Siekmann / Benzmüller / Melis

Project MI 4:

OMEGA:
Resource-adaptive Proof Planning

3.1 Overview

Funding period:	1 January 2002 – 31 December 2004
Period covered by report:	1 February 2001 – 31 January 2004
Project directors:	Prof. Dr. Siekmann, Jörg, Dr. Benzmüller, Christoph, PD Dr. Melis, Erica
Work address:	Universität des Saarlandes Fachbereich Informatik Postfach 15 11 50 66041 Saarbrücken
Phone numbers:	(0681) 302 – 5275 / 4574 / 4629
Fax number:	(0681) 302 – 5076
Email addresses:	siekmann@ags.uni-sb.de chris@ags.uni-sb.de melis@dfki.de
Researchers:	Dr. Christoph Benzmüller (Grundausstattung) Dr. Andreas Meier, 1/02 – 10/03 Dipl. Math. Martin Pollet, 11/03 – current Dr. Quoc Bao Vo, 8/02 – current PD Dr. Helmut Horacek (Grundausstattung) PD Dr. Erica Melis (DFKI) Dr. Claus-Peter Wirth, 03/02 – 09/02

3.2 State of the Art at the Time of the Proposal

Modularization and decentralization are aspects that increasingly influenced the design of modern mathematical assistance environments in the late 90's and since then this trend gained on impact. The OMEGA group has been a driving force of this move.

Several research projects and international networks have been initiated during this period

that concentrate on frameworks and infrastructure for the integration of heterogeneous reasoning tools (e.g. CALCULEMUS[1] and PROSPER[2]), the vision of a mathematical semantic web (e.g. MONET[3] and MOWGLI[4]) and the transition from pen and paper-based mathematical practice to modern computer-based environments (e.g. MKM[5]). Most research teams who develop mathematical assistant systems follow a bottom-up approach: system integration or the development of the user interface usually start with the functionalities of the existing system.

At the time of our original proposal a majority of proof assistant systems constructed explicit proof objects, where representation formats range from the 'proofs as types' paradigm to more ad hoc representations. The proof data structure of the ΩMEGA system is not only one of the first, but also one of the most elaborate proof-object representations as it maintains the proof at different levels of granularity and abstraction.

Proof planning as a cognitively motivated, abstract approach to mathematical theorem proving is investigated mainly in Edinburgh, Birmingham, and Saarbrücken. With its strengths on domain-specific meta-reasoning and on the exploitation of proof planning functionalities to guide the integration of external reasoning systems, the OMEGA group has taken the leadership in this small, albeit important subarea and demonstrated that proof planning offers a suitable platform for the integration of mathematical systems into the process of computer-supported theorem proving. The external components can be used either inside of a method (for example, a computer algebra system for simplification) or for the instantiation of meta-variables (for example, the constraint solver $\mathcal{CoSIE}$ (Zimmer & Melis, 2004) to solve inequalities over real numbers). However, improving the flexibility of meta-reasoning was still a key research issue at the time of the proposal. And both the group of Alan Bundy in Edinburgh and our group analyzed shortcomings and problems of proof planning, e.g., its brittleness and the dependency of proof planning on the rigid influence of the underlying logic layer (Bundy, 2002; Benzmüller, Meier, Melis, Pollet, & Sorge, 2001a).

As an alternative to deliberative proof planning, reactive and agent-based theorem proving gained interest in the proof assistant community and agent-based theorem proving has also been investigated since the early 90's for traditional automated theorem provers.

[1] http://www.calculemus.net
[2] http://www.dcs.gla.ac.uk/prosper
[3] http://monet.nag.co.uk
[4] http://www.mowgli.cs.unibo.it
[5] http://monet.nag.co.uk/mkm

Learning techniques were used to acquire control knowledge in the context of equational and superposition-based theorem proving (Schulz, 2000; Fuchs & Fuchs, 1998; Denzinger & Schulz, 1996). The knowledge gained from the analysis of the given proofs is expressed in heuristic evaluation functions which then guide the search process for a new problem. Silver (1984) and Desimone (1987) used precondition analysis to learn new inference schemas from the pre- and postconditions of given inference steps. These inference steps are completely specified as a terminating expansion into formal proof steps of the underlying calculus, whereas methods in ΩMEGA may contain arbitrary function calls, for example, the computation of a computer algebra system.

In cooperation with the University of Birmingham, we developed a first approach to learning in proof planning (Jamnik, Kerber, & Benzmüller, 2001), which was used as a starting point for the learning of control knowledge as reported in Section 3.4.5.

3.3 Methods

The scientific method in our research in the OMEGA project is proof planning driven by empirical case studies. These case studies are used to evaluate former models and their implementation, and the analysis of case studies gives insight for the extension of a model and the development of new models. In addition, more often than not new applications give rise to new requirements which have to be integrated into the extended model. Successive extension and evaluation of models was applied in the reported period in particular with respect to the improvement of proof planning in the ΩMEGA system (Siekmann et al., 2003).

However, the result of a critical and more principled analysis of ΩMEGA's overall design showed more fundamental shortcomings of its logical basis (a higher-order natural deduction calculus so far) and we developed the CORE calculus as an alternative. On top of this new logical basis, we are building the ΩMEGA-CORE system (Hübner, Autexier, Benzmüller, & Meier, 2004). Case studies are currently carried out to evaluate the anticipated benefits of this decision. Strong evidence for the anticipated benefit of the new system comes also from the empirical data on human-constructed proofs in a tutorial context as obtained in the DIALOG project.

3.4 Results

3.4.1 Integration of Proof Planning and Agent-Based Theorem Proving (AP1)

Deliberative automated proof planning as realized in MULTI (Meier, 2004), which is ΩMEGA's new multi-strategy proof planner, originated separately from pro-active agent-based techniques for interactive theorem proving (as realized in the Ω-ANTS system (Sorge, 2001; Benzmüller & Sorge, 2000)). Because of the complementary strengths and weaknesses of both approaches, we now started their integration into a first prototypical implementation (Pollet, Melis, & Meier, 2003; Meier, Melis, & Pollet, 2004): used in standard mode, the proof planner MULTI makes all decisions at choice points automatically. When applied in the interactive mode however, the user can make these decisions and construct the proof plan interactively with MULTI. This requires detailed knowledge about the underlying proof planner which the user may not always have, so MULTI employs now Ω-ANTS as one of its strategies. Ω-ANTS agents compute suggestions for possible proof steps, which are then presented to the user for a selection of an appropriate next step. This is an improvement over the prior interaction mode of MULTI because suggestions are now generated pro-actively by the Ω-ANTS system. This is particularly valuable for interactive proof planning in a tutoring system for mathematical proofs (see (Melis, 2002; Melis & G. Goguadse, 2003; Melis, 2003)).

Furthermore we have started to unify the basic concepts underlying both approaches. Automated proof planning and interactive agent-based theorem proving used different data structures (i.e., methods and tactics) to represent the knowledge about an inference step as well as different formalisms (i.e., tasks and foci) to represent and maintain the proof plan under construction. We unified methods and tactics into a common data structure, and we defined tasks as a uniform representation for these new methods. Formerly, tasks and foci were generated from the actual proof and from annotations of the proof, and a method or tactic application acted on the proof itself. Now, the new task level exists as an abstraction separately from the proof level and the application of methods acts on tasks, that is, they add new tasks or remove tasks. The task level will serve as a common interface between various proof construction components of the ΩMEGA-CORE system and support a synergetic interplay of automated and interactive proof construction (Hübner, Benzmüller, Autexier, & Meier, 2003). The new specification of the task level is currently implemented on top of the CORE system and it is now tested in first experiments on interactive proof construction in the DIALOG project.

The CORE system (Autexier, 2001, 2003) provides a communication infrastructure that mediates between the user and the automatic reasoning procedures. It is based on a new uniform

meta proof theory for contextual reasoning and encompasses most aspects of communication from the presentation of the proof state and the supply of relevant contextual information about possible proof continuations to the support for a hierarchical proof development. The proof theory is uniform for a variety of logics and admits the extension to choice-expressions (Wirth, 2002) and human-oriented inductive theorem proving (Wirth, 2004). It exploits proof theoretic annotations in formulas for a contextual reasoning style that is as far as possible intuitive for the user while at the same time still adequate for automatic reasoning procedures. Furthermore, concepts are defined to accommodate both the use and the explicit representation of hierarchies that are inherent in problem solving in general.

3.4.2 Proof Planning (AP2)

This workpackage consists of the following independent parts:

Meta-Reasoning and Proof Planning

One of the advantageous features of proof planning is the explicit representation of control rules that encode mathematically motivated heuristics for the traversal of the search space. With control rules it is not only possible to reason about the current goals and the current assumptions, but also about the proof planning history and the proof planning context (e.g., about the theory within which the problem is stated or about the collected constraints), see (Meier, 2004).

Some important features of meta-reasoning in MULTI are the following:

1. Meta-reasoning is used to analyze failed proof attempts and to use the failure for guidance of backtracking or plan modification. For instance, we realized control rules to guide case-splits and lemma speculation, two "eureka"-steps whose necessity in a proof is difficult to spot and whose introduction is difficult to guide in general. Some theorems in the limit domain, for instance, require a case split, such as the following theorem, which states that function f is continuous at point a, if it has a derivative f' at point a, or more formally

   Theorem: $cont(f, a)$ follows from *Assumption:* $deriv(f, a) = f'$.

   When tackling this problem, MULTI reduces the theorem to the goal $|f(c_x) - f(a)| < c_\epsilon$, which could be solved by deriving $|\frac{f(M_{x_1}) - f(a)}{M_{x_1} - x} - f'| < M_{\epsilon_1}$ from the initial assumption.[6]

[6] c_x and M_y, respectively, denote the skolem constant and the meta-variable replacing the universally and existentially quantified variables x and y.

However, in this derivation the condition $|c_x - a| > 0$ has been derived as a new goal and then MULTI fails to prove this goal. The partial success, i.e., the solution of the initial goal, suggests a patch for this proof attempt, introducing a case split $|c_x - a| > 0 \vee \neg(|c_x - a| > 0)$ on the failing condition. Then, MULTI has to solve $|f(c_x) - f(a)| < c_\epsilon$ twice: once with $|c_x - a| > 0$ and once with $\neg(|c_x - a| > 0)$ as an additional condition, which is now successful.

In general, the failure and its patch follow the pattern: There is a goal G, which MULTI can solve with support nodes S under conditions *Conds*, which are introduced as new goals. Now suppose the system fails to prove one of these new goals $C_i \in$ *Conds*. This can now be used productively to introduce a case-split on the failing condition, i.e., $C_i \vee \neg C_i$, and the main goal G is proved twice: once with the hypothesis C_i and once with the hypothesis $\neg C_i$. This reasoning pattern is implemented in a strategic control rule that analyzes failing search branches, guides the backtracking, and introduces the above case split.

With a similar goal, Ireland and Bundy discuss failure analysis (Ireland & Bundy, 1995; Ireland, 1992) using the so-called *critics*. Critics in the Edinburgh system CLAM are associated with a method and capture patchable exceptions to the application of that method rather than a general modification of reasoning. This leads to a rather ad hoc addition of "patching code" for specific methods, and for that reason we realized failure reasoning in the more general formalism of control rules. These are not necessarily associated with a single method and can analyze the whole plan history, not just one application of a method.

While reasoning about failure was abstracted from limit problems, the speculation of case splits is a general meta-reasoning pattern and it is useful in other domains, too.

2. Meta-reasoning queries the constraint solver[7] if proof planning fails to compute an instantiation for a meta-variable. For instance, consider the constraints on the meta-variable M_v: $\frac{|c_x - c|}{c_\epsilon} < M_v$ and $M_v < |c_x * c|$. These constraints are consistent, but solvable only if $\frac{|c_x - c|}{c_\epsilon} < |c_x * c|$. In other words, to compute an instantiation, further constraints are necessary. The continuation of proof planning by collecting more constraints is realized in a strategic control rule.

3. Two subgoals are dependent if they share some meta-variable. For instance, suppose that there are two subgoals G and G' sharing a meta-variable M_v. If MULTI first closes G and M_v is now constrained to, say t, it may fail to solve G' with t. Should the system backtrack

[7] Constraint solver acts as an external system in proof planning (see (Zimmer & Melis, 2004)).

G' in this situation? Not necessarily, because it is also possible that backtracking on G may lead to another constraint for M_v, for which G' can be closed.
This meta-reasoning is encoded into control rules that reason about dependencies between subgoals (by shared meta-variables) and lead to different backtracking strategies.

Multiple-Strategy Proof Planning

MULTI provides a general framework for the incorporation of heterogeneous, parameterized algorithms for proof plan refinements and modifications. Currently, the following algorithms are available in MULTI's implementation:

PPlanner refines a proof plan by introducing method applications.

InstMeta refines a proof plan by instantiating meta-variables.

BackTrack modifies a proof plan by deletion of steps.

Exp refines a proof plan by expanding complex steps.

ATP refines a proof plan by solving subproblems with appropriate automated theorem provers.

CPlanner refines a proof plan by transferring steps from a source proof plan or proof fragment.

Instances of these algorithms define (different) strategies. Technically, a strategy is a condition-action pair. The condition part describes when the strategy is applicable. The action part consists of a modification or refinement algorithm and an instantiation of its parameters. Similar to the applicability of methods we separate the legal and the heuristic knowledge about the applicability of strategies. The condition part of a strategy states the legal conditions of applicability, whereas strategic control rules reason about the heuristic expected utility of the application of the strategy.

MULTI was tested in several case studies (see Section 3.4.4) for which we developed new strategies as well as new strategic meta-reasoning patterns. Among others, we developed strategies for different kinds of backtracking and different forms of meta-variable instantiation.

In some of the strategies employed by MULTI, external systems are called to provide additional functionalities. One of the external systems that now cooperates with proof planning

is the theory formation system HR (Colton, 2002) (see (Meier, Sorge, & Colton, 2002)). It instantiates meta-variables in a similar manner to that of the constraint solver $\mathcal{CoSIE}$. HR was used, for example, for an instantiation strategy in the residue class case study to prove non-isomorphism theorems: To prove that two given structures are non-isomorphic it suffices to find an invariant under isomorphism and to show that it differs for the two structures. Finding an invariant for two structures is the key step in this proof and usually involves some creativity in the problem solving process. HR is used to automatically invent suitable invariants. The instantiation strategy invoking HR introduces the invariant as an instantiation for the meta-variable. Afterwards, MULTI completes a proof plan for a non-isomorphism theorem by proving that the invariant holds for one structure, whereas it does not hold for the other. For example, consider the pairwise non-isomorphic quasi-groups S^1, S^2, S^3 whose respective multiplication tables are depicted in Figure 1. When comparing the tables of S^1 and S^2, one suitable invariant is fairly obvious: while S^1 has only $\bar{0}_5$ on the main diagonal, all elements on the main diagonal of S^2 are distinct. Thus, the property $\exists x_{\cdot} \forall y_{\cdot} x = y \circ y$ is a suitable invariant that differs for the two structures S^1 and S^2. A discriminating criterion is less obvious for the multiplication tables of S^2 and S^3. Here, one property of S^3, which does not hold for S^2, is $\forall x_{\cdot} \forall y_{\cdot} (x \circ x = y) \Rightarrow (y \circ y = x)$.

$S^1 = (\mathbb{Z}_5, \bar{-})$

S^1	$\bar{0}_5$	$\bar{1}_5$	$\bar{2}_5$	$\bar{3}_5$	$\bar{4}_5$
$\bar{0}_5$	$\bar{0}_5$	$\bar{4}_5$	$\bar{3}_5$	$\bar{2}_5$	$\bar{1}_5$
$\bar{1}_5$	$\bar{1}_5$	$\bar{0}_5$	$\bar{4}_5$	$\bar{3}_5$	$\bar{2}_5$
$\bar{2}_5$	$\bar{2}_5$	$\bar{1}_5$	$\bar{0}_5$	$\bar{4}_5$	$\bar{3}_5$
$\bar{3}_5$	$\bar{3}_5$	$\bar{2}_5$	$\bar{1}_5$	$\bar{0}_5$	$\bar{4}_5$
$\bar{4}_5$	$\bar{4}_5$	$\bar{3}_5$	$\bar{2}_5$	$\bar{1}_5$	$\bar{0}_5$

$S^2 = (\mathbb{Z}_5, \lambda xy_{\cdot}(\bar{2}_5 \bar{*} x) \bar{+} y)$

S^2	$\bar{0}_5$	$\bar{1}_5$	$\bar{2}_5$	$\bar{3}_5$	$\bar{4}_5$
$\bar{0}_5$	$\bar{0}_5$	$\bar{1}_5$	$\bar{2}_5$	$\bar{3}_5$	$\bar{4}_5$
$\bar{1}_5$	$\bar{2}_5$	$\bar{3}_5$	$\bar{4}_5$	$\bar{0}_5$	$\bar{1}_5$
$\bar{2}_5$	$\bar{4}_5$	$\bar{0}_5$	$\bar{1}_5$	$\bar{2}_5$	$\bar{3}_5$
$\bar{3}_5$	$\bar{1}_5$	$\bar{2}_5$	$\bar{3}_5$	$\bar{4}_5$	$\bar{0}_5$
$\bar{4}_5$	$\bar{3}_5$	$\bar{4}_5$	$\bar{0}_5$	$\bar{1}_5$	$\bar{2}_5$

$S^3 = (\mathbb{Z}_5, \lambda xy_{\cdot}(\bar{3}_5 \bar{*} x) \bar{+} y)$

S^3	$\bar{0}_5$	$\bar{1}_5$	$\bar{2}_5$	$\bar{3}_5$	$\bar{4}_5$
$\bar{0}_5$	$\bar{0}_5$	$\bar{1}_5$	$\bar{2}_5$	$\bar{3}_5$	$\bar{4}_5$
$\bar{1}_5$	$\bar{3}_5$	$\bar{4}_5$	$\bar{0}_5$	$\bar{1}_5$	$\bar{2}_5$
$\bar{2}_5$	$\bar{1}_5$	$\bar{2}_5$	$\bar{3}_5$	$\bar{4}_5$	$\bar{0}_5$
$\bar{3}_5$	$\bar{4}_5$	$\bar{0}_5$	$\bar{1}_5$	$\bar{2}_5$	$\bar{3}_5$
$\bar{4}_5$	$\bar{2}_5$	$\bar{3}_5$	$\bar{4}_5$	$\bar{0}_5$	$\bar{1}_5$

Figure 1. Some quasi-group multiplication tables.

We also developed and applied a *new type of method*. Since the expansion of a method is usually specified by a sequence of tactic applications to be carried out once the proof plan is completed, we needed a different kind of method, which we call critical methods. For these methods the expansion is done already within the proof planning process. Critical methods contain only minor and efficient conditions for their application and they lead to backtracking of the whole proof plan, if their expansion fails.

There are several reasons for the introduction of critical methods. The expansion of methods is now more robust, because it does not always depend on an exactly specified sequence of tactics. Critical methods make proof planning more hierarchical, especially the extension of strategies to new theorems, which contain former simpler theorems as subproblems. The resulting top-level proof plan becomes shorter. The application of critical methods distributes the resources of the proof search to the main problem and leaves simpler subproblems for later expansion.

More Abstract Proof Planning

Proof planning aims at an abstract proof object, which will be successively refined and expanded until a logic-level proof is reached. However, as our case studies showed, the proof planning level is still far too much influenced by the underlying calculus (Benzmüller, Meier, Melis, Pollet, & Sorge, 2001b). The peculiarities of the natural deduction calculus used in the ΩMEGA system, that is, the treatment of hypotheses, and the order of forward and backward steps, and the elimination and introduction of quantifiers, etc. are propagated to the planning level. This conflicts with the conceptual view that the planning level is to represent the reasoning of a working mathematician, which may influence the proof construction in the underlying logic formalism but not vice versa.

These restrictions are now overcome in CORE (Autexier, 2003), the new basic reasoning calculus engine we developed. CORE supports reasoning at the assertion-level in a contextual rewriting proof style and provides a uniform mechanism for the application of assertions that abstracts from logical details. The calculus of CORE now provides the new logic-layer below the task-layer, which is used for proof planning (see Section 3.4.1).

Proof planning methods and strategies operate on statements that are expressed in a uniform formal logical language. The representation of concepts, however, is often a key issue in mathematical textbooks. There is usually a wide variety of different representations for the same concept and the choice of a representation is often a key step in mathematical problem solving (Kerber & Pollet, 2002a, 2002b). As a first step towards a more adequate mathematical vernacular we introduced annotated constants (Pollet & Sorge, 2003) in order to encode mathematical concepts that are difficult to express in formal logic. For instance, a set $\{a_1, \ldots, a_n\}$ can be expressed in logic by a lambda term. This representation, however, results in lengthy logical lambda expressions that are difficult to analyze and process. Constants are now annotated with special data-structures comprising the mathematical information for which the

constant stands. For example, an annotated constant for the set $\{a_1, \ldots, a_n\}$ is a constant annotated with a duplicate-free ordered list containing the elements of the set. Control rules and methods access the mathematical content of annotated constants via these data-structures and operate on these annotations.

Each annotated constant is also associated with its formal definition in ΩMEGA's logic language (e.g., a lambda expression for sets), and annotated constants are later replaced by their definition during the expansion of a proof plan. The correctness is finally checked by the construction of a logic-level proof including the expanded definitions. So far, we have annotated constants for integers, variable-free finite sets, lists, and cycles of permutations. Input and pretty-printing functions enable a presentation conforming the standard notation found in a textbook.

Island Proof Planning

Island Proof Planning was used inter alia in a case study on the irrationality of $\sqrt{2}$ (Wiedijk, 2003), where it turned out to be crucial (see Section 3.4.4 for more information).

Cognitive Adequacy of Proof Planning Methods

Teaching of mathematics, in particular the teaching of mathematical proof techniques, is usually implicit in the sense that the student is shown examples (in a textbook or on the board in the class room). An interesting question is: Would students perform better if taught explicitly the tricks of the trade?

The knowledge encapsulated in the proof planning methods of a particular field contains exactly this kind of knowledge. For that and other reasons, we conducted psychological tests and experiments, involving instructions for simple ϵ-δ proofs, a typical bottleneck in education (Melis, Glasmacher, Ullrich, & Gerjets, 2001; Melis, 2003). These experiments were conducted in collaboration with the psychology department at Saarland University. They suggested a statistically significantly improved performance of the students who were instructed with proof planning methods instead of learning in a traditional way.

3.4.3 Agent-based Theorem Proving (AP3)

Encapsulating external reasoning systems, such as specialized higher-order theorem provers, model generators, and computer algebra systems, into a software agent and exploiting the Ω-ANTS blackboard mechanism of the ΩMEGA environment was evaluated on several case studies (Benzmüller, Jamnik, Kerber, & Sorge, 2001). This architecture offers a flexible way to integrate each single external system in the form of a pro-active (software) agent with focus on those parts of the problem it is designed for, without the need to specify a priori a hierarchy of calls. All agents pick up and investigate the central proof object, given in a higher-order natural deduction style augmented with additional facilities abstracted from the pure calculus. In case they are applicable in the current proof context, they carry out their task by calling the external system they encapsulate. If the external system succeeds, the job is done. Otherwise the agent consumes the available resources and returns control, for example, to make a bid in terms of a (probably) modified proof object. A bid is accepted and executed by the central system based on heuristic criteria[8] and the remaining ones are stored for backtracking purposes. The agent paradigm now overcomes many limitations of an otherwise static and hard-wired integration. Accessing external systems is orchestrated by MATHWEB.

The agent approach was also used for the task of finding suitable knowledge in a mathematical database (Benzmüller, Meier, & Sorge, 2003). The architecture comprises in that case a (very large) mathematical database specialized in storing, retrieving and administrating theorems, definitions, and theories and a theorem proving system for proof construction. The communication between the two systems is carried out by special mediator agents. A first prototype was implemented with MBASE (Franke & Kohlhase, 2000; Franke, 2003) as the mathematical database, ΩMEGA as the theorem proving system, and mediators in form of Ω-ANTS agents (Franke, Moschner, & Pollet, 2002). MBASE is a mathematical database for any kind of mathematical documents based on OMDOC (Kohlhase, 2000), an extension of the OPENMATH standard. MBASE provides a query mechanism based on the OPENMATH syntax for terms, but has no predefined semantics, i.e., the choice of the particular underlying logic and its semantics is open to the user to define.

There are currently two kinds of agents which assist the user in interactive proof construction: the first kind of agents searches for applicable theorems containing equations or equivalences for possible rewriting-steps, the second kind searches for suitable assertion applications, i.e.,

[8] For instance, bids with closed (sub)goals are preferred over partial results and big steps in the search space are preferred over calculus level steps.

theorems, definitions and lemmas involving non-equivalences and non-equations. Both agents take the current open line, and transform the formula into a query to MBASE. Different queries are sent in parallel, for example, those whose only the head symbol has to match and those whose the formula has to match except local constants. The answer from the database MBASE is filtered according to certain applicability conditions before the resulting suggestions are presented to the user.

We proposed a distributed mediator between a theorem proving system and the mathematical knowledge bases, which is independent of the particular proof and knowledge representation formats of the theorem proving system and the mathematical databases. The mediator module is again based on a multi-agent system architecture so that reasoning agents can be assigned to assertions. These agents, called *assertion agents* (Vo, Benzmüller, & Autexier, 2003; Vo, 2003), operate in parallel with the theorem proving system and apply a resolution-based algorithm to analyze the logical consequences of the assigned assertions for the proof context at hand. The assertion agents have been developed in the DIALOG (MI 3) project. In the context of the DIALOG project, there is an additional motivation, namely to support resolution of under-specification in proof step utterances.

An important advantage of the proposed mediator module is that the whole system, i.e. the theorem proving system plus the mediator plus the mathematical database, may pursue different proving strategies at the same time: While the theorem prover, for instance a proof planner, tries to apply its most promising proving strategy, e.g. refining the goal of the proof task, assertion agents might follow other strategies controlled by their own set of heuristics, e.g. a forward derivation on the premises or a simplification of a complex expression, etc. (Vo, 2004).

3.4.4 Case Studies (AP4)

Our original case study of ϵ–δ proofs was largely extended beyond the problems from Bledsoe's challenges (Bledsoe, 1990). Furthermore we proved about 60 conjectures from the core chapters in the analysis textbook Bartle and Sherbert (1982) including theorems involving limit of sequences, limit of functions, continuity of functions, and derivative of functions (Meier, 2004). The newly solved problems require meta-reasoning patterns at the planning level *and* at the strategy level, for instance, failure reasoning. This case study stimulated the development of new strategies and generic meta-reasoning as discussed in Section 3.4.2.

We also extended our case study on the classification of algebraic structures. The goal of

this case study is to classify given residue class structures (i.e., a set of residue classes and a binary operation $\circ$) in terms of their basic algebraic properties and to classify the given residue class with respect to isomorphic structures. Meanwhile we classified about $18,000$ structures, including a large set of structures with the set $\mathbb{Z}_{10}$ (Meier, Pollet, & Sorge, 2002). The integration of the automatic theory formation system HR (see Section 3.4.2) turned out to be particularly valuable for the classification of structures into isomorphism classes (Meier, Pollet, & Sorge, 2001; Meier et al., 2002).

Another case study deals with the verification of the computation of the computer algebra system (CAS) GAP (Cohen, Murray, Pollet, & Sorge, 2003). The computation consisted of eight queries for the membership and the non-membership of a permutation to a group, the orbit and the stabilizer of a group, and the order of a group. As opposed to another and earlier approach to the verification of algebraic computations, where the CAS provides the trace which is used to create a sequence of tactic applications (Kerber, Kohlhase, & Sorge, 1998), this new case study points to a more flexible coupling between theorem prover and CAS: there is no need for the tight correspondence between traces and tactics in proof planning because search is used to find a sequence of methods justifying the computation.

We also participated in Freek Wiedijk's international case study to prove the irrationality of $\sqrt{2}$ (Wiedijk, 2003). This well known theorem was used for a comparison of fifteen (mostly interactive) theorem proving systems, whose solution of the problem had to be documented and submitted to a jury. This case study is particularly significant as it represents an important shift of emphasis in the field of automated deduction away from the somehow artificial problems of the past (e.g. the TPTP contest) back to real mathematical challenges. In (Siekmann et al., 2003) we show three different solutions of the problem with ΩMEGA.

The first solution is based on the most elementary use of ΩMEGA, namely as a traditional tactical theorem prover, where the proof is constructed interactively by the application of our existing tactics, just as in say HOL (?) or ISABELLE (Paulson & Nipkow, 1990).

The second solution is still interactive, but it is based on island proof planning, where the user states the main proof islands, i.e., he freely states his proof sketch without reference to pre-defined tactics. Then, ΩMEGA proves the gaps between the islands, involving a computer algebra system (MAPLE) and two traditional automated theorem provers, namely OTTER and SPASS. The final result is a verified calculus-level proof starting from the proof sketch of the user. The island proof planning idea has stimulated the development of the task interface for proof planning and theorem proving (see (Hübner et al., 2004)).

The third solution is fully automated with MULTI. To achieve this, we implemented special but still generic methods and control rules that simulate the creative steps of the user. Because of the general concepts, MULTI can not only prove the irrationality of $\sqrt{2}$, but also other $\sqrt[j]{l}$-problems for natural numbers j and l.

Further experiments and case studies are:

- An evaluation of the learning of control knowledge is described in (Jamnik, Kerber, Pollet, & Benzmüller, 2003; Jamnik, Kerber, & Pollet, 2002a); see also Section 3.4.5.
- Examples for agent-based reasoning with external reasoning systems are presented and discussed in (Benzmüller et al., 2001).
- Assertion application in naive set theory with the ΩMEGA-CORE system is described in (Vo et al., 2003; Vo, 2003) and (Vo, 2004); this work has been carried out in collaboration with the DIALOG project.
- Diagonalization proofs (Cheikhrouhou, 1997; Cheikhrouhou & Siekmann, 1998).

3.4.5 Learning of Control Knowledge (AP5)

Learning of control knowledge amounts to learning the skills of how to prove a new theorem given several related proofs (and possibly failed proof attempts). The possible sources for this knowledge are contained in previous proof plans, in the form of the formulas in the proof, the concepts within these formulas, the applied methods, the control rules, the history and time span of the proof search, and the probability distribution for finding a proof. Hence the items to learn from are complex objects and comparing different proof plans is even more complex. One of our approaches focused on learning sequences of method applications.

Figure 2 shows the structure of this approach: first, the user has to choose "typical" examples of proof plans. The proof plans are then abstracted, and only the sequences of method specifiers remain. A generalization of the sequences is created, which is a pattern of all the input sequences in a language containing operators for disjunction, branching points, arbitrary number of repetitions, and also a fixed number of repetitions of sequences of methods.

The system, which is called LEARNΩMATIC, was evaluated in three domains: group theory, set theory, and residue classes. The results corroborated learned methods can make the search for a proof plan more directed, that is, less methods are tested for applicability if the new methods are preferred. In group theory, it was possible to prove theorems which could not be

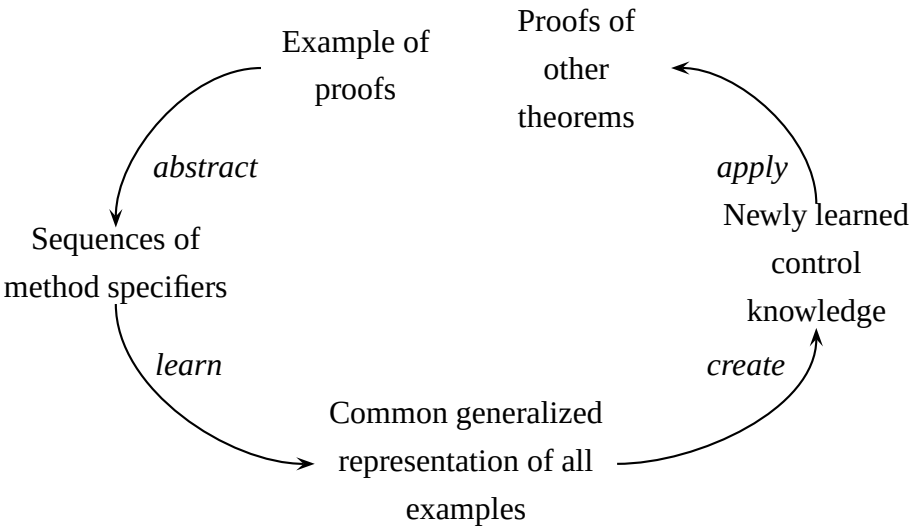

Figure 2. Approach to learning control knowledge for proof methods

proved without the learned methods. The length of the proofs was shorter with the learned methods in all domains. The work was reported in (Jamnik et al., 2003, 2002a) and a system description was given in (Jamnik, Kerber, & Pollet, 2002b).

Another approach (Meier, Gomes, & Melis, 2001) focused on randomization and restart techniques for learning strategic control rules. These techniques are truly resource-adaptive ones, where the resource considered is time. The mathematical statical background of randomization and restart makes it a methodologically sound method that has been applied for other AI-problems as well, but has not yet been used for learning control rules.

3.4.6 The Infrastructure of ΩMEGA (AP6)

To enhance the communication of ΩMEGA with other systems by a standardized format, an XML-RPC protocol interface was added to ΩMEGA. The protocol is also used for the communication with the ACTIVEMATH tutoring system in proof exercises and for the communication with the mathematical knowledge base MBASE. Communication with all other external systems is based on the MATHWEB Software Bus that has been improved in several ways. MATHWEB-SB is now based on a dynamic network of brokers that register (resp. unregister from) each other (Zimmer & Kohlhase, 2002). This drastically increased scalability and availability of the MATHWEB-SB and, therefore, of ΩMEGA with sometimes several thousand theorems or small lemmas proved each day by external users over the Internet. Furthermore, we integrated new reasoning systems into MATHWEB-SB which are thus available also to

ΩMEGA: The theory formation system HR supports the instantiation of the meta-variables in ΩMEGA (Colton, 2002), the tptp2X utility supports a uniform input format for automated theorem provers (ATPs) which are employed in ΩMEGA. All ATPs now return one of 22 well-defined states which unambiguously describe the provers' results.

The mathematical services available via MATHWEB-SB are:

- Proof assistants: ΩMEGA, CORE, PVS.
- Mathematical database: MBASE.
- Theory formation system: HR.
- Constraint solver: $\mathcal{CoSIE}$.
- Proof explanation system: *P.rex*; an external system developed by the OMEGA group (Fiedler, 2001).
- Generalization module of LEARNΩMATIC.
- Proof planners: MULTI, λCLAM.
- Computer algebra systems: MAPLE, GAP, COCOA, MAGMA.
- Automated higher-order theorem provers: TPS, LEO.
- Automated first-order theorem provers: BLIKSEM, OTTER, PROTEIN, SPASS, VAMPIRE.
- Automated equational provers: E, EQP, WALDMEISTER.
- Model generators/checkers: SEM, FINDER, MACE, SATCHMO.
- Proof transformation system: TRAMP.
- Translators for OMDOC and TPTP.

Further related work: The completeness of the higher-order natural deduction calculus, which is employed within the proof planner of the ΩMEGA system, has been proved by Benzmüller, Brown, and Kohlhase (2004).

3.5 Comparison With Research Outside of the Collaborative Research Center

The OMEGA group has become a leading force in the field of mathematical assistance systems, as evidenced inter alia by its role as coordinator of the EU research training network

CALCULEMUS-I (Benzmüller, 2003b, 2003a), as coordinator of the funding proposal for a successor network CALCULEMUS-II, as the main organizer of the CALCULEMUS Autumn School in Pisa in 2002, as organizer of two workshops at the International Joint Conference on Automated Reasoning (IJCAR 2004), as a driving force in the emerging Mathematical Knowledge Management community, and by its many international research collaborations, the most important ones are:

With joint publications: Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Germany; Department of Computer Science, Yale University, USA (Schürmann); Department of Mathematics, Carnegie Mellon University, Pittsburgh, USA (Brown); Department of Informatics, The University of Edinburgh, UK (Dennis, Bundy); RIACA, Eindhoven Technical University, Eindhoven, Netherlands (Cohen); Department of Computer Science, The University of Birmingham, UK (Kerber, Sorge); Computer Laboratory of the University of Cambridge, Cambridge, UK (Jamnik); International University of Bremen, Germany (Kohlhase); Department of Informatics, University of Kaiserslautern, Germany (Avenhaus, Madlener); Department of Computer Science, Technical University Munich, Germany (Schulz); Department of Computer Science, University of Miami, USA (Sutcliff); Department of Computing, Imperial College London, UK (Colton).

With joint funding, but no publications so far: Institut für Informatik, Universität Potsdam, Germany (Kreitz); Uniwersytet w Bialymstoku, Bialystok, Poland (Trybulec); Universita degli Studi di Genova, Genova, Italy (Armando); Department of Computer Science, University of Karlsruhe, Germany (Calmet); Research Institute for Symbolic Computation, Linz, Austria (Buchberger); Instituto per la Ricerca Scientifica e Tecnologica, Trento, Italy (Giunchiglia).

The collaboration with the University of Birmingham (M. Kerber and V. Sorge, who also employ ΩMEGA), and with the University of Edinburgh (A. Bundy et. al., they collaborate in the development of MATHWEB), has been particularly fruitful. Under the leadership of the International University of Bremen (M. Kohlhase) our group contributed to the development of MBASE and OMDOC, and a joint workshop series[9] with RISC Linz has been set up.

The international recognition of the group's achievements is also reflected in the recent appointment of M. Kohlhase as full professor at the International University Bremen, Volker Sorge's appointment as a lecturer at the University in Birmingham, the invited talks at highly recognized international workshops and conferences and by the number of high quality publications in international journals and international conferences (Siekmann et al., 2003, 2002; Kerber & Pollet, 2002a; Jamnik et al., 2002b, 2002a; Meier et al., 2002; Hübner et al., 2004; Vo, 2004; Vo et al., 2003; Wirth, 2002; Melis et al., 2001, 2003; Benzmüller et al., 2004; Benzmüller, 2002; Meier et al., 2001; Zimmer & Melis, 2004).

Cooperations within the Collaborative Research Center include:

- DIALOG (MI 4): empirical data in DIALOG are fertilizing the current redevelopment of

[9] `http://www.ags.uni-sb.de/~omega/workshops/TheoremaOmega03/`

ΩMEGA on top of CORE and vice versa the capabilities and features of the ΩMEGA/CORE environment are a crucial factor for the DIALOG system.

- NEP (MI 7): Constraint solving and the general computational model.

3.6 Open Issues

Not applicable

References

Autexier, S. (2001). A proof-planning framework with explicit abstractions based on indexed formulas. In M.-P. Bonacina & B. Gramlich (Eds.), *Proceedings of the 4^{th} Workshop on Strategies in Automated Deduction (STRATEGIES'01)* (Vol. TR DII 10/01, p. 87-99). Universitá degli studi di Siena.

Autexier, S. (2003). *Hierarchical contextual reasoning.* Unpublished doctoral dissertation, Fachrichtung Informatik, Universität des Saarlandes, Saarbrücken, Germany.

Bartle, R. G., & Sherbert, D. (1982). *Introduction to real analysis* (2 ed.). Wiley.

Benzmüller, C. (2002). Comparing approaches to resolution based higher-order theorem proving. *Synthese, An International Journal for Epistemology, Methodology and Philosophy of Science, 133*(1-2), 203–235.

Benzmüller, C. (2003a). The calculemus research training network: A short overview. In *Proceedings of the 11th Symposium on the Integration of Symbolic Computation and Mechanized Reasoning(CALCULEMUS 2003).* Rome, Italy.

Benzmüller, C. (Ed.). (2003b). *Systems for integrated computation and deduction — interim report of the calculemus ihp network* (SEKI report Nos. SR–03–05).

Benzmüller, C., Brown, C., & Kohlhase, M. (2004). Higher order semantics and extensionality. *Journal of Symbolic Logic*. (To appear)

Benzmüller, C., Jamnik, M., Kerber, M., & Sorge, V. (2001). Experiments with an agent-oriented reasoning system. In *Proceedings of KI' 2001* (Vol. 2174). Springer.

Benzmüller, C., Meier, A., Melis, E., Pollet, M., & Sorge, V. (2001a). Proof planning: A fresh start? In *Proceedings of the IJCAR 2001 Workshop: Future Directions in Automated Reasoning.* Siena, Italy.

Benzmüller, C., Meier, A., Melis, E., Pollet, M., & Sorge, V. (2001b). Proof planning: A fresh start? In *Proceedings of the IJCAR 2001 Workshop: Future Directions in Automated Reasoning.* Siena, Italy.

Benzmüller, C., Meier, A., & Sorge, V. (2003). Bridging theorem proving and mathematical knowledge retrieval. In *Festschrift in Honour of Jörg Siekmann.* Springer. (To appear)

Benzmüller, C., & Sorge, V. (2000). Oants – an open approach at combining interactive and automated theorem proving. In M. Kerber & M. Kohlhase (Eds.), *Proceedings of the Calculemus Symposium 2000.* St. Andrews, UK: AK Peters, New York, NY, USA.

Bledsoe, W. W. (1990). Challenge problems in elementary calculus. *Journal of Automated*

Reasoning, 6(3), 341–359.

Bundy, A. (2002). A critique of proof planning. In A. C. Kakas & F. Sadri (Eds.), *Computational Logic: Logic Programming and Beyond - Essays in Honour of Robert A. Kowalski, Part II* (Vol. 2408, p. 160-177). Springer.

Cheikhrouhou, L. (1997). Planning Diagonalization Proofs. In G. Brewka, C. Habel, & B. Nebel (Eds.), *21st Annual German Conference on Artificial Intelligence (KI'97)* (p. 377-380).

Cheikhrouhou, L., & Siekmann, J. H. (1998). Planning Diagonalization Proofs. In F. Giunchiglia (Ed.), *Proceedings of 8th International Conference on Artificial Intelligence: Methodology, Systems, Applications (AIMSA'98)* (pp. 167–180). Sozopol, Bulgaria: Springer Verlag, Berlin, Germany, LNAI 1480.

Cohen, A., Murray, S., Pollet, M., & Sorge, V. (2003). Certifying solutions to permutation group problems. In F. Baader (Ed.), *19th Conference on Automated Deduction (CADE-19)* (Vol. 2741, pp. 258–273). Springer.

Colton, S. (2002). The HR program for theorem generation. In *Proceedings of the 18th International Conference on Automated Deduction (CADE-18)* (Vol. 2392, pp. 285–289). København: Springer.

Denzinger, J., & Schulz, S. (1996). Learning domain knowledge to improve theorem proving. In *Proc. of 13th International Conference on Automated Deduction CADE-13* (p. 62-76). Springer.

Desimone, R. (1987). Learning control knowledge within an explanation-based learning framework. In I. Bratko & N. Lavrač (Eds.), *Progress in Machine Learning – Proceedings of 2nd European Working Session on Learning, EWSL-87*. Wilmslow, UK: Sigma Press. (Also available from Edinburgh as DAI Research Paper 321)

Fiedler, A. (2001). Dialog-driven adaptation of explanations of proofs. In B. Nebel (Ed.), *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 1295–1300). Seattle, WA: Morgan Kaufmann.

Franke, A. (2003). *Inhaltsorientierte Verwaltung mathematischen Wissens.* Unpublished master's thesis, Fachbereich Informatik, Universität des Saarlandes, Saarbrücken.

Franke, A., & Kohlhase, M. (2000). System description: MBASE, an open mathematical knowledge base. In D. McAllester (Ed.), *Automated Deduction – CADE-17* (pp. 455–459). Springer Verlag.

Franke, A., Moschner, M., & Pollet, M. (2002). Cooperation between the mathematical knowledge base MBASE and the theorem prover omega [Seki Report]. In O. Caprotti & V. Sorge (Eds.), *10th Symposion on the Integration of Symbolic Computation and Mechanized Reasoning, Work in Progress Papers* (pp. 68–70). Saarbrücken, Germany.

Fuchs, M., & Fuchs, M. (1998). Feature-based learning of search-guiding heuristics for theorem proving. *AI Communications, 11*, 175–189.

Hübner, M., Autexier, S., Benzmüller, C., & Meier, A. (2004). Interactive theorem proving with tasks. *Electronic Notes in Theoretical Computer Science*. (To appear)

Hübner, M., Benzmüller, C., Autexier, S., & Meier, A. (2003). Interactive proof construction at the task level. In *Proceedings of the Workshop User Interfaces for Theorem Provers (UITP 2003).* Rome, Italy.

Ireland, A. (1992). The use of planning critics in mechanizing inductive proofs. In A. Voronkov (Ed.), *Proceedings of LPAR 92* (pp. 178–189). Springer.

Ireland, A., & Bundy, A. (1995). Productive use of failure in inductive proof. *Special Issue*

of the Journal of Automated Reasoning, *16*(1–2), 79–111.

Jamnik, M., Kerber, M., & Benzmüller, C. (2001). *Learning method outlines in proof planning* (Cognitive Science Research Paper Nos. CSRP–01–08). School of Computer Science, The University of Birmingham.

Jamnik, M., Kerber, M., & Pollet, M. (2002a). Automatic learning in proof planning. In F. van Harmelen (Ed.), *15th European Conference on Artificial Intelligence (ECAI)* (pp. 282–286). IOS Press.

Jamnik, M., Kerber, M., & Pollet, M. (2002b). Learnomatic: System description. In A. Voronkov (Ed.), *Proceedings of the 18th International Conference on Automated Deduction (CADE-18)* (Vol. 2392, pp. 150–155). Springer.

Jamnik, M., Kerber, M., Pollet, M., & Benzmüller, C. (2003). Automatic learning of proof methods in proof planning. *Logic Journal of the IGPL*, *11*(5), 647-674.

Kerber, M., Kohlhase, M., & Sorge, V. (1998). Integrating computer algebra into proof planning. *Journal of Automated Reasoning*, *21*(3), 327–355.

Kerber, M., & Pollet, M. (2002a). On the design of mathematical concepts. In B. McKay & J. Slaney (Eds.), *15th Australian Joint Conference on Artificial Intelligence* (Vol. 2557, p. 716). Springer.

Kerber, M., & Pollet, M. (2002b). On the design of mathematical concepts [Seki Report]. In O. Caprotti & V. Sorge (Eds.), *10th Symposion on the Integration of Symbolic Computation and Mechanized Reasoning, Work in Progress Papers* (pp. 33–49). Saarbrücken, Germany.

Kohlhase, M. (2000). OMDOC: An infrastructure for OPENMATH content dictionary information. *Bulletin of the ACM Special Interest Group on Symbolic and Automated Mathematics (SIGSAM)*, *34*(2).

Meier, A. (2004). *Proof planning with multiple strategies.* Unpublished doctoral dissertation, Computer Science Department, Saarland University, Saarbrücken, Germany.

Meier, A., Gomes, C., & Melis, E. (2001). Randomization and restarts in proof planning. In A. Cesta & D. Borrajo (Eds.), *European Conference on Planning* (p. 403-408). Springer-Verlag.

Meier, A., Melis, E., & Pollet, M. (2004). Adaptable mixed-initiative proof planning for educational interaction. *Electronic Notes in Theoretical Computer Science*. (To appear)

Meier, A., Pollet, M., & Sorge, V. (2001). Classifying Isomorphic Residue Classes. In R. Moreno-Diaz, B. Buchberger, & J.-L. Freire (Eds.), *A Selection of Papers from the 8th International Workshop on Computer Aided Systems Theory (EuroCAST 2001)* (Vol. 2178, pp. 494 – 508). Las Palmas, Spain: Springer Verlag, Berlin Germany. (In print.)

Meier, A., Pollet, M., & Sorge, V. (2002). Comparing approaches to the exploration of the domain of residue classes. *Journal of Symbolic Computation, Special Issue on the Integration of Automated Reasoning and Computer Algebra Systems*, *34*(4), 287–306. (Steve Linton and Roberto Sebastiani, eds.)

Meier, A., Sorge, V., & Colton, S. (2002). Employing theory formation to guide proof planning. In J. Calmet, B. Benhamou, O. Caprotti, L. Henocque, & V. Sorge (Eds.), *Artificial Intelligence, Automated Reasoning, and Symbolic Computation — Joint International Conference, AISC 2002 and Calculemus 2002* (Vol. 2385, pp. 275–289). Marseille, France: Springer.

Melis, E. (2002). Knowledge representation for web-based user-adaptive education systems. In *BMBF-workshop: Standardisierung im eLearning* (p. 78-81).

Melis, E. (2003). Why proof planning for maths education and how? In D. Hutter & W. Stephan (Eds.), *Festschrift in Honor of Jörg Siekmann.* Springer-Verlag.

Melis, E., Buedenbender, J., Andres, E., Frischauf, A., Goguadse, G., Libbrecht, P., Pollet, M., & Ullrich, C. (2003). Knowledge representation and management in activemath. *International Journal on Artificial Intelligence and Mathematics, Special Issue on Management of Mathematical Knowledge*, *38*(1-3), 47-64.

Melis, E., Buedenbender, J., Andres, E., Frischauf, A., Goguadze, G., Libbrecht, P., Pollet, M., & Ullrich, C. (2001). ACTIVEMATH: A generic and adaptive web-based learning environment. *Artificial Intelligence and Education*, *12*(4).

Melis, E., & G. Goguadse, C. U., P. Libbrecht. (2003). Wissensmodellierung und -nutzung in ACTIVEMATH. *KI*(1), 12-18.

Melis, E., Glasmacher, C., Ullrich, C., & Gerjets, P. (2001). Automated proof planning for instructional design. In *Annual Conference of the Cognitive Science Society* (p. 633-638).

Paulson, L. C., & Nipkow, T. (1990). *Isabelle tutorial and user's manual* (Tech. Rep. No. 189). Computer Laboratory, University of Cambridge.

Pollet, M., Melis, E., & Meier, A. (2003). User interface for adaptive suggestions for interactive proof. In *Proceedings of the Workshop User Interfaces for Theorem Provers (UITP 2003)* (pp. 133–142). Rome, Italy: Aracne Editrice S.R.L. (Also available as: Technical Report No. 189, Institut für Informatik, Albert-Ludwig-Universität, Freiburg.)

Pollet, M., & Sorge, V. (2003). Integrating computational properties at the term level. In T. Hardin & R. Rioboo (Eds.), *Proceedings of the 11^{th} Symposium on the Integration of Symbolic Computation and Mechanized Reasoning (Calculemus 2003).* Rome, Italy.

Schulz, S. (2000). *Learning search control knowledge for equational deduction.* Unpublished doctoral dissertation, Fakultät für Informatik, Technische Universität München, Munich, Germany.

Siekmann, J., Benzmüller, C., Brezhnev, V., Cheikhrouhou, L., Fiedler, A., Franke, A., Horacek, H., Kohlhase, M., Meier, A., Melis, E., Moschner, M., Normann, I., Pollet, M., Sorge, V., Ullrich, C., Wirth, C.-P., & Zimmer, J. (2002). Proof development with Omega. In *Proceedings of the 18th International Conference on Automated Deduction (CADE-18)* (Vol. 2392, pp. 143–148). København: Springer.

Siekmann, J., Benzmüller, C., Fiedler, A., Meier, A., Normann, I., & Pollet, M. (2003). Proof development in OMEGA: The irrationality of square root of 2. In F. Kamareddine (Ed.), *Thirty five years of automating mathematics* (p. 271-314). Kluwer Academic Publishers. (ISBN 1-4020-1656-5)

Silver, B. (1984). Precondition analysis: Learning control information. In R. Michalski, J. Carbonell, & T. Mitchell (Eds.), *Machine Learning 2.* Palo Alto, CA: Tioga Press.

Sorge, V. (2001). *A blackboard architecture for the integration of reasoning techniques into proof planning.* Unpublished doctoral dissertation, Department of Computer Science, Saarland University, Saarbrücken, Germany.

Vo, Q. B. (2003). A task-oriented agent-based mechanism for theorem proving. In *The 2003 IEEE/WIC International Conference on Intelligent Agent Technology (IAT 2003)* (pp. 275–281).

Vo, Q. B. (2004). A task-oriented information mediator for mathematical assistant systems. *International Journal of Web Intelligence and Agent Systems.* (to appear)

Vo, Q. B., Benzmüller, C., & Autexier, S. (2003). Assertion application in theorem proving

and proof planning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 1343–1344). Acapulco, Mexico. (poster description)

Wiedijk, F. (2003). Comparing mathematical provers. In A. Asperti, B. Buchberger, & J. Davenport (Eds.), *Mathematical Knowledge Management, Proceedings of MKM 2003* (Vol. 2594, pp. 188–202). Springer.

Wirth, C.-P. (2002). A new indefinite semantics for Hilbert's epsilon. In U. Egly & C. G. Fermüller (Eds.), *Automated Reasoning with Analytic Tableaus and Related Methods. International Conference, TABLEAU 2002* (Vol. 2381, pp. 298–314). Springer.

Wirth, C.-P. (2004). Descente infinie + Deduction. *96 pp., accepted by Logic Journal of the IGPL*. (`www.ags.uni-sb.de/~cp/p/d/welcome.html`)

Zimmer, J., & Kohlhase, M. (2002). System description: The MathWeb software bus for distributed mathematical reasoning. In A. Voronkov (Ed.), *18th Conference on Automated Deduction (CADE-18)* (Vol. 2392). Copenhagen, Denmark: Springer.

Zimmer, J., & Melis, E. (2004). Constraint solving for proof planning. *Journal of Automated Reasoning*. (accepted)