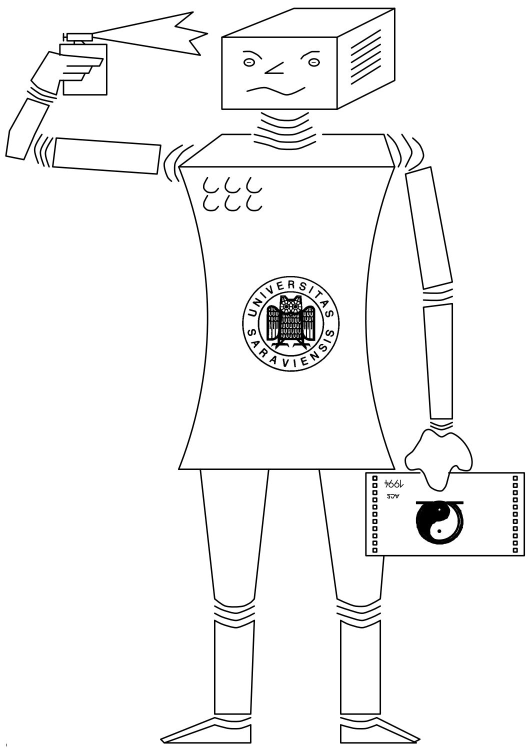
UNIVERSITÄT DES SAARLANDES
FACHBEREICH INFORMATIK
D-66041 SAARBRÜCKEN
GERMANY
WWW: http://js-sfbsun.cs.uni-sb.de/pub/www/



PROVERB – A System Explaining Machine-Found Proofs

Xiaorong Huang

Published as: Proc. of 16th Annual Conference of the Cognitive Science Society,Atlanta, Georgia, 1994

PROVERB– A System Explaining Machine-Found Proofs

Xiaorong Huang
Fachbereich Informatik, Universität des Saarlandes
Postfach 15 11 50, 66041 Saarbrücken, Germany
E-mail: huang@cs.uni-sb.de

Abstract

This paper outlines an implemented system called *PROVERB* that explains machine-found natural deduction proofs in natural language. Different from earlier works, we pursue a *reconstructive* approach. Based on the observation that natural deduction proofs are at a too low level of abstraction compared with proofs found in mathematical textbooks, we define first the concept of so-called *assertion level* inference rules. Derivations justified by these rules can intuitively be understood as the application of a definition or a theorem. Then an algorithm is introduced that *abstracts* machine-found ND proofs using the assertion level inference rules. Abstracted proofs are then verbalized into natural language by a *presentation module*. The most significant feature of the presentation module is that it combines standard hierarchical text planning and techniques that locally organize argumentative texts based on the derivation relation under the guidance of a focus mechanism. The behavior of the system is demonstrated with the help of a concrete example throughout the paper.

1 Introduction

This paper describes *PROVERB*, a system that presents proofs found by an automated reasoning system in natural language. The main aim is to provide an overview of the entire spectrum of a *reconstructive* presentation process. To achieve this, we are often forced to sacrifice precise descriptions to the global architecture.

1.1 Identifying the Problem

Similar to the *reconstructive* approach employed by the explanation component of some expert systems (Wick and Thompson 92), efforts were made to transform proofs from machine-oriented formalisms into more natural formalism (Andrews 80, Miller 83, Pfenning 87, Lingenfelder 90). As the target formalism, usually a variation of the natural deduction (ND) proof first proposed by G. Gentzen (Gentzen 35) is chosen. Until recently the reconstruction stops here and the resulting ND proofs are used as inputs by natural language generators (Chester 76, McDonald 83, Edgar & Pelletier 93). In general, the presentation of ND proofs has been carried out by resorting to *ordering*, *pruning*, and *augmentation*.

All these verbalizations suffer from the same problem: The derivations they convey are exclusively at the level of the inference rules of the ND calculus. In contrast to informal proofs found in standard mathematical textbooks, such proofs are composed of derivations familiar from elementary logic, where the focus of attention is on syntactic manipulations rather than on the underlying semantic ideas. The main problem, we believe, lies on the lack of intermediate structures of ND proofs, which allow atomic justifications at a higher level of abstraction.

PROVERB solves this problem by carrying the reconstructive approach one step further. The first section below defines a new intermediate representation, called *assertion level* inference rules. Next we illustrate the reconstruction of more abstract proofs from machine-found ND proofs using assertion level rules. Subsequently, we sketch out a computational model adapts and combines several established NL generation techniques for the verbalization of a abstracted proof.

2 Abstracting ND Proofs to the Assertion Level

Our analysis on proofs in mathematical textbooks shows that most derivations are justified in terms of the application of a definition or a theorem, collectively called an *assertion* (Huang 92). In this section, we first introduce a computational model for informal mathematics. According to this model, assertion level justifications can be used both for compound proof segments exhibiting certain syntactical structure, and for atomic steps derived by assertion level inference rules. Then we turn to an abstraction process that reconstructs a proof using assertion level rules.

2.1 Compound proof segments and assertion level rules

Our computational model for *informal mathematical reasoning* basically follows the psychological models for reasoning based the natural logic hypothesis (Braine 78, Rips 1983), and the proof planning framework proposed by A. Bundy (1988). According to this model, our intuitive notion of the application of an assertion is either achieved by constructing an ND proof segment that satisfies certain structural constraint, or by the application of an assertion level inference rule.

Figure 1 is an example of a compound proof segment that infers $a_1 \in F_1$ from $U_1 \subset F_1$ and $a_1 \in U_1$ by applying the definition of subset encoded as in the leaf with the label $\mathcal{A}$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{A}: \ \forall_{S_1,S_2} S_1 \subset S_2 \Leftrightarrow (\forall_x x \in S_1 \Rightarrow x \in S_2)}{U_1 \subset F_1 \Leftrightarrow (\forall_x x \in U_1 \Rightarrow x \in F_1)}\forall E}{U_1 \subset F_1 \Rightarrow (\forall_x x \in U_1 \Rightarrow x \in F_1)}\Leftrightarrow E, \quad U_1 \subset F_1}{\forall_x x \in U_1 \Rightarrow x \in F_1}\Rightarrow E}{a_1 \in U_1 \Rightarrow a_1 \in F_1}\forall E, \quad a_1 \in U_1}{a_1 \in F_1}\Rightarrow E$$

Figure 1: Natural Expansion 1 for Subset Definition

The procedure that applies assertions by constructing a compound proof segment is specified in terms of a so-called *decomposition-composition* constraint imposed on such proof segments identified in our preliminary empirical study on mathematical proofs (Huang 92). To illustrate this constraint, we first introduce two definitions.

Definition: An inference rule of the form $\frac{\triangle \vdash F, \triangle \vdash P_1, \ldots, \triangle \vdash P_n}{\triangle \vdash Q}$ is a *decomposition* rule with respect to the formula schema F, if all applications of it, written as $\frac{\triangle \vdash F', \triangle \vdash P'_1, \ldots, \triangle \vdash P'_n}{\triangle \vdash Q'}$ satisfy the following condition: each $P'_1, \ldots, P'_n$ and Q' is

- a proper subformula of F', or
- a specialization of F' or of one of its proper subformula, or
- a negation of one of the first two cases.

Under this definition, $\wedge D, \Rightarrow D, \forall D$ are the only elementary decomposition rules in the natural deduction calculus $\mathcal{NK}$. Compare Figure 1 for the meaning of the rules.

Definition: An inference rule of the form $\frac{\triangle \vdash P_1, \ldots, \triangle \vdash P_n}{\triangle \vdash Q}$ is called a *composition* rule if all applications of it, written as $\frac{\triangle \vdash P'_1, \ldots, \triangle \vdash P'_n}{\triangle \vdash Q'}$, satisfy the following condition: each $P'_1, \ldots P'_n$ is

- a proper subformula of Q', or
- a specialization of Q' or of one of its proper subformula, or
- a negation of one of the first two cases.

Roughly speaking, the *decomposition-composition* constraint requires that a proof segment applying an assertion $\mathcal{A}$ consists primarily of a linear decomposition of $\mathcal{A}$. As illustrated in Figure 1, this is carried out along the branch from $\mathcal{A}$ to the root by applying decomposition rules. Other premises involved in this series of decompositions (the leaves $U_1 \subset F_1$ and $a_1 \in U_1$ in Figure 1 for instance) can be constructed by compositions (not used in Figure 1). For a precise definition of this constraint, see (Huang 92). In the sequel, proof segments satisfying this constraint will be referred to as the *natural expansion* of the corresponding assertion level justification. This constraint is closely related to one of Johnson-Laird's *effective procedures* (Johnson-Laird 1983), accounting for spontaneous daily reasoning.

Assertion level justifications are also used for proof steps derived by an assertion level inference rule. There are two ways for acquiring such rules in our computational model: learning by *chunking-and-variablization* and learning by *contraposition* (Huang 94b).

First, we assume that patterns of repeated applications of an assertion may be remembered as new rules, similar to the *chunking* operation in the cognitive architecture Soar (Newell 90). On account of this, assertion level rules are also referred to as *compound* rules. We continue with our subset example to illustrate this.

Example 1 (Continued):

Suppose that a reasoner has just derived $a_1 \in F_1$ from the premises $a_1 \in U_1$ and $U_1 \subset F_1$ by constructing the proof tree in Figure 1. Our assumption is that apart from merely drawing a concrete conclusion from the premises, possibly he learns the following compound rule as well:

$$\frac{\triangle \vdash a \in U, \triangle \vdash U \subset F}{\triangle \vdash a \in F}$$

where a, U and F are metavariables standing for object variables. Note that this rule is obtained by variablization and removing the intermediate steps in Figure 1.

The second way of acquiring assertion level rules can be viewed as a generalized *contraposition*. For instance, after the acquisition of rule above, the rules

$$\frac{\triangle \vdash a \in U, \triangle \vdash a \notin F}{\triangle \vdash U \not\subset F} \qquad \frac{\triangle \vdash a \notin F, \triangle \vdash U \subset F}{\triangle \vdash a \notin U}$$

can be derived as contrapositions.

In (Huang 94b), it is shown that a finite set of compound rules can be constructed for each assertion, so that this set covers all possible applications of this assertion, which can be achieved by constructing natural expansions. The two ways of applying assertions are therefore logically equivalent. The set of assertion level rules for a typical mathematical assertion can usually be represented as one or two proof tree schemata satisfying the decomposition-composition constraint (Huang 94b). These rules are used to abstract ND proofs.

2.2 Abstracting ND Proofs to the Assertion Level

This section describes an algorithm that replaces as many compound proof segments in machine-found ND proofs as possible, by atomic derivations justified by assertion level rules. Note that proof segments replaced may contain machine-generated detours and redundancies, and are not necessarily natural expansions.

Algorithm: Go through the proof tree starting from the root, for each proof node N,

1) Choose as the set of assertions AS the definitions and theorems contributing to the proof of N, namely the

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{Subset : \forall_{S_1,S_2} S_1 \subset S_2 \Leftrightarrow \forall_x x \in S_1 \Rightarrow x \in S_2}{[a'] : U_1 \subset F_1 \Leftrightarrow \forall_x x \in U_1 \Rightarrow x \in F_1}}{[b'] : U_1 \subset F_1 \Rightarrow \forall_x x \in U_1 \Rightarrow x \in F_1} \quad , [p_1] : U_1 \subset F_1}{[c'] : \forall_x x \in U_1 \Rightarrow x \in F_1}}{[d'] : b \in U_1 \Rightarrow b \in F_1} \quad , b \notin F_1}{b \notin U_1} \quad , [p_2] : b \in U_1}{\perp}}{[e'] : b \in F_1}
$$

Figure 2: A Natural Deduction Proof with Detours

leaves of the subtree rooted by N, which are definitions or theorems used.

2) Among the nodes in the subtree rooted by N, test if there exist nodes $p_1, \ldots, p_n$, from which N can be derived by an assertion level rule associated with an assertion $\mathcal{A}$ in AS. If successful, reduce the input proof so that N has $p_1, \ldots, p_n$ as its direct children and $\mathcal{A}$ as its new justification.

For example, the proof segment Figure 2 can be abstracted to the following step justified by an assertion level rule learned from Figure 1:

$$
\frac{[p_1] : U_1 \subset F_1, [p_2] : b \in U_1}{[e'] : b \in F_1} Subset
$$

The restrictive choice of AS and the search for $p_1, \ldots, p_n$ in a breath-first way sacrifice optimal solutions to efficiency. Neatly written ND proofs are reduced to 1/3 of their original proof nodes. The best-case complexity is linear. Most significant reduction is observed with input proofs which contain machine-generated detours and redundancies. The algorithm performs poorly on proofs which are mainly indirect, where in most of the node only $\perp$ (contradiction) is derived. The worst-case complexity is $O(n^2)$ (Huang 94b).

Figure 3 is a proof abstracted from an input proof of 134 lines, generated in the proof development environment Ω–MKRP(Huang et al 94) the theorem below:

Theorem: Let F be a group and U a subgroup of F, if 1_U is a unit element of U, then $1 = 1_U$.

To illustrate the difference between derivations at the assertion level and logic level, we want to indicate that the best ND proof for the assertion level derivation in step 7 corresponds to a compound proof given in Figure 1. In a machine-generated ND proof, it can be more complex since such a proof often contains machine-generated detours, see Figure 2. It is given in a linearized format, where the last column contains the justification as well as the premises. Eleven of the remaining fifteen steps are at the assertion level. The rest is justified by ND rules of more structural import: they introduce new temporary hypothesis and then discharge them (the *Hyp* and the *Choice rule* in this example). These steps are usually presented later explicitly. Groups of trivial steps instantiating quantifiers or manipulating logical connectives are largely abstracted to assertion level steps. The formula $solution(a, b, c, F, *)$ should be read as "c is a solution of the equation $a * x = b$ in F."

3 The Verbalization of an Abstracted Proof

The presentation module of *PROVERB* accepts an abstracted proof as input, and produces a proof in natural language. It is cast as a two stage process: The *macroplanner* chooses a sequence of proof communicative acts (PCAs), being speech acts in this domain of application. The *microplanner* makes more syntactic decisions.

The macroplanner combines hierarchical planning (Hovy 88, Moore 89, Reithinger 91) and local organization of text (Sibun 90) in a uniform planning framework. While hierarchical planning views language as planned behavior and maps a goal into subgoals, local organization simulates the more spontaneous part of text generation. Under the latter mode, local structures suggest the next objects available, once a discourse is started.

3.1 The Framework of the Text Planner

PROVERB combines the two presentation modes by encoding communication knowledge for both modes as plan operators, called top-down and bottom-up presentation operators respectively, in a uniform planning framework. Since top-down presentation operators embody explicit communicative norms, they are given a higher priority than the bottom-up ones. A bottom-up operator is chosen only when no top-down presentation operator applies. The overall planning framework is realized by a function `Present`. It is first called with the entire proof as the presentation task. The execution of a top-down presentation operator generates new subtasks by calling it recursively. Furthermore, we assume that every discourse segment produced by a call to "`Present`" forms an attentional unit in the discourse model (compare the subsection on reference choices).

4 Top-Down Presentation operators

In contrast with operators employed in RST-based planners that split pending goals according to rhetorical structures, our operators encode standard schemata for presenting proofs of some particular structures. The top-down presentation operators can roughly be divided into two categories:

- those containing complex schemata for the presentation of proofs of a specific pattern,

NNo	S;D		Formula	Reason
1.	1;	$\vdash$	$group(F,*) \wedge subgroup(U,F,*) \wedge unit(F,1,*) \wedge unit(U,1_U,*)$	(Hyp)
2.	1;	$\vdash$	$U \subset F$	(Def-subgroup 1)
3.	1;	$\vdash$	$1_U \in U$	(Def-unit 1)
4.	1;	$\vdash$	$\exists_x x \in U$	($\exists$ 3)
5.	;5	$\vdash$	$u \in U$	(Hyp)
6.	1;5	$\vdash$	$u * 1_U = u$	(Def-unit 1 5)
7.	1;5	$\vdash$	$u \in F$	**(Def-subset 2 5)**
8.	1;5	$\vdash$	$1_U \in F$	(Def-subset 2 3)
9.	1;5	$\vdash$	$semigroup(F,*)$	(Def-group 1)
10.	1;5	$\vdash$	$solution(u,u,1_U,F,*)$	(Def-solution 6 7 8 9)
11.	1;5	$\vdash$	$u * 1 = u$	(Def-unit 1 7)
12.	1;5	$\vdash$	$1 \in F$	(Def-unit 1)
13.	1;5	$\vdash$	$solution(u,u,1,F,*)$	(Def-solution 7 11 12 9)
14.	1;5	$\vdash$	$1 = 1_U$	(Th-solution 11 10 13)
15.	1;	$\vdash$	$1 = 1_U$	(Choice 4 14)

Figure 3: Abstracted Proof about Unit Element of Subgroups

- those embodying general presentation norms, concerning splitting proofs and ordering subgoals.

Due to space restrictions, we only elaborate on one top-down operator devised for proof segments containing cases. A corresponding proof schema is shown in Figure 4.

$$\frac{\begin{array}{c}\overline{\quad\cdots\quad}\\ \vdots \\ ?L_4 : \triangle \vdash F \vee G\end{array} , \quad \begin{array}{c}\overline{\quad F \vdash F\quad}\\ \vdots \\ ?L_2 : \triangle, F \vdash Q\end{array} , \quad \begin{array}{c}\overline{\quad G \vdash G\quad}\\ \vdots\\ ?L_3 : \triangle, G \vdash Q\end{array}}{?L_1 : \triangle \vdash Q}\,\mathrm{CASE}$$

Figure 4: A Proof Schema Involving Cases

Under two circumstances a writer may recognize that he is confronted with a proof segment containing cases. First, when a corresponding subproof is the current presentation task, tested by (task $?L_1$)[1]. Second, when a disjunction has just been presented, tested by (local-focus $?L_4$). Under both circumstances there is a communication norm motivating the writer to try first to present the part leading to $F \vee G$ (in the second case this subgoal has already been achieved), and then proceeds with the two cases. Certain PCAs are used to mediate between parts of a proof. This procedure is captured by the operator below.

Case-Implicit

- Proof: the proof schema in Figure 4
- Applicability Condition: ((task $?L_1$) $\vee$ (local-focus $?L_4$)) $\wedge$ (not-conveyed ($?L_2$ $?L_3$))
- Acts: `(if (not-conveyed` $?L_4$`)`
  `(present` $?L_4$`))` `;subgoal 1`
  `(Case-First` F`)`
  `(present` $?L_2$`)` `;subgoal 2`
  `(Case-Second` G`)`
  `(present` $?L_3$`)` `;subgoal 3`
  `(Set-Conveyed :Conclusion` $?L_1$`))`
- features: (top-down compulsory implicit)

[1]Labels stand for the corresponding nodes

This operator posts three new subgoals. The entire verbalization is of the pattern below:

The verbalization of the subproof leading to $F \vee G$ rooted by $?L_4$. (subgoal 1)
"First, let us consider the first case by assuming F."
The verbalization of the subproof rooted by $?L_2$. (subgoal 2)
"Next, we consider the second case by assuming G."
The verbalization of the subproof rooted by $?L_3$. (subgoal 3)

There are two kinds of feature values. "Implicit" is a stylistic one, indicating that the splitting into the three subgoals is not made explicit in the verbalization. In its explicit dual **Case-Explicit** this can be achieved by adding the PCA below to the beginning of the ACTs slot.

`(Subgoal Goal:` *Label_1*
`Subgoals: (`$?L\_4$ $?L\_2$ $?L\_3$`))`

which produces the verbalization:

"To prove Q, let us first prove $F \vee G$, and then consider the two cases separately."

The feature value "compulsory" indicates the specificity of this operator: If the operator is applicable, and its style conforms to the global style (which can be specified by the user in *PROVERB*), this operator should be chosen. Besides "compulsory", there are two weaker stages reflecting the specificity: "specific" and "general".

Twelve top-down presentation operators that embody communication schemata for proofs exhibiting a particular structure are currently integrated in *PROVERB*. The other five realize more general ordering and splitting principles (Huang 94a, Huang 94d).

4.1 Bottom-up Presentation

The *bottom-up presentation* process simulates the unplanned part of proof presentation. Instead of splitting presentation goals into subgoals, it follows the local derivation relation to find a proof node or a subproof to be presented next. In this sense, it is similar to the local organization techniques used in (Sibun 90). Only when no top-down operator applies, will a bottom-up operator be chosen.

4.2 The Local Focus

The node to be presented next is suggested by the *local focus*, the proof node last presented. Although logically any proof node having the local focus as a child could be chosen for the next step, usually the one with the greatest semantic overlap with the *focal centers* is preferred. Focal centers are semantic objects mentioned in the local focus. This is based on the observation that if one has proved a property about some semantic objects, one tends to continue to talk about these objects before turning to new objects. Let us examine the following situation.

$$\frac{\dfrac{[1]:\ P(a,b)}{[2]:\ Q(a,b)},\quad \dfrac{[1]:\ P(a,b),\ [3]:\ S(c)}{[4]:\ R(b,c)}}{[5]:\ Q(a,b)\wedge R(b,c)}$$

Assume that node [1] is the local focus, the set $\{a,b\}$ is therefore the focal centers. [3] is a previously presented node and the proof rooted by [5] is the current task. [2] is chosen as the next node, since it does not (re)introduce any new semantic element and its overlap with the focal centers ($\{a,b\}$) is larger than that of [4] with the focal centers ($\{b\}$).

4.3 The Bottom-Up Presentation Operators

Due to space restrictions, we only examine the most frequently used bottom-up operator, which presents one step of derivation:

Derive-Bottom-Up

- Proof: $\frac{?Node_1,\ldots,?Node_n}{?Node_{n+1}}\,?M$
- Applicability Condition: (eq next-node $?Node_{n+1}$) $\wedge$ (conveyed ($?Node_1$, ..., $?Node_n$))
- Acts: (Derive Derived-Formula: $?Node_{n+1}$
Reasons: $?Node_1, \cdots, ?Node_n$
Method: $?M$)
- Features: (bottom-up general explicit detailed)

The precondition says, a node $Node_{n+1}$ can be chosen as the next to be presented, if all its premises have already been conveyed, and if it is recommended by the focus mechanism. Only one piece of PCA is generated. If the Derived-Formula, Reasons and Method slots are instantiated by $a \in S_1$, ($a \in S_2, S_1 \in S_2$), and *def-subset* respectively, the following verbalization can be produced (depending on the context):

> "Since a is an element of S_1, and S_1 is a subset of S_2, a is an element of S_2 according to the definition of subset."

4.4 Reference Choices

Macroplanning produces a sequence of PCAs. Our microplanner is restricted to the treatment of the *reference choices* for the inference methods and for the previously presented intermediate conclusions. While the former depends on the static salience related to the domain knowledge (Pattabhiraman & Cercone 93), the latter is very similar to the subsequent references, and is therefore sensitive to the context, in particular to its segmentation into an attentional hierarchy (Reichman 85, Grosz & Sidner 86). Below is an example of a PCA enriched with reference choices for the reasons and the method by the microplanner, see (Huang 94c) for more details.

```
(Derive Reasons: ((((ELE a U) explicit))
                   ((SUBSET U F) omit))
        Conclusion: (ELE a F)
        Method: (Def-Subset omit))
```

With the help of a dictionary, this is translated into the input language of our surface generator based on the TAG formalism, which finally produces the utterance:

> "Since a is an element of U, a is an element of F."

Note that, only the reason labeled as "explicit" is verbalized. Finally, to demonstrate the quality of proofs currently generated by *PROVERB*, the complete output for the abstracted proof in Figure 3 2 is listed below:

> " (1) Let F be a group, U be a subgroup of F, 1 be a unit element of F and 1_U be a unit element of U. (2) According to the definition of unit element, $1_U \in U$. (3) Therefore there is an X, $X \in U$. (4) Now suppose that u_1 is such an X. (5) According to the definition of unit element, $u_1 * 1_U = u_1$. (6) Since U is a subgroup of F, $U \subset F$. (7) Therefore $1_U \in F$. (8) Similarly $u_1 \in F$, since $u_1 \in U$. (9) Since F is a group, F is a semigroup. (10) Because $u_1 * 1_U = u_1$, 1_U is a solution of the equation $u_1 * X = u_1$. (11) Since 1 is a unit element of F, $u_1 * 1 = u_1$. (12) Since 1 is a unit element of F, $1 \in F$. (13) Because $u_1 \in F$, 1 is a solution of the equation $u_1 * X = u_1$. (14) Since F is a group, $1_U = 1$ by the uniqueness of solution. (15) This conclusion is independent of the choice of the element u_1. "

5 Conclusion

This paper presents a system that explains machine-found proofs following a *reconstructive* approach. The power of our architecture is derived in large part from the intermediate representation, namely, natural deduction style proofs at the *assertion level*. In contrast to original ND proofs, where the focus of attention is placed on syntactic manipulations, proofs reconstructed at the assertion level contain mostly inferences in terms of semantically meaningful operators that apply a definition or a theorem valid in the context.

The most important feature of our presentation module is that hierarchical planning and unplanned spontaneous presentation are integrated in a complementary way. Based on explicit communicative knowledge, the former splits a presentation task into subtasks. The latter chooses a proof node or a subproof to be presented next under the guidance of the local focus mechanism. Distinguishing between planned and unplanned generation also leads to a natural segmentation of the discourse. This, in turn, provides an appropriate basis for a discourse theory which handles the *reference choices*.

A prototype of *PROVERB* has been implemented in Allegro Common Lisp with CLOS. While the abstraction module and the main part of the text planner have been completed, the current interface to our surface generator is very simple and is currently under extension. The abstraction module can also be used as a stand-alone utility.

Compared with proofs found in a typical mathematical textbook, the output of *PROVERB* is still tedious and inflexible. The tediousness is largely ascribed to the lack of *plan level* knowledge of the input proofs, which distinguishes crucial steps from unimportant details. Therefore, we need both a more full-fledged model for human informal mathematics, as well as sophisticated plan recognition techniques, which must be incorporated into the reconstruction process. The inflexibility of text currently produced is partly inherited from the schemata-based approach, for which a fine-grained planning in terms of single PCAs might be a remedy. It is also partly due to the fixed lexicon choice, which we are currently reimplementing. Finally, although it is hard to judge the naturalness of the texts generated by *PROVERB* by comparing with naturally occuring mathematical proofs since they differ still significantly with respect to the level of abstraction, it might be useful to build a small corpus as a standard.

6 Acknowledgments

This work was supported by the Deusche Forschungsgemeinschaft, SFB 314 (D2). Thanks are due to Manfred Kerber and Daniel Nesmith, who read earlier versions of parts of this paper.

7 References

Andrews, P. B. (1980), Transforming Matings into Natural Deduction Proofs. In *Transforming Matings into Natural Deduction Proofs* (pp 281-292). Springer.

Bundy, A. (1988). The use of explicit plans to guide inductive proofs. In *Proceedings of 9th CADE.* Springer.

Edgar, A. and Pelletier, J. P. (1993). Natural Language Explanation of Natural Deduction Proofs. In *Proc. of the first Conference of the Pacific Association for Computational Linguistics.* Simon Fraser University, Vancouver, Canada.

Gentzen, G. (1935). Untersuchungen über das logische Schließen I. *Mathematische Zeitschrift*, 39, 176–210.

Grosz, B. J. & Sidner, C. L. (1986), Attention, Intentions, and The Structure of Discourse. *Computational Linguistics* 12(3), 175-204.

Hovy, E. H. (1988). *Generating Natural Language under Progmatic Constraints.* Lawrence Erlbaum Associates.

Huang, X., Kerber, M., Kohlhase M., Melis, E., Nesmith, D., and Richts J. and Siekmann, J. (1994) Ω–MKRP– A Proof Development Environment. In *Proc. of 12th CADE.* Springer, forthcoming.

Huang, X (1992) Applications of Assertions as Elementary Tactics in Proof Planning, In V. Sgurev & B. du Boulay (Eds), Artificial Intelligence V - Methodology, Systems, Applications (pp 25-34). Elsevier Science.

Huang, X (1994a) *A Reconstructive Approach to Human-Oriented Proof Presentation.* Doctoral dissertation. Saarbücken: Universität of the Saarland, forthcoming.

Huang, X (1994b) Reconstructing Proofs at the Assertion Level. In *Proc. of 12th CADE.* Springer, forthcoming.

Huang, X (1994c) Planning Reference Choices for Argumentative Texts. In *Proc. of 7th international workshop on natural language generation.* forthcoming.

Huang, X (1994d) Planning Argumentative Texts. In *Proc. of 15th International Conference on Computational Linguistics*, forthcoming.

Johnson-Laird, Ph. N. (1983) *Mental Models*, Harvard Univ. Press.

Lingenfelder, C. (1989) Structuring Computer Generated Proofs. In *Proc. of IJCAI-89*, Morgan-Kaufmann.

McDonald, D. D. (1983) Natural language generation as a computational problem. In *Brady/Berwick: Computational Models of Discourse.* MIT PRESS.

McKeown, K. R. (1985) *Text Generation.* Cambridge Univ. Press.

Miller, D. A. (1983), Proofs in Higher-Order Logic. Doctoral dissertaion. Pittsburgh: CMU.

Moore, J. D. (1989) *A Reactive Approach to Explanation in Expert and Advice-Giving Systems.* PhD thesis, Univ. of California.

Newell, A. (1990) *Unified Theories in Cognition.* Harvard Univ. Press.

Pattabhiraman, T. & Cercone, N (1993) Decision-Theoretic Salience Interactions in Language Generation. In *Proc. IJCAI-93*(pp 1246-1252). Morgen Kaufmann.

Pfenning, F. (1987) *Proof Transformation in Higher-Order Logic.* Doctoral dissertaion. Pittsburgh: CMU.

Reichman, R. (1985) *Getting Computer to Talk Like You and Me.* MIT Press.

Reithinger, N. (1991) Eine parallele Architektur zur inkrementeller Dialogbeiträge. Infix, Sankt Augustin, Germany.

Rips, L. J. (1983) Cognitive processes in propositional reasoning. *Psychological Review*, 90.

Sibun, P. (1990) The local organization of text. In *Proc. of the fifth international natural language generation workshop.*

Wick, M. R. and Thompson, W. B. (1992) Reconstructive expert system explanation. *Artificial Intelligence*, 54.