needed for finding a proof. Hence, extraction of irrelevant hypothesis is avoided, thereby pruning the search space. Our aim is to analyze such proof strategies for human-oriented calculi, to transfer and, if possible, improve those aspects relevant to our framework.

4 Supporting interactive proof search

Section 2 has shown that in every proof state two choices have to be made:

1. **Focus-Placement:** It has to be decided whether the current focus should be changed, that is, the user has to decide whether he wants to expand, narrow or even shift the focus.
2. **Rule-Selection:** Focusing on a subformula leads to generation of a set of transformation rules from the context. Here, the user has to check which of the rules are applicable and then select one of the applicable rules. This choice is complicated since additional rules, expressing background-knowledge like axioms or definitions have to be considered as well.

Interactive proof search could therefore benefit enormously from a suggestion mechanism that:

1. Indicates the need to change the focus–including a suggestion of how to change the focus,
2. Checks for applicable transformation rules and suggests the most promising rules to the user,
3. Explains dynamically generated inference rules to the user. This is an important aspect, because for complex proofs it might be difficult for the user to reconstruct how the rules have been generated from the context.

An agent based suggestion mechanism Ω-ANTS which supports the user in this way has been developed by Benzmüller and Sorge [BS00]. Their Ω-ANTS system is currently implemented in the ΩMEGA system, which supports interactive proof search in a variant of the natural deduction calculus. The key idea underlying Ω-ANTS is that for each of the available inference rules of the form

$$\frac{p_1 \ldots p_n}{c}\ name\ (t_1, \ldots, t_n)$$

a society of agents concurrently checks for instantiation of the parameters $c, p_1, \ldots p_n$. If at least one such parameter can be found the rule is considered to be applicable. Furthermore, a society of function agents computes possible instantiations for

$t_1, \ldots t_n$. Applicable rules are then sorted heuristically and the most promising rules are suggested to the user for application.

One main advantage of Ω-ANTS is the concurrent check for applicability of inference rules. Often, the resources (e.g. time) required to search successfully for instances of parameters might vary between rules. Even worse, checking for applicability of a rule might involve an undecidable problem, such as higher-order unification and hence, the applicability check for that rule might never terminate. Were the rules sequentially checked for application this could lead to a situation where some rules may never be considered. However, due to the concurrent treatment of rules in Ω-ANTS, search for parameter instances of a particular rule does not affect the search for other rules. Or, to be more precise, if a process that checks the applicability of one rule gets stuck with an undecidable problem, computation of parameter instantiations for other rules proceed nonetheless.

Furthermore, although Ω-ANTS is implemented within the ΩMEGA system, it is in principle a calculus independent suggestion mechanism.

In order to support interactive proof search in the new framework and to benefit from a concurrent suggestion-mechanism we intend to adapt Ω-ANTS to the new framework. The problems to be tackled during this enterprise are twofold:

1. Unlike ΩMEGA, where all inference rules and tactics are known a priori, only a fixed set of tactics is used in the new framework. Basic calculus rules do not exist, but are subsumed by the dynamic generation of transformation rules from the context.
2. Although tactics can still be defined in the common form $\frac{p_1 \ldots p_n}{c}\, name\ (t_1, \ldots, t_n)$, they have to be translated into transformation rules, employed by the new framework, in order to be applicable.

However, as we will argue below, neither of the problems is substantial.

4.1 Handling dynamically generated inference rules

Ω-ANTS is designed to support proof search for a fixed set of inference rules in the following way: The architecture is divided into three levels. At the lowest level, a society of *argument agents* is associated with each command (i.e. an inference rule or tactic). Agents of one society cooperatively compute instantiations of the parameters for the corresponding command. Communication between agents of one society is done via so called *command blackboards* on which the agents write partially instantiated commands.

Each command blackboard is monitored by a command agent, which sorts the entries on his blackboard according to a simple heuristic (e.g. preference is given to suggestions that are most complete). The most promising suggestions on each blackboard are then passed on to by the command agents to the *suggestion blackboard* which in turn is monitored by a *suggestion agent.*

The suggestion agent monitors the suggestion blackboard in order to select the most promising entries on the blackboard to the user. Suggestion agent and

suggestion blackboard thus constitute the highest level in the architecture. While argument agents for each command have to specified in a Lisp-like declarative language, the other agents together with the corresponding blackboards are built automatically for each new proof attempt.

Since for every command a set of argument agents has to be specified a priori, the system cannot deal with dynamically changing rules without being modified.

To handle dynamically generated transformation rules we intend to specify a generic set of argument agents. For each dynamically generated inference rule an instance of this set together with a corresponding command blackboard can be generated and linked to the architecture. When the focus is shifted and the set of transformation rules is dynamically adapted, this dynamically generated part of the architecture is deleted and new agent societies for the new transformations will be created. Hence, the part of the architecture that corresponds to the dynamic transformation rules will be permanently changing.

Problems can only arise if the fraction of the average number of dynamically generated rules becomes too large in relation to the number of tactics used. In this case our approach has to be reconsidered. An alternative solution would be to have one society of argument agent search for parameter instances sequentially. Part of the work proposed will be to compare pros and cons for both approaches.

4.2 Generation of Transformation Rules from Tactics

Adapting Ω-ANTS to compute suggestions for rule application of the form $[\Phi]\ u \rightarrow v$ from a pre-specified tactic can probably be done straightforwardly.

In ΩMEGA s ND-calculus, tactics are of the form $\dfrac{p_1 \ldots p_n}{c}\ Name\ (t_1, \ldots, t_n)$. The Ω-ANTS mechanism searches for instantiations for each of the $c, p_1, \ldots, p_n$. A rule is applied in *backward direction*[1], by substituting an open line c by the open lines $p_{i_1}, \ldots, p_{i_l}$ $(l \leq n)$, the subset of premises for which no instantiation could be found.

While in ΩMEGA the search for instantiations has to be performed in the support-lines of the open goal, the new framework requires searching for instantiations amongst certain subformulas of the overall formula.

We therefore suggest that tactics should still be specified in the common form, i.e. as a set of premises and a set of conclusions. Ω-ANTS will be adapted to search for parameter instantiations amongst the subformulas of the overall formula. If $c, p_{i_1}, \ldots, p_{i_m}$ is the set of parameters for which instantiations could be found, the backward application of a tactic can then be emulated by applying a transformation rule of the form:

$$[p_{i_1}, \ldots, p_{i_m}]\ c\ \rightarrow p_{i_{m+1}} \wedge \ldots \wedge p_{i_n}$$

This would also allow us to use heuristics specified in Ω-ANTS which favor those rules, which introduce the least open subgoals $p_{i_{m+1}}, \ldots, p_{i_n}$.

[1] Forward and sidewards-application of the rules is also possible and can be treated in a similar way.

5 Automating Proof Search

To be able to search for proofs automatically we plan to combine the work sketched in section 3 and 4. In particular, we intend to incorporate the Ω-ANTS mechanism to realize automatic proof search in the new framework.

Benzmüller and Sorge [BS00] have shown that this can be done relatively easy. For each open node of the search tree Ω-ANTS computes all applicable inference rules. The rule considered to be most promising by ΩMEGA is used to expand that node. The other applicable inference rules are stored for backtracking. Hence, in automation mode, Ω-ANTS simulates a user who always applies the most promising inference rule. If it turns out that this choice made it impossible to close the corresponding branch of the proof tree Ω-ANTS backtracks by applying the inference rule with the next highest ranking.

The issue of completeness has not been sufficiently discussed by Benzmüller and Sorge so far and will be addressed in our future work. We will first attempt to develop strategies that yield a complete search procedure for first order logic. Thereafter, we will set out to investigate whether this work can be extended to realize a complete proof search for higher-order logic with respect to Henkin-Semantics.

We plan to implement automated proof search for the new framework similar to [BS00]. The strategies mentioned in section 3 should be used to restrict the search in a way to explore only those parts of the search space, that are necessary to find a proof. Heuristic knowledge provided by the Ω-ANTS mechanism in the way described above hopefully helps to apply the 'right' inferences before less promising rules are considered.

Furthermore, on top of the heuristics and strategies described above, we intend to employ explicit meta-knowledge as is used in proof-planners. For instance, the proof planner MULTI [MM99] uses meta-knowledge to select a subset of the available methods which are appropriate for a given domain. Meta-knowledge is then also used to suggest a partial order for the application of the selected rules.

We aim to incorporate such Meta-knowledge into the Ω-ANTS framework. Hence, in addition to the general and mainly domain-independent criteria employed so far, Ω-ANTS will use domain-dependent Meta knowledge to heuristically sort applicable inference rules. Ideally, the automation of Ω-ANTS will result in a full proof-planner, extending the complete proof search on base calculus level. Our hope is that we will thus be able to make a step towards overcoming the problem of brittleness in proof planning.

References

[Aut01] Serge Autexier. A proof-planning framework with explicit abstractions based on indexed formulas. In Maria Paola Bonacina and Bernhard Gramlich, editors, *Electronic Notes in Theoretical Computer Science*, volume 58. Elsevier Science Publishers, 2001.

[BCF+97] C. Benzmüller, L. Cheikhrouhou, D. Fehrer, A. Fiedler, X. Huang, M. Kerber, M. Kohlhase, K. Konrad, E. Melis, A. Meier, W. Schaarschmidt, J. Siekmann, and V. Sorge. ΩMega: Towards a Mathematical Assistant. In W. McCune, editor, *Proceedings of the 14th Conference on Automated Deduction (CADE–14)*, LNAI, Townsville, Australia, 1997. Springer Verlag, Berlin, Germany.

[BMM+01] Christoph Benzmüller, Andreas Meier, Erica Melis, Martin Pollet, and Volker Sorge. Proof planning: A fresh start? In *Proceedings of the IJCAR 2001 Workshop: Future Directions in Automated Reasoning*, Siena, Italy, 2001.

[BS00] Christoph Benzmüller and Volker Sorge. Ω-ANTS – an open approach at combining interactive and automated theorem proving. In M. Kerber and M. Kohlhase, editors, *Proceedings of the Calculemus Symposium 2000*, St. Andrews, United Kingdom, 6–7 August 2000. AK Peters, New York, NY, USA. forthcoming.

[Byr99] John Byrnes. *Proof Search and Normal Forms in Natural Deduction.* PhD thesis, Carnegie Mellon University, 1999.

[MM99] Erica Melis and Andreas Meier. Proof planning with multiple strategies. Seki Report SR-99-06, Fachbereich 14 Informatik, Universität des Saarlandes, Saarbrücken, Germany, 1999.

[Wal90] Lincoln A. Wallen. *Automated Proof Search in Non-Classical Logics. Efficient Matrix Proof Methods for Modal and Intuitionistic Logics.* MIT Press, Cambridge, Massachusetts;London, England, 1990.