

Agent Based Mathematical Reasoning¹

Christoph Benz Müller^a Mateja Jamnik^b
Manfred Kerber^b Volker Sorge^a

^a *Fachbereich Informatik
Universität des Saarlandes
66041 Saarbrücken, Germany
{chris|sorge}@ags.uni-sb.de
<http://www.ags.uni-sb.de/>*

^b *School of Computer Science
The University of Birmingham
Birmingham B15 2TT, England
{M.Jamnik|M.Kerber}@cs.bham.ac.uk
<http://www.cs.bham.ac.uk/>*

Abstract

In this contribution we propose an agent architecture for theorem proving which we intend to investigate in depth in the future. The work reported in this paper is in an early state, and by no means finished. We present and discuss our proposal in order to get feedback from the Calculemus community.

1 Introduction

There are two major approaches to automated theorem proving, machine-oriented methods like the resolution method (with all its ramifications) and human-oriented methods. Most prominent amongst the human-oriented methods is the proof planning approach first introduced by Bundy [8].

In this contribution we argue that an integration of the two approaches and the simultaneous pursuit of different lines in a proof can be very beneficial. One way of integrating the approaches is to consider a reasoner as a collection of agents, in which machine-oriented methods and planning play different rôles.

¹ This work was partly supported by EPSRC grant GR/M22031.

One of the main distinctions between machine-oriented and human-oriented methods is the generality of the approaches. Machine-oriented theorem provers like classical first-order theorem provers (e.g., BLIKSEM, OTTER, SPASS), analytical provers (e.g., SATCHMO or PROTEIN), and provers based on completion methods (e.g., EQP, WALDMEISTER) have reached a considerable reasoning power. This is underlined by the recent solution of the Robbins problem by EQP [21]. However, these traditional systems follow fixed search strategies which are unlikely to fully model the problem solving expertise of human mathematicians. Also classical higher-order theorem provers like TPS [1], or the LEO-system [4] get lost in the enormous search spaces stretched on a fine-grained calculus level. General complexity results demonstrate that no practical algorithm can be constructed which can solve arbitrary tasks. Even propositional logic is in a class that is generally considered intractable since it is NP-complete.

The success of human mathematicians can largely be ascribed to the fact that they are generally specialised in some fields and can rely on *domain-specific* problem solving techniques they have accumulated throughout their professional experiences. Mathematicians learn during their academic training not only facts like definitions or theorems, but also problem-solving *know-how* for proving mathematical theorems. An important part of this know-how can be described in terms of reasoning methods like the diagonalisation procedure, the application of a definition, or the application of the homomorphy property. Human-oriented theorem proving tries to model this human approach by making use of domain-specific knowledge.

One approach to model human-oriented theorem proving on a computer is proof planning which adopts the planning paradigm. The so-called methods play the rôle of plan operators and their executions fill gaps in a partial proof. Bundy views methods essentially as a triple consisting of a tactic, a precondition, and a postcondition. A tactic can be seen as a piece of program code that can manipulate the actual proof in a controlled way. A precondition and a postcondition form a declarative specification of the deductive ability of the tactic. The approach to mechanising reasoning using methods had resulted in a significant progress compared to a mere tactic language. Within such a planning framework it is now possible to develop proof plans with the help of the declarative knowledge in the preconditions and postconditions. In this view proof planning makes use of traditional planning techniques in order to find proofs on an abstract level. State of the art proof planners are CLAM [10], λ -CLAM [24], and the proof planner of Ω MEGA [3].

One of the first successful approaches to theorem proving within an agent architecture is Denzinger's TEAMWORK approach [12]. Some of the state of the art agent architectures for automatic and interactive theorem proving are discussed in [16,5]. The agent architecture for proof planning proposed here

will be developed within the framework of Ω MEGA. An advantage of Ω MEGA is that it already provides various integrated classical reasoning systems (e.g., BLIKSEM, OTTER, EQP, SPASS, SATCHMO, PROTEIN, WALDMEISTER, TPS, LEO) as well as some specialised decision procedures (for instance, a constraint solver and the integrated computer algebra systems MAPLE and μ CAS [19]), and an analogy module [22]. Additional features are a multi-modal graphical user interface [25], a proof verbalisation tool [18] and a connected database of mathematical theories. Using the MATHWEB agent architecture [17], most of these integrated systems can be distributed over the Internet. Information on successful or unsuccessful proof attempts of the integrated systems (e.g., partial proofs) can be translated back into Ω MEGA's central proof data structure, which is based on a higher-order variant of Gentzen's natural deduction calculus. Translation of different results into the uniform representation in Ω MEGA clarifies the integrated results of very heterogeneous agents.

Human proof search behaviour may perhaps be best modelled as a mixture of proof planning, classical theorem proving, computing, and model generation. In Ω MEGA (and in related systems like ILF [11] or DISCOUNT [14], which integrate relatively homogeneous first-order reasoning systems) this largely has to be done by the user. Rather poor support is provided for a fruitful and guided cooperation of the available subsystems.

While Ω MEGA and MATHWEB provide the technical background, the work described here aims to investigate how a meaningful cooperation between different agents within the architecture can be established. We want to draw as much as possible on already existing technology both in Ω MEGA and in other external systems (i.e., theorem provers, computer algebra systems, etc.). Therefore, the communication between agents will be organised so that successful and unsuccessful proof attempts or partial proofs are communicated via Ω MEGA. The assessment of single agents and societies of agents will be embedded within Ω MEGA as well as agent shells surrounding the single systems in use. Hence, theorem provers which have communication features readily available (e.g., TPS) will be used off the shelf, as black-box systems. The information they provide will be incorporated at run-time into the reasoning process searching for a proof of a conjecture.

2 Deliberation versus Reactiveness

A classical approach to model intelligence consists of carefully selecting a knowledge representation formalism and then modelling parts of a domain in this formalism. The power of the reasoner depends on the ability to reason in the knowledge representation formalism. In classical approaches to planning, the situation calculus [20] and in STRIPS [15], the domain is modelled by an initial state, a goal state, and planning operators. Proof planning with

abstraction also makes use of a model of the world and can be viewed as a deliberative approach to solve reasoning tasks.

As an antithesis to this classical AI paradigm (and in particular to the planning paradigm) an approach has been developed that explicitly does not make use of knowledge representation and complicated deliberations (such as planning the best next step in the proof search). Brooks phrased it as “*Intelligence without Reason*” [7]. In this approach it is possible to obtain complex, apparently goal directed and intentional behaviour which has no long term internal state and no internal communication. This is referred to as a reactive form of modelling behaviour. Its key notions are:

Situatedness: *The world is its own best model.*

Embodiment: *The world grounds regress.*

Intelligence: *Intelligence is determined by the dynamics of interaction with the world.*

Emergence: *Intelligence is in the eye of the observer.*

Although the machine-oriented approaches were not designed in the light of recent work in reactive systems, they can be reinterpreted in this framework. The main aspect is the locality of the search for a solution. For instance, when we consider binary resolution theorem proving, the decision on which two literals to perform a resolution step is often made on the basis of the knowledge of the current proof state only. That is, it does not depend on what has been done previously (of course this view simplifies matters and is not true in a strict sense for all strategies). In particular, there is no overall long term strategy to derive the empty clause. We can view the behaviour of the theorem prover as a reactive process: the world consists of clauses and there is no abstract model of these clauses. The theorem prover acts directly in this world, and the behaviour is determined by the interaction with the world. It is a characteristic of reactivity that some reactive systems such as OTTER normally do not do any backtracking. Furthermore, some reactive systems do complete restarts when the search for a proof is lost in the search space. Such restarts can also be viewed as a typical characteristic of reactive systems. For a detailed discussion of Brooks’ approach and its relationship to theorem proving see [9].

Recent years have seen an attempt to reconcile the deliberative and the reactive approaches in single agent architectures [26]. This is partly motivated by looking at the human way of acting and reasoning which can be better explained as a combination of the two cases rather than by any one of them alone. Also, practical issues play an important rôle: in certain cases reactive behaviour is computationally more efficient, while in others reactive behaviour gets stuck. In the latter case deliberative behaviour can sometimes prevent blocking of a reasoning process.

3 Agent based mathematical reasoning

A weakness of most state of the art reasoning systems is that they usually follow rigid and inflexible solution strategies in their search for proofs. Instead, human mathematicians use — depending on their level of expertise — “a *colourful mixture of proof strategies*” (as Wittgenstein phrases it). In an attempt to prove a mathematical theorem they typically first try a well known standard technique in the focus of the mathematical theory. If this technique does not lead to the wanted results in a reasonable amount of time, they may doubt that the theorem holds at all and look for a counterexample. If this also fails, they may try again by widening or deepening the proof search.

The aim of our approach is to emulate this flexible problem solving behaviour of human mathematicians in an agent based reasoning approach. Thus, our system will reflect at least some of the ideas of a *sophisticated and experienced problem solver* as described by Pólya in [23], p. 64: “... *when he does not succeed in guessing the whole answer, [he] tries to guess some part of the answer, some feature of the solution, some approach to the solution, or some feature of an approach to the solution. Then he seeks to expand his guess, and so he seeks to adapt his guess to the best information he can get at the moment.*”

Agents allow a number of proof search attempts to be executed in parallel. Each agent may try a different proof strategy to find the proof of a conjecture. Hence, a number of different proof strategies are used at the same time in the proof search. However, following all the available strategies simultaneously would quickly consume the available system resources consisting of computation time and memory space. In order to prevent this, and furthermore, to guide the proof search we propose to develop and employ a resource management concept in proof search. Resource management is a technique which distributes the available resources amongst the available agents (cf. [28]). Periodically, it assesses the state of the proof search process, evaluates the progress and redistributes the available resources accordingly. Hence, only successful or promising proof attempts will be allowed to continue searching for a proof. This process is repeated until a proof is found, or some other terminating condition is reached. An important aspect will be that in each assessment/evaluation phase the global proof state is updated, that is, promising partial proofs and especially solved subproblems are inserted into the global proof tree. Furthermore, interesting results may be communicated between the agents (for instance, an open subproblem may be passed to a theorem prover that seems to be more appropriate). The resource management mechanism analyses the theorem and decides which agents, i.e., provers, need to be launched and what proportion of the resources needs to be assigned to a particular agent. The mechanism is also responsible for restricting the amount of information exchange between agents, so that not all of the resources are

allocated to the communication. Figure 1 demonstrates this agent based proof planning architecture.

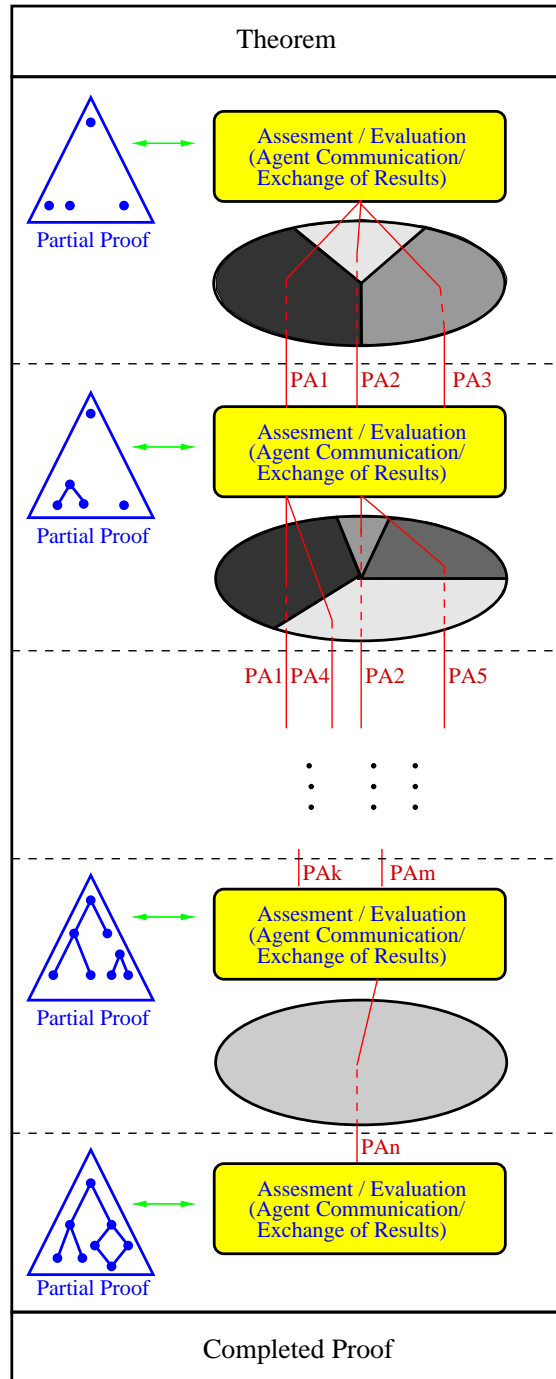


Fig. 1. The reasoning process – iterative allocation of resources to proof agents (PAx) by assesment/evaluation, and the subsequent construction of a proof of a given theorem.

Of course, the evaluation of the success of a proof strategy is crucial for determining the amount of resources that is allocated to an agent. This evaluation is based on the contribution that the agent has made in the proof attempt as well as on its prospect of success in the rest of the search. For example, a favourable contribution is a partial problem solution. Ω MEGA integrates most external systems as glass boxes. That is, it provides mechanisms to map particular results of external systems (e.g., single clauses derived by SPASS or OTTER) to ND-derivations in Ω MEGA's central proof data structure (PDS). This feature of Ω MEGA will benefit our approach in that the evaluation of the contribution of external systems can be based on the examination of the corresponding ND-proofs or proof plans. The future prospect of an agent is estimated with respect to the updated global proof tree and according to the information communicated between the agents.

The proposed system should be able to tackle mathematical problems that are currently not automatically solvable by any of the embedded systems alone (nor by any other system). An example of such a problem is described in detail in [2], we briefly summarise it here. The problem states that if there is a partition p of some set, then there is an equivalence relation q whose equivalence classes are exactly the elements of p :

$$\forall p. \text{partition}(p) \Rightarrow (\exists q. \text{equivalence-rel}(q) \wedge (\text{equivalence-classes}(q) = p))$$

Note that *partition*, *equivalence-classes*, and *equivalence-rel* are derived higher-order concepts defined in the Ω MEGA knowledge base of mathematical theories.

The search for a proof of this theorem has not been automated yet. We indicate here how the proof might be found within our proposed architecture. First, the initial proof goal is split into three subgoals (e.g., with the help of a proof planner). Namely, from a given partition p we can derive the existence of an equivalence relation q , constituting subgoal (1). For the same equivalence relation q it holds that its equivalence classes are exactly p . The two directions of the set equality give us subgoals (2) and (3). Next, higher-order equality and extensionality reasoning is required. The first two subgoals (1) and (2) can be solved automatically by the higher-order prover TPS [1]. The last subproblem (3), which requires a fair amount of extensionality reasoning, we expect to be solvable by cooperation between the higher-order extensionality prover LEO [4] and a first-order automated theorem prover. LEO provides the necessary higher-order extensionality treatment, however, it cannot cope with the large number of first-order clauses that are generated subsequently. Therefore, this set of clauses could be passed via Ω MEGA to the first-order specialist available within our agent society. Our proposed system will be able to organise the sketched cooperation between the integrated systems in a goal oriented way in order to solve such kinds of problems automatically.

4 Theorem Proving by a Society of Agents

The system we propose will provide a powerful architecture for reasoning systems consisting of a society of specialised reasoning agents. These agents are aware of their own capabilities and partly even of those of the other agents. The knowledge can initially be provided by the user or the implementor of a single agent. However, additional knowledge can be gained by evaluating successful and unsuccessful proof attempts in various mathematical domains as well as by feedback from other agents (for instance, the usefulness of results from some agents can be used in a reinforcement learning approach).

Initially, a given mathematical problem is investigated in order to estimate and classify the potential of the solution strategies, i.e., of the agents, available for solving this problem. Depending on the evaluation process an initial resource distribution is computed, in particular, a main strategy line may be manifested. An infrastructure allowing to distribute resources is already provided by the MATHWEB architecture [17]. An automatic evaluation module will then be added to the system. The goal is not to remove the human in this process; on the contrary, the agent and resource approach should strongly facilitate the communication between the human and the machine. In fact, human expertise can be incorporated during each of the assessment/evaluation phases.

After consuming the available resources, the reasoning agents terminate and investigate whether they have produced useful information or not. For instance, the OTTER-agent could look for the shortest derived clauses with assertion clauses as ancestors in order to estimate how close it is to a completed proof. More interesting in our context will be contributions of the TPS system, since it can return partial proofs to the Ω MEGA-system. These results may be evaluated using adequate criteria like the complexity and the number of the remaining open subproblems. For example, the only open subproblem might be a first-order goal, whereas the original problem was a higher-order one. Then, the partial proof may be communicated to other systems and the open subgoal can be passed to first-order provers. Depending on the evaluation of the agents' contributions, a new resource distribution is computed.

The starting point for the design of the system we propose here consists of a proof data structure and a proof planning mechanism. The first prototype of the system can use the existing proof data structure, proof planning components, and proof methods of Ω MEGA. The system will be extended by implementing a mechanism for knowledge based automatic distribution of subproblems to societies of agents, and an assessment module which will enable an interaction between agents.

The agent results can be incorporated directly into Ω MEGA's partial proofs, enabling the evaluation of usefulness of heterogeneous agents on some uniform

level. The information can then be propagated to other agents. However, in case of an unsuccessful proof attempt of the overall system a special backtracking mechanism needs to be supplied. It has to do book-keeping on the parts of the proofs which have been computed by each agent. Furthermore, the mechanism must be able to subsequently remove both, whole and partial results of an agent from the overall proof.

One of the potential problems, which we foresee, is that increasing the heterogeneity of a system might increase the organisational complexity of the communication between the agents. Namely, the greater the variety of the systems that are integrated, the less there might be a common interest to the different agents, and furthermore, the general viewpoint of the problem solving process of the overall system might be lost. For instance, some intermediate result that is of central importance to one prover might not be of interest to another prover, because the proof strategies that they use are very different. Hence, establishing the communication between agents might prove to be difficult. As a first approximation, our approach will be to broadcast the results of each agent to every other agent in the hope that the results might be useful to other agents. In further refinements we will look into more sophisticated forms of communication which will allow for a more efficient exchange of information between agents. The possibility to translate into the standard form of Ω MEGA's partial proofs should help in this task.

5 Conclusion

We summarise our proposal by delineating some of the most challenging research tasks in this project:

- (i) The extension of Ω MEGA's underlying MATHWEB-architecture and its proof data structure by suitable resource distribution, communication and backtracking facilities. In a first attempt we want to adapt the blackboard mechanism underlying Ω MEGA's interactive suggestion mechanism [5,6] and integrate it with the MATHWEB architecture.
- (ii) The development and realisation of a suitable evaluation criteria; some obvious candidates are the simplicity/complexity of partial proofs, the theory/logic a subproblem belongs to (e.g., first-order logic, set theory), and the similarity of open subproblems to already solved problems stored in the database.
- (iii) The extension of the system, such that it allows a grouping of homogeneous agents tackling similar kinds of problems into one single meta-agent. For instance, it may be useful to group classical first-order reasoners together to form a centre of expertise for classical first-order logic. Ideally, such centres of expertise may use a mechanism analogous to the

overall system in order to organise the communication between its systems (sub-agents) and to further distribute the resources they obtain at the upper level.

The systems in a centre of expertise can be evaluated using a fine-grained evaluation criteria. Evaluation experiments of this kind have been carried out in the past on, for instance, first-order theorem provers and other homogeneous systems (cf. [16,27,13]). They proved to be successful and gave positive results. Hence, we could realise a more homogeneous system communication within the centre of expertise. Furthermore, the centres of expertise could have a dynamic nature, that is, they might remodel themselves differently for different problem domains or explicitly learn in which areas their particular strengths and weaknesses are.

Related to our proposal is the work on TECHS in [13] where no restriction is imposed on the type of the provers that can be integrated into a system. A comparison of both architectures might provide some useful insights into the potential problems as well as the advantages of the approach proposed in this paper. One difference between the proposed approach and TECHS is that the latter does not provide techniques to translate selected results of the reasoning agents (e.g., clauses derived by a first-order theorem prover) into derivations in a uniform proof data structure, whereas for most systems in our approach this will be possible. Hence, our evaluation criteria may exploit knowledge on a more abstract level and may relate the contributions of the agents to the current partial proof in the global proof attempt.

In conclusion, we propose that a reasoning system with an agent based architecture incorporated into a proof planning framework, as we described in this paper, will result in improved mechanised reasoning capabilities. Unlike a conventional distributed parallel model of theorem proving, an agent architecture provides a paradigm where the communication between agents and the management of resources for agents can be realised. The hope is that such a system will be able to prove theorems that have previously not been proved automatically.

References

- [1] P. B. Andrews, M. Bishop, S. Issar, D. Nesmith, F. Pfenning, and H. Xi. TPS: A theorem proving system for classical type theory. *Journal of Automated Reasoning*, 16(3):321–353, 1996.
- [2] C. Benzmüller, M. Bishop, and V. Sorge. Integrating TPS and Ω MEGA. *Journal of Universal Computer Science*, 1999. Special issue on Integrating of Deduction System, forthcoming.
- [3] 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. Ω MEGA: Towards a mathematical assistant. In W. McCune, editor, *Proceedings of the 14th Conference on Automated Deduction*, pages 252–255. Townsville, Australia, 1997. Springer, LNAI 1249.
- [4] C. Benzmüller and M. Kohlhase. LEO — a higher-order theorem prover. In C. Kirchner and H. Kirchner, editors, *Proceedings of the 15th Conference on Automated Deduction*, pages 139–144, Lindau, Germany, 1998. Springer, LNAI 1421.
- [5] C. Benzmüller and V. Sorge. A blackboard architecture for guiding interactive proofs. In F. Giunchiglia, editor, *Artificial Intelligence: Methodology, Systems and Applications*, pages 102–114, Sozopol, Bulgaria, 1998. Springer, LNAI 1480.
- [6] C. Benzmüller and V. Sorge. Critical agents supporting interactive theorem proving. SEKI-Report SR-99-02, Fachbereich Informatik, Universität des Saarlandes, 1999.
- [7] R. A. Brooks. Intelligence without reason. Memo No. 1293, MIT, April 1991.
- [8] A. Bundy. The use of explicit plans to guide inductive proofs. In E. Lusk and R. Overbeek, editors, *Proceedings of the 9th Conference on Automated Deduction*, pages 111–120, Argonne, Illinois, USA, 1988. Springer, LNCS 310.
- [9] A. Bundy. A subsumption architecture for theorem proving. In *Philosophical Transactions of the Royal Society of London*, pages 71–85, Volume 349/1689, October 1994.
- [10] A. Bundy, F. van Harmelen, C. Horn, and A. Smaill. The OYSTER-CLAM system. In M. Stickel, editor, *10th Conference on Automated Deduction*, pages 647–648, 1990. Springer, LNAI 449.
- [11] I. Dahn. Integration of automated and interactive theorem proving in ILF. In W. McCune, editor, *Proceedings of the 14th Conference on Automated Deduction*, pages 57–60. Townsville, Australia, 1997. Springer, LNAI 1249,
- [12] J. Denzinger. Knowledge-Based Distributed Search Using Teamwork. In *Proceedings of the First International Conference on Multiagent Systems (ICMAS-95)*, pages 81–88, 1995.
- [13] J. Denzinger and I. Dahn. Cooperating theorem provers. In W. Bibel and P. H. Schmitt, editors, *Automated Deduction, a Basis for Application – Handbook of the German Focus Programme on Automated Deduction*, chapter II.14, pages 383–416. Kluwer academic publishers, Dordrecht, Netherlands, 1998.
- [14] J. Denzinger, M. Kronenburg, and S. Schulz. DISCOUNT – a distributed and learning equational prover. *Journal of Automated Reasoning*, 18(2):189–198, 1997.
- [15] R. E. Fikes and N. J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

- [16] M. Fisher. An open approach to concurrent theorem proving. In J. Geller, H. Kitano, and C. Suttner, editors, *Parallel Processing for Artificial Intelligence*, volume 3. Elsevier/North Holland, 1997.
- [17] A. Franke, S. Hess, C. Jung, M. Kohlhase, and V. Sorge. Agent-oriented integration of distributed mathematical services. *Journal of Universal Computer Science*, 5(3):156–187, 1999. Special issue on Integration of Deduction System.
- [18] X. Huang and A. Fiedler. Proof verbalization in *PROVERB*. In *Proceedings of the First International Workshop on Proof Transformation and Presentation*, pages 35–36, Schloss Dagstuhl, Germany, 1997.
- [19] M. Kerber, M. Kohlhase, and V. Sorge. Integrating Computer Algebra Into Proof Planning. *Journal of Automated Reasoning*, 21(3):327–355, 1998.
- [20] J. McCarthy and P. Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 4:463–502, 1969.
- [21] W. McCune. Solution of the Robbins problem. *Journal of Automated Reasoning*, 19(3):263–276, 1997.
- [22] E. Melis. A model of analogy-driven proof-plan construction. In C. S. Mellish, editor, *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI95)*, pages 182–189, Montréal, Canada, 1995. Morgan Kaufmann.
- [23] G. Pólya. *Mathematical Discovery*. John Wiley & Sons, 3rd edition, 1981.
- [24] J. Richardson, A. Smaill, and I. Green. System description: proof planning in higher-order logic with λ -CLAM. In C. Kirchner and H. Kirchner, editors, *15th Conference on Automated Deduction*, 1998. Springer, LNAI 1421.
- [25] J. Siekmann, S. Hess, C. Benz Müller, L. Cheikhrouhou, A. Fiedler, H. Horacek, M. Kohlhase, K. Konrad, A. Meier, E. Melis, M. Pollet, and V. Sorge. LOUI: Lovely Ω MEGA User Interface. Special Issue on Integrated Systems of the *Journal of Formal Aspects of Computer Science*, 1999. forthcoming.
- [26] A. Sloman. Architectural requirements for human-like agents – both natural and artificial. (What sorts of machines can love?). In K. Dautenhahn, editor, *Human Cognition and Social Agent Technology*. John Benjamins Publishing, 1999.
- [27] A. Wolf. P-SETHEO: Strategy parallelism in automated theorem proving. In H. de Swart, editor, *Proceedings of the International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX-98)*, 1998, page 320. Springer, LNAI 1397.
- [28] S. Zilberstein. Models of Bounded Rationality. In *AAAI Fall Symposium on Rational Agency*, Cambridge, Massachusetts, November 1995.