Chapter 9
Publication/Citation: A Proof-Theoretic Approach to Mathematical Knowledge Management*

Dexter Kozen and Ganesh Ramanarayanan

9.1 Introduction

There are many real-life examples of formal systems that support certain constructions or proofs, but that do not provide direct support for remembering them so that they can be recalled and reused the future. This task is usually left to some metasystem that is typically provided as an afterthought. For example, programming language design usually focuses on the programming language itself; the mechanism for accumulating useful code in libraries is considered more of a systems issue and is generally treated as a separate design task. Mathematics deals with the construction of proofs, but not with their publication and citation; that is the domain of the journals.

Automated deduction systems such as NuPrl [4, 6] and Mizar [16] have language support for accumulating results in libraries for later reference. However, the mechanisms for providing this support are typically not considered interesting enough to formalize in the underlying logic, although it is possible in principle to do so.

We regard publication/citation as an instance of *common subexpression elimination* on proof terms. These operations permit proofs to be reused, perhaps specialized to a particular context, without having to reconstruct them in every application.

In this paper we attempt to develop this idea from a proof-theoretic perspective. We describe a simple complete proof system for universal Horn equational logic with three new proof rules, **publish**, **cite** and **forget**. The first two rules allow the inclusion of proved theorems in a library and later citation. The last allows removal of theorems from the library. These rules encapsulate all the bookkeeping that must be done to ensure correctness in their application.

Dexter Kozen
Department of Computer Science, Cornell University, Ithaca, NY 14853-7501, USA,
e-mail: kozen@cs.cornell.edu
Ganesh Ramanarayanan
Department of Computer Science, Cornell University, Ithaca, NY 14853-7501, USA,
e-mail: graman@cs.cornell.edu

* In Honor of Professor Rohit Jivanlal Parikh on the Occasion of his 70th Birthday

J. van Benthem et al. (eds.), *Games, Norms and Reasons*, Synthese Library 353,
DOI 10.1007/978-94-007-0714-6_9,

The **publish** rule has the effect of publishing the universal closure of a theorem to the library. The combinator corresponding to **publish** wraps the proof term to inhibit β-reduction upon later citation. The result of this operation is a *citation token* that is type-equivalent to the original proof, thus interchangeable with it, but that does not perform the specialization that is supplied with the citation.

The **cite** rule allows for the application of published theorems using the type-equivalent citation token. The theorem is applied under a specialization given by a variable substitution and provided at the time of the citation. However, because of the wrapping done by **publish**, the substitution is not actually performed on the proof term.

The **forget** rule is correspondingly more involved, since it must remove all citations to the forgotten theorem from the library. This is accomplished by unwrapping all occurrences of the citation token, allowing the deferred substitutions and β-reductions to take place during proof normalization. Effectively, this replaces each citation of the forgotten theorem with an inlined specialization of the original proof, where the specialization is the one supplied when the theorem was cited.

A major advantage of our approach is that it avoids namespace management issues, allowing us to focus on the pure structure of publication/citation. In real systems, when a lemma is added to a library, it is usually given a name, and all subsequent references are by this name. This introduces the possibility of name collisions. To address this issue, these systems typically introduce some form of scoping or module structure. However, our proof-theoretic treatment of publication/citation allows us to avoid the problem entirely.

In this paper, we develop this idea for constructive universal equational Horn logic. However, it is clear that the mechanisms are more general and could be adapted to richer theories. Significant progress along these lines has already been made [1–3].

9.2 A Classical Proof System

We first present a classical proof system for constructive universal equational Horn logic as a basis for comparison. Let Σ be a signature consisting of function symbols $\Sigma = \{f, g, \ldots\}$, each with a fixed finite arity. Let Σ_n denote the set of elements of Σ of arity n. Let X be a set of individual variables $X = \{x, y, \ldots\}$, and let $T_\Sigma(X)$ denote the set of individual terms over Σ and X. Formally, an *individual term* is either

- a variable $x \in X$, or
- an expression of the form $ft_1 \ldots t_n$, where $t_1, \ldots, t_n$ are individual terms and $f \in \Sigma_n$.

For example, the signature of groups consists of a binary operator $\cdot$, a unary operator $^{-1}$, and a constant 1.

A *formula* is either

- an equation $s = t$ between individual terms,

- an implication of the form $s = t \to \varphi$, where φ is a formula, or
- a quantified expression for the form $\forall x\ \varphi$, where x is an individual variable and φ is a formula.

We use $d, e, \ldots$ to denote equations and $\varphi, \psi, \ldots$ to denote formulas.

We are primarily interested in the universal Horn formulas, which are universally quantified formulas of the form

$$\forall x_1\ \ldots\ \forall x_n\ \ e_1 \to \cdots \to e_n \to e. \tag{9.1}$$

In the system presented in this section, the quantification is implicit.

Let S denote the set of all substitutions $\sigma : X \mapsto T_\Sigma(X)$. The notation $[x/t]$ denotes the substitution that simultaneously substitutes the term t for all occurrences of the variable x in a term or formula.

The following axioms E are the axioms of classical equational logic, implicitly universally quantified.

$$\begin{aligned}
&x = x\\
&x = y \to y = x\\
&x = y \to y = z \to x = z\\
&x_1 = y_1 \to \cdots \to x_n = y_n \to f x_1 \ldots x_n = f y_1 \ldots y_n, \quad f \in \Sigma_n.
\end{aligned}$$

Besides E, we will also allow an *application theory* Δ of universal Horn formulas to serve as additional, application-specific axioms. For example, for group theory, Δ would consist of the equational axioms for groups.

We now give the deduction rules. Let A be a set of equations, $d, e, \ldots$ equations, and φ a Horn formula.

$$\vdash \sigma(\varphi), \quad \varphi \in \Delta \cup E, \quad \sigma \in S$$

$$e \vdash e$$

$$\frac{A \vdash \varphi}{A, e \vdash \varphi}$$

$$\frac{A, e \vdash \varphi}{A \vdash e \to \varphi}$$

$$\frac{A \vdash e \to \varphi \quad A \vdash e}{A \vdash \varphi}$$

The following rule is derived:

$$\frac{A \vdash e}{A[x/t] \vdash e[x/t]}$$

provided x does not occur in t. This rule obviates the need for an explicit universal quantifier and corresponding introduction and elimination rules.

One can also give annotated versions of these rules in which formulas are annotated with explicit proof terms, which are terms of the simply typed λ-calculus. Let $P = \{p, q, r, \ldots\}$ be a set of *proof variables*. A *proof term* is either

- a variable $p \in P$,
- a constant ref, sym, trans, cong_f for $f \in \Sigma$, or axiom_φ for $\varphi \in \Delta$,
- an application $\pi\tau$, where π and τ are proof terms,
- an application πt, where π is proof term and t is an individual term,
- an abstraction $\lambda p.\tau$, where p is a proof variable and τ is a proof term,
- an abstraction $\lambda x.\tau$, where x is an individual variable and τ is a proof term, or
- an expression $\mathsf{pub}\,\tau$, where τ is a proof term.

The combinator pub is just a fixed constant. Proof terms are denoted $\pi, \rho, \tau, \ldots$.

As usual in constructive mathematics, according to the Curry–Howard isomorphism, we can view proofs as constructions, formulas as types, and the deduction system as a set of typing rules. An annotated formula takes the form of a type judgement $\tau : \varphi$, where τ is a proof term. The interpretation of these constructs is the same as in the simply-typed λ-calculus.

The annotated rules are as follows.

$$\vdash \mathsf{axiom}_\varphi\, \sigma : \sigma(\varphi), \quad \varphi \in \Delta \cup E, \quad \sigma \in S$$

$$p : e \vdash p : e$$

$$\frac{A \vdash \tau : \varphi}{A,\ p : e \vdash \tau : \varphi}$$

$$\frac{A,\ p : e \vdash \tau : \varphi}{A \vdash \lambda p.\tau : e \to \varphi}$$

$$\frac{A \vdash \pi : e \to \varphi \quad A \vdash \rho : e}{A \vdash \pi\rho : \varphi}$$

Not all proof terms as defined above are well typed. In particular, abstractions over an individual variable $\lambda x.\tau$ and the pub combinator will only become relevant in Section 9.3, when we reintroduce explicit universal quantification.

9.3 A New System

The new system we present in this section builds directly on the classical system presented in Section 9.2, adding in the notion of a *library* $\mathscr{L}$. This library minimally contains all of the axioms, and we introduce rules to add and remove new theorems to and from the library along with their proofs.

In contrast to the system of Section 9.2, the system of the present section will have explicit universal quantification for all axioms and theorems in the library.

This will allow arbitrary specialization via substitution of individual terms for the quantified variables upon citation.

As in the system of Section 9.2, our system has three main syntactic categories: *individual terms*, *formulas*, and *proof terms*. Also as in Section 9.2, we will start by presenting the unannotated version of our proof system, which does not have any proof terms.

As before, let $X = \{x, y, \ldots\}$ be a set of *individual variables*, $\Sigma = \{f, g, \ldots\}$ a first-order signature, and $T_\Sigma(X) = \{s, t, \ldots\}$ the set of *individual terms* over X and Σ. Let Δ be a set of universal Horn formulas over X and Σ that serve as the application theory.

The unannotated version of our proof system consists of the following axioms and rules. We restate the equational axioms E, this time with explicit universal quantification. We use $\bar{x}$ to denote a tuple $x_1, \ldots, x_n$.

$$\forall x \;\; x = x$$
$$\forall x \, \forall y \;\; x = y \to y = x$$
$$\forall x \, \forall y \, \forall z \;\; x = y \to y = z \to x = z$$
$$\forall \bar{x} \, \forall \bar{y} \;\; x_1 = y_1 \to \ldots \to x_n = y_n \to f x_1 \ldots x_n = f y_1 \ldots y_n, f \in \Sigma$$

In the descriptions below, the separator character ; is used to distinguish proof tasks from the contents of the library. Judgements are of the form $\mathscr{L} \,;\, A \vdash \varphi$, where $\mathscr{L}$ is the current library consisting of a list of universally quantified Horn formulas of the form (9.1), A is a list of unquantified equational premises, and φ is an unquantified Horn formula. Elements of $\mathscr{L}$ are separated by commas, as are elements of A.

$$(\textbf{ident}) \;\; \frac{\mathscr{L} \,;}{\mathscr{L} \,;\, e \vdash e}$$

$$(\textbf{assume}) \;\; \frac{\mathscr{L} \,;\, A \vdash \varphi}{\mathscr{L} \,;\, A, e \vdash \varphi}$$

$$(\textbf{discharge}) \;\; \frac{\mathscr{L} \,;\, A, e \vdash \varphi}{\mathscr{L} \,;\, A \vdash e \to \varphi}$$

$$(\textbf{mp}) \;\; \frac{\mathscr{L} \,;\, A \vdash e \to \varphi \quad \mathscr{L} \,;\, A \vdash e}{\mathscr{L} \,;\, A \vdash \varphi}$$

$$(\textbf{publish}) \;\; \frac{\mathscr{L} \quad ;\, \vdash \varphi}{\mathscr{L}, \forall \bar{x} \, \varphi \,;}$$

$$(\textbf{cite}) \;\; \frac{\mathscr{L}, \forall \bar{x} \, \varphi \;\;;}{\mathscr{L}, \;\forall \bar{x} \, \varphi \;;\; \vdash \varphi[\bar{x}/\bar{t}]}$$

$$(\textbf{forget}) \;\; \frac{\mathscr{L}, \forall \bar{x} \, \varphi \,;}{\mathscr{L} \quad ;}, \quad \forall \bar{x} \, \varphi \notin \Delta \cup E$$

where in the **publish** rule, $\bar{x} = x_1, \ldots, x_n$ are the free variables of φ.

The rules of the proof system build on the classical set of rules, with the addition of the three new rules **publish**, **cite** and **forget**. We do not allow the equational and application theory axioms to be removed from the library, thus it is always be the case that $\Delta \cup E \subseteq \mathscr{L}$. The rules **publish** and **cite** serve as introduction and elimination rules for the universal quantifier, respectively, but quantifiers appear only in published theorems (i.e., those in the library). The **forget** rule is simply an ordinary weakening rule in this version; however, once annotations are added, the effect of this rule on proof terms is much more involved.

Now we add annotations. For the equational axioms and the application theory,

$$\begin{aligned}
\mathsf{ref} &: \forall x \;\; x = x \\
\mathsf{sym} &: \forall x \, \forall y \;\; x = y \to y = x \\
\mathsf{trans} &: \forall x \, \forall y \, \forall z \;\; x = y \to y = z \to x = z \\
\mathsf{cong}_f &: \forall \bar{x} \, \forall \bar{y} \;\; x_1 = y_1 \to \ldots \to x_n = y_n \to f x_1 \ldots x_n = f y_1 \ldots y_n, \quad f \in \Sigma \\
\mathsf{axiom}_\varphi &: \varphi, \quad \varphi \in \Delta.
\end{aligned}$$

Thus each axiom of equational logic and the application theory ($\Delta \cup E$) is inhabited by a constant. These type judgements are always present in $\mathscr{L}$.

In addition to these, we have the following annotated rules:

$$(\textbf{ident}) \quad \frac{\mathscr{L} \;;}{\mathscr{L} \;; p : e \vdash p : e}$$

$$(\textbf{assume}) \quad \frac{\mathscr{L} \;; A \vdash \tau : \varphi}{\mathscr{L} \;; A, p : e \vdash \tau : \varphi}$$

$$(\textbf{discharge}) \quad \frac{\mathscr{L} \;; A, p : e \vdash \tau : \varphi}{\mathscr{L} \;; A \vdash \lambda p.\tau : e \to \varphi}$$

$$(\textbf{mp}) \quad \frac{\mathscr{L} \;; A \vdash \pi : e \to \varphi \quad \mathscr{L} \;; A \vdash \rho : e}{\mathscr{L} \;; A \vdash \pi\rho : \varphi}$$

$$(\textbf{publish}) \quad \frac{\mathscr{L} \qquad\qquad\qquad \;; \vdash \tau : \varphi}{\mathscr{L}, \mathsf{pub}\, \lambda \bar{x}.\tau : \forall \bar{x} \; \varphi \;;}$$

$$(\textbf{cite}) \quad \frac{\mathscr{L}, \pi : \forall \bar{x} \; \varphi \;;}{\mathscr{L}, \pi : \forall \bar{x} \; \varphi \;; \vdash \pi \bar{t} : \varphi[\bar{x}/\bar{t}]}$$

$$(\textbf{forget}) \quad \frac{\mathscr{L}, \mathsf{pub}\, \pi : \forall \bar{x} \; \varphi \;;}{\mathscr{L}[\mathsf{pub}\, \pi/\pi] \qquad \;;}, \quad \forall \bar{x} \; \varphi \notin \Delta \cup E$$

Publication forms the universal closure of the formula and the corresponding λ-closure of the proof term before wrapping with pub. Thus published theorems, like axioms, are always closed universal formulas. The proof term is closed by binding all the free individual variables appearing in the body of the proof term in the same order as they were bound in the formula (the free individual variables in formulas and corresponding proof terms are always the same).

As in Section 9.2, the interpretation of annotated formulas is the same as in the simply-typed λ-calculus. However, our type system is somewhat more restrictive than the usual one. For example, the type system prevents a binding operator λx from occurring in the scope of a binding operator λp. This enforces the universal Horn form (9.1) for published theorems.

The constant pub is polymorphic of type $\varphi \to \varphi$. Its main purpose is to wrap proof terms to inhibit β-reduction without altering their type. An expression of the form $\mathsf{pub}\,\pi$ is called a *citation token*. Intuitively, we can think of a citation token as a short abbreviation (a name or a pointer) for the proof π in the library. Since the proof and its citation token are type-equivalent, we can use them interchangeably.

Ordinarily, when a universally quantified theorem is cited in a special case defined by a substitution $[\overline{x}/\overline{t}]$, the proof term would be specialized as well by applying it to the sequence of individual terms $t_1, \ldots, t_n$. Without the pub wrapper, proof normalization would cause those terms to be substituted for $x_1, \ldots, x_n$ in the body of the λ-expression as in ordinary β-reduction. The pub wrapper prevents this from happening, since the expression $(\mathsf{pub}\,\pi)\tau$ is in normal form.

An alternative approach might use new names and bindings for published proofs, but this would introduce namespace management issues that are largely orthogonal to the publish/cite structure and which our approach circumvents.

For an accurate complexity analysis on the size of proofs, one could define the size of proof terms inductively in some reasonable way, taking the size of citation tokens to be 1. This would reflect the fact that in practice, a proof term would only be represented once, and citations would reference the original proof by name or by pointer.

9.4 An Example

To illustrate the operation of this proof system, we will go through a simple example. Supposing we wanted to prove the theorem

$$\forall x \;\; x = fx \to x = f(fx).$$

We will provide a proof of this fact, along with the corresponding extraction of proof terms. For the first part, we will omit the library $\mathscr{L}$ for readability, but we will reintroduce it later when we show the operation of publication.

Defining

$$\begin{aligned} \pi_1 &= \mathsf{cong}_f\, x\, (fx) \\ \pi_2 &= \mathsf{trans}\, x\, (fx)\, (f(fx)), \end{aligned}$$

we have by citation of the transitivity axiom and the congruence axiom for f that

$$\vdash \pi_1 \ : \ x = fx \to fx = f(fx) \tag{9.2}$$
$$\vdash \pi_2 \ : \ x = fx \to fx = f(fx) \to x = f(fx). \tag{9.3}$$

From (9.3) and **(assume)**, we have

$$p : x = fx \vdash \pi_2 : x = fx \to fx = f(fx) \to x = f(fx). \tag{9.4}$$

Also, by **(ident)**,

$$p : x = fx \vdash p : x = fx. \tag{9.5}$$

Applying **(mp)** with premises (9.4) and (9.5) gives

$$p : x = fx \vdash \pi_2\, p : fx = f(fx) \to x = f(fx). \tag{9.6}$$

Similarly, from (9.2) and **(assume)**, we have

$$p : x = fx \vdash \pi_1 : x = fx \to fx = f(fx), \tag{9.7}$$

and applying **(mp)** with premises (9.5) and (9.7) gives

$$p : x = fx \vdash \pi_1\, p : fx = f(fx). \tag{9.8}$$

Now applying **(mp)** with premises (9.6) and (9.8), we obtain

$$p : x = fx \vdash \pi_2\, p(\pi_1\, p) : x = f(fx), \tag{9.9}$$

and we conclude from **(discharge)** that

$$\vdash \lambda p.\pi_2\, p\, (\pi_1\, p) : x = fx \to x = f(fx). \tag{9.10}$$

We can now publish the universal closure of (9.10) using the publication rule, which adds the annotated theorem

$$\mathsf{pub}\, (\lambda x.\lambda p.\pi_2\, p\, (\pi_1\, p)) : \forall x \ \ x = fx \to x = f(fx) \tag{9.11}$$

to the library.

Now we show how (9.11) can be cited in a special case by proving the theorem

$$\forall y \ \ gy = f(gy) \to gy = f(f(gy)). \tag{9.12}$$

This is a more specific version of (9.11) obtained by substituting gy for x. We start by citing (9.11) with the term gy using the rule **(cite)**, which gives

$$\mathsf{pub}\, (\lambda x.\lambda p.\pi_2\, p\, (\pi_1\, p))(gy) : gy = f(gy) \to gy = f(f(gy)). \tag{9.13}$$

Publishing this theorem using **(publish)** results in the annotated theorem

$$\mathsf{pub}\,(\lambda y.\mathsf{pub}\,(\lambda x.\lambda p.\pi_2\,p\,(\pi_1\,p))(gy)) : \forall y\ \ gy = f(gy) \to gy = f(f(gy)) \quad (9.14)$$

being added to the library.

Now suppose we wish to use the **(forget)** rule to forget the original theorem (9.11) that was cited in the proof of (9.12). This removes (9.11) from the library and strips the pub combinator from all citations. The theorem (9.14) becomes

$$\mathsf{pub}\,(\lambda y.(\lambda x.\lambda p.\pi_2\,p\,(\pi_1\,p))(gy)) : \forall y\ \ gy = f(gy) \to gy = f(f(gy)) \quad (9.15)$$

The theorem itself remains in the library unchanged, but its proof is no longer in normal form, since the inner pub combinator has been stripped. Normalizing the proof, we obtain

$$\mathsf{pub}\,(\lambda y.\lambda p.\pi_2[x/gy]\,p\,(\pi_1[x/gy]\,p)) : \forall y\ \ gy = f(gy) \to gy = f(f(gy)),$$

where

$$\begin{aligned}\pi_1[x/gy] &= \mathsf{cong}_f\,(gy)\,(f(gy))\\ \pi_2[x/gy] &= \mathsf{trans}\,(gy)\,(f(gy))\,(f(f(gy))).\end{aligned}$$

The proof now has the specialization of the proof of (9.11) "inlined" into it. This is equivalent to what we would have obtained had we set out to prove (9.12) directly, without appealing to the more general theorem (9.11) first.

9.5 Related and Future Work

Proof generalization, proof reuse, and mathematical knowledge management are active areas of research. Much of the work on proof generalization and reuse is oriented toward heuristic methods for discovering simple modifications of old proofs that apply in new but similar situations. Schairer et al. [14, 15] have suggest replaying tactics to develop proofs by analogy. Hutter [8, 9] has given a system of proof annotations and rules for manipulating them. The annotations are used to include planning information in proofs to help guide the proof search. Kolbe and Walther [10, 11] study the process of proof generalization by abstracting existing proofs to form *proof shells*. Their approach involves replacing occurrences of function symbols by second-order variables. Felty and Howe [7] also suggest a system of proof reuse using higher-order unification and metavariables to achieve abstraction. Melis and Whittle [12] study proof by analogy, focussing on the process of adapting existing proofs to new theorems with a similar structure. Piroi and Buchberger [5, 13] present a graphical environment for editing mathematics and managing a mathematical knowledge library. Allen et al. [4] also propose a structure for a formal digital library and discuss the problem of naming conflicts.

It would be interesting to explore the possibility of identifying similar proofs and finding common generalizations in a more proof-theoretic context such as the

publication/citation mechanism presented in this paper. It would also be useful to extend the system to handle full first-order and higher-order logics.

One area of recent progress is the proof-theoretic representation of tactics [1, 2]. Another recent advance is the enhancement of the proof-theoretic apparatus to better capture natural dependencies among theorems, lemmas, and corollaries in the library and the locality of definitions. Most libraries are flat, which does not adequately capture the richness of mathematical knowledge. Recent progress in this direction has been made by Aboul-Hosn and Andersen [3], who present a hierarchical representation along with natural proof rules for restructuring.

Acknowledgements We are indebted to Kamal Aboul-Hosn, Samson Abramsky, Terese Damhøj Andersen, and Anil Nerode for valuable ideas and comments. This work was supported in part by NSF grant CCF-0635028. Any views and conclusions expressed herein are those of the authors and do not necessarily represent the official policies or endorsements of the National Science Foundation or the United States government.

References

1. K. Aboul-Hosn. A proof-theoretic approach to tactics. In J.M. Borwein and W.M. Farmer, editors, *Proceedings of the 5th International Conference on Mathematical Knowledge Management (MKM'06)*, volume 4108 of *Lecture Notes in Computer Science*, pages 54–66. Springer, Berlin, Germany, August 2006.
2. K. Aboul-Hosn. *A Proof-Theoretic Approach to Mathematical Knowledge Management*. PhD thesis, Cornell University, January 2007.
3. K. Aboul-Hosn and T.D. Andersen. A proof-theoretic approach to hierarchical math library organization. In *Proceedings of the 4th International Conference on Mathematical Knowledge Management (MKM'05)*, pages 1–16. Springer, Berlin, Germany, October 2005.
4. S. Allen, M. Bickford, R. Constable, R. Eaton, C. Kreitz, and L. Lorigo. *FDL: A prototype formal digital library*, 2002. http://www.nuprl.org/FDLproject/02cucs-fdl.html
5. B. Buchberger. Mathematical knowledge management using theorema. In B. Buchberger, O. Caprotti, editors, *1st International Conference on Mathematical Knowledge Management (MKM 2001)*, RISC-Linz, A-4232 Schloss Hagenberg, Austria, September 24–26, 2001.
6. R.L. Constable, S.F. Allen, H.M. Bromley, W.R. Cleaveland, J.F. Cremer, R.W. Harper, D.J. Howe, T.B. Knoblock, N.P. Mendler, P. Panangaden, J.T. Sasaki, and S.F. Smith. *Implementing Mathematics with the Nuprl Development System*. Prentice-Hall, Englewood Cliffs, NJ, 1986.
7. A. Felty and D. Howe. Generalization and reuse of tactic proofs. In *Proceedings of the 5th International Conference on Logic Programming and Automated Reasoning (LPAR94)*. Springer, Berlin, Germany, 1994.
8. D. Hutter. Structuring deduction by using abstractions. In T. Ellman, editor, *Proceedings of the International Symposion on Abstraction, Reformulation, and Approximation (SARA98)*, pages 72–78. Pacific Grove, CA, 1998.
9. D. Hutter. Annotated reasoning. *Annals of Mathematics and Artificial Intelligence, Special Issue on Strategies in Automated Deduction*, 29:183–222, 2000.
10. T. Kolbe and C. Walther. Reusing proofs. In *Proceedings of the 11th European Conference on Artificial Intelligence (ECAI-11)*, pages 80–84. Seattle, WA, May 21–23, 1994.
11. T. Kolbe and C. Walther. Proof management and retrieval. In *Proceedings of the Workshop on Formal Approaches to the Reuse of Plans, Proofs and Programs (IJCAI-14)*. Montreal, CA, 1995.

12. E. Melis and J. Whittle. Analogy in inductive theorem proving. *Journal of Automated Reasoning*, 22(2):117–147, 1999.
13. F. Piroi and B. Buchberger. An environment for building mathematical knowledge libraries. In A. Trybulec, A. Asperti, G. Bancerek, editor, *Proceedings of the 3rd International Conference Mathematical Knowledge Management (MKM 2004)*, volume 3119 of *Lecture Notes in Computer Science*. Springer, Berlin, Germany, September 2004.
14. A. Schairer. *A Technique for Reusing Proofs in Software Verification*. PhD thesis, FB 14 (Informatik) der Universität des Saarlandes und Institut A für Mechanik der Universität Stuttgart, Stuttgart, AT, 1998.
15. A. Schairer, S. Autexier, and D. Hutter. A pragmatic approach to reuse in tactical theorem proving. *Electronic Notes in Theoretical Computer Science*, 58(2), 2001.
16. A. Trybulec. The Mizar system. http://mizar.uwb.edu.pl/system/