TR-505

General-purpose Reasoning Assistant System EUODHILOS

by
T. Minami, H. Sawamura, K. Yokota,
K. Ohashi & T. Ohtani (Fujitsu)

September, 1989

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

General-purpose Reasoning Assistant System EUODHILOS
汎用論証支援システム EUODHILOS

Toshiro Minami*, Hajime Sawamura*, Kaoru Yokota**, Kyoko Ohashi**, and Takeshi Ohtani*
南 俊朗, 沢村 一, 横田 かおる, 大橋 恭子, 大谷 武

* *International Institute for Advanced Study of Social Information Science(IIAS-SIS), Fujitsu Ltd., 140 Miyamoto, Numazu 410-03, Japan.* *Email:* minami%iias.fujitsu.co.jp@uunet.uu.net
** *Software Development Lab., Fujitsu Laboratories Ltd., 1015 Kamikodanaka Nakahara-ku, Kawasaki 211, Japan*

EUODHILOS(Every universe of discourse has its logical structure. [10]) は，汎用性を特徴とする論証のための対話型支援システムである．ユーザは，対象領域を表わすための確定節文法を元にした言語表現及びその論理的構造を規定する推論規則・書き換え規則・公理を最初に定義する．システムはこれらの定義に基づき，その論理系の下での定理の証明を支援するための思考シートと呼ばれる証明構築環境を提供する．思考シート上では前向き・後向きの両方向の導出を自由に混在することが可能である他，ある部分証明を２つの部分証明に分離することや，その反対に部分証明を結合して，より大きな証明にする等，柔軟かつ容易に証明を構成できる．

1. Introduction

Logical reasoning plays important roles in many fields like mathematics, computer science, artificial intelligence. Various logics such as first-order, higher-order, equational, temporal, modal, intuitionistic, and type theoretic logics are used there. We have been making research for a reasoning system assisting reasonings in such a variety of logics.

EUODHILOS is a general-purpose reasoning assistant system. Two major subjects are pursued to realize it; which become characteristic features of the system. The first one is the "generality" in reasoning. The phrase "general-purpose" indicates the independence from any specific logics. As S. K. Langer[10] told, we recognize that "Every universe of discourse has its logical structure." That is a thought that for each object we mention, there must be the logic best suited for expressing and discussing about it. In order to assist human reasoning for various objects, the reasoning assistant system must have the ability to describe a variety of logical structures and manipulate the expressions under these logics. The name EUODHILOS is an acronym of the phrase by Langer, and it is taken as the system name to emphasize the generality of the system.

The other subject is the investigation of "reasoning-oriented human-computer interface." The fundamental recognition in this subject is that the reasoning essentially proceeds through trial and error. A system is helpful for one to conceive ideas in reasoning if it has a good interface so that one can reason easily.

By using the general-purpose reasoning assistant system EUODHILOS, we can treat various kinds of knowledge represented as logical expressions in a uniform way, and we can investigate the relationships between them. We do not take up the completely automated reasoning style. Instead, we choose the semi-automated one, that is, (i) the user takes the initiative and uses the system as an intelligent tool, and (ii) the work achieved by the system is comparatively an easy one and it can be achieved automatically and will terminate in a reasonable time.

2. Configuration

EUODHILOS is a prototype of general-purpose reasoning assistant system. We intend to clarify the image of the ideal reasoning system by developing and using it. It is designed by considering the two issues: (i) Realization of the generality of the system, and (ii) Environment for experimenting logical model construction.

Figure 1 is an illustration of how the system is organized. In the upper half of the figure which corresponds to the feature (i) above, the user specifies components of a logic, such as symbols, syntax of expressions including formulas, derivation rules. In the lower half, which corresponds mainly to (ii), the user tries to construct proofs of theorems under the logic defined in the previous step. In EUODHILOS, partially constructed proofs, which are called proof fragments, appear scatteringly on the screen. User edits

This work is a part of the major research and development of the FGCS project conducted under the program set up by MITI.

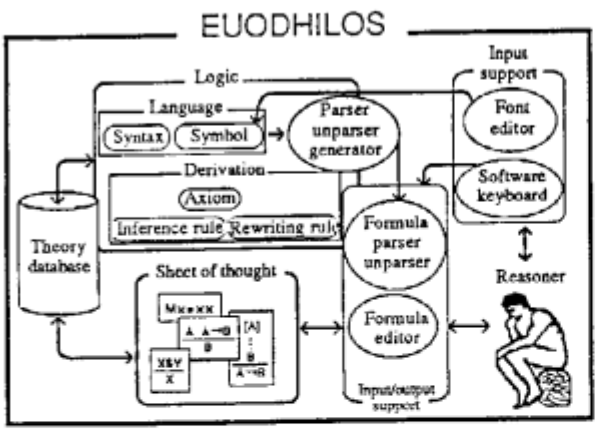

Figure 1: Configuration of EUODHILOS

these proof fragments by the editing functions such as create, delete, derive, connect, separate, and so forth. The sheet of thought is an environment for creating theorems and their proofs. The theorems constructed on the sheet can be saved to the library of theorems so that they can be reused as a starting formula in the later proofs for other theorems. In this way, users can cumulatively construct many theorems in the theory.

2.1 Language Description

In EUODHILOS, the language system to be used is designed and defined by the user at the beginning. The language system consists of the definitions of the syntax for the logical expressions. The syntax of the expressions is given by the definite clause grammar (DCG)[13]-based description method.

From the description, with some auxiliary data, a bottom-up parser based on BUP[11] and the unparser for the defined language are automatically generated. The parser translates from the external expression– the string of characters– to the internal expressions, while the unparser translates in the other direction. The parser and unparser are used in all the succeeding phases of symbol manipulations. When an expression is entered, the parser is invoked to check its validation. At the same time the internal structure of the expression in the language are constructed as well. Owing to this function, one can omit the specifications on the internal structures of expressions in the syntax definitions in EUODHILOS. When derivation commands are given by the user, the internal expressions of the formulas are manipulated and new internal expressions are generated. These expressions are presented to the user after being translated into the external ones by the unparser.

Figure 2 is an example description of the language for a puzzle of mocking bird by R. Smullyan[14] which is an interpretation for combinatory logic. From

```
formula→term, "=", term
term→b_term
term→b_term, "•", b_term
b_term→variable_symbol
b_term→constant_symbol
b_term→"(", b_term, "*", b_term, ")"
b_term→"(", term, ")"
variable_symbol→ "A"-"E" | "s"-"z"
constant_symbol→ "I"-"N"
meta_formula→ "F"-"H"
meta_term→ "Y"-"Z"
meta_b_term→ "X"
```

Figure 2: A description of the language for a puzzle

the definition, we can see that expressions such as "M•x=x•x" and "(A*B)•x=A•(B•x)" are formulas of this logic. Meta symbols are used in the definitions of axioms, inference rules, and rewriting rules and also in a schematic proof on a sheet of thought.

2.2 Axiom and Derivation Rule Description

A derivation system in EUODHILOS consists of axioms and derivation rules. Inference and rewriting rules are used as derivation rules. A list of formulas is given as the axioms. Inference rules are given in a natural deduction style. That is, an inference rule consists of three parts; the first one is the premises of the rule, each of which may have an assumption, the second is the conclusion of the rule, and finally the third is the optional restriction that is imposed on the derivations of the premises, such as variable occurrence conditions (eigenvariable). Well-known typical styles of logics such as Hilbert's, Gentzen's, equational can be treated within this framework.

Schematically, an inference rule is given in the following form:

$$\frac{\begin{array}{cccc}[\text{Assumption}_1] & [\text{Assumption}_2] & \cdots & [\text{Assumption}_n] \\ \vdots & \vdots & & \vdots \\ \text{Premise}_1 & \text{Premise}_2 & \cdots & \text{Premise}_n\end{array}}{\text{Conclusion}}$$

In this form, each of the assumption parts is optional. If a premise has its assumption, it indicates that the premise is obtained under the assumption, and otherwise that it is obtained by any means. An inference rule may have a condition on its application. An inference rule can be applied if all the premises are obtained in this manner, and the restrictive condition, if any, is satisfied. If it is applied, the conclusion is obtained as the result of the derivation.

Figure 3 is the definitions of axioms and inference rules for the logic of mocking bird. In the definition of inference rule named substitution, the expressions

'[X]' and '[Y]' indicate the occurrences of the expressions 'X' and 'Y' in a formula 'F' respectively.

Axioms:
`M•x=x•x` *Existence of the mocking bird.*
`(A*B)•x=A•(B•x)` *Composition.*
Inference rules:

$$\frac{F[X]}{F[Y]}\ \text{(substitution)} \qquad \frac{F[Y] \quad Y=Z}{F[Z]}\ \text{(equality)}$$

Figure 3: Axioms and inference rules for the logic of mocking bird

Considering the fact that mathematicians use rewriting rules so much as inference rules, we decided to add rewriting rules as a kind of derivation rules. Rewriting rules are presented in the following form:

$$\frac{\text{Pre\_Expression}}{\text{Post\_Expression}}$$

A rewriting rule is applied to an expression when it has a subexpression which matches to the pre_expression part of the rule. The resultant expression is obtained by replacing the subexpression with the appropriate expression corresponding to the post_expression part of the rule.

One can obtain a derivation tree by iterating the applications of the derivation rules. To reduce the similar applications of several rules appearing in many places, one can define the derivation rules. Once defined, a derivation rule can be used just like a primitive one.

2.3 Constructing Proofs

In EUODHILOS an environment called the "sheet of thought" provides the assistance for finding theorems and their proofs. It allows one to draft a proof, to connect proof fragments (i.e. partially constructed proofs), separate a proof, to reason by using lemmas, and so on. It is designed from our thought that theorems and proofs are found under the user's initiative through the process which would proceed with trial and error. On a sheet of thought, proof fragments are the elementary units for manipulation. Proof fragments are newly created as assumptions, axioms, or theorems of the theory obtained beforehand. These fragments are composed, and deleted according to the operations given by the user. Inference and rewriting rules are applied and expressed schematically just like those printed on the paper. This naturally induces that the appearance of a derivation tree on the sheet is also displayed in a tree form. This way of treating is a kind of proof visualizations.

It is desirable that reasoning during proof construction can be done along the natural way of thinking of human reasoners. Therefore EUODHILOS supports the typical method for reasoning, that is, forward (or top-down) reasoning, backward (or bottom-up) reasoning, interpolation (i.e. filling the gap between proof fragments) and reasoning in a mixture of them. They are accomplished interactively by manipulating the fragments on a sheet of thought. It is planned to incorporate not only such a proving methodology but also the methodologies of science.

As an example of derivation process on a sheet, we will illustrate how one can proceed derivations in the example of the mocking bird. The problem is to prove the statement: 'Any bird is fond of some bird.' That is, for any bird 'A' there exists a bird 'X' such that "`A•X=X`" holds. Figure 4 is the actual screen image of sheet of thoughts for this example. At first, in a sheet of thought at the top-right corner of the figure, one enters two axioms "`M•x=x•x`" and "`(A*B)•x=A•(B•x)`" on a sheet. To deduce some formula, he may deduce "`M•A=A•A`" from the axiom "`M•x=x•x`" by substituting 'A' to the variable 'x.' He cannot proceed any more in this case, so he tries other substitution. Next, at the middle-left sheet, he may substitute 'A*B' to 'x'. In this case, he gets "`M•(A*B)=(A*B)•(A*B)`" and "`(A*B)•(A*B)=A•(B•(A*B))`." After looking these, he makes aware that by substituting 'M' to 'B' he gets the desired formula "`(A*M)•(A*M)=A•((A*M)•(A*M))`." This indicates that a bird 'A' is fond of the bird denoted by the expression "`(A*M)•(A*M)`." If he re-reads his proof carefully, he may become aware that the proof is redundant, and he can get the final proof of the theorem; By substituting 'A*M' to 'x' and 'M' to 'B', and by the inference rule of equality, one can get the desired formula. (It is shown on the bottom-left sheet.) Proofs on the sheet of thought proceed like this. To see the more practical examples, refer to [12] and other papers in it.

2.4 Comparisons to Related Systems

In this section, we will compare the reasoning assistant system (RAS for short) with the following three types of the system which can be used for assisting human reasoning: (1) ATP(automated theorem prover), (2) proof checker, and (3) proof constructor.

Two of the most significant and characteristic features of RAS are (i) that it is independent of specific logics, and (ii) that it supports the natural and easy-to-use interactive proof constructions.

An ATP is a system which searches a proof of a formula given by the user. In a RAS, on the other hand, proofs are searched and found through the interactions between the system and the user. The initiative is taken by the user. This is the major difference between a theorem prover and a RAS.

A proof checker is a system which verifies the correctness of a proof described by the user. In a proof checker, the user has a putative proof of a theo-

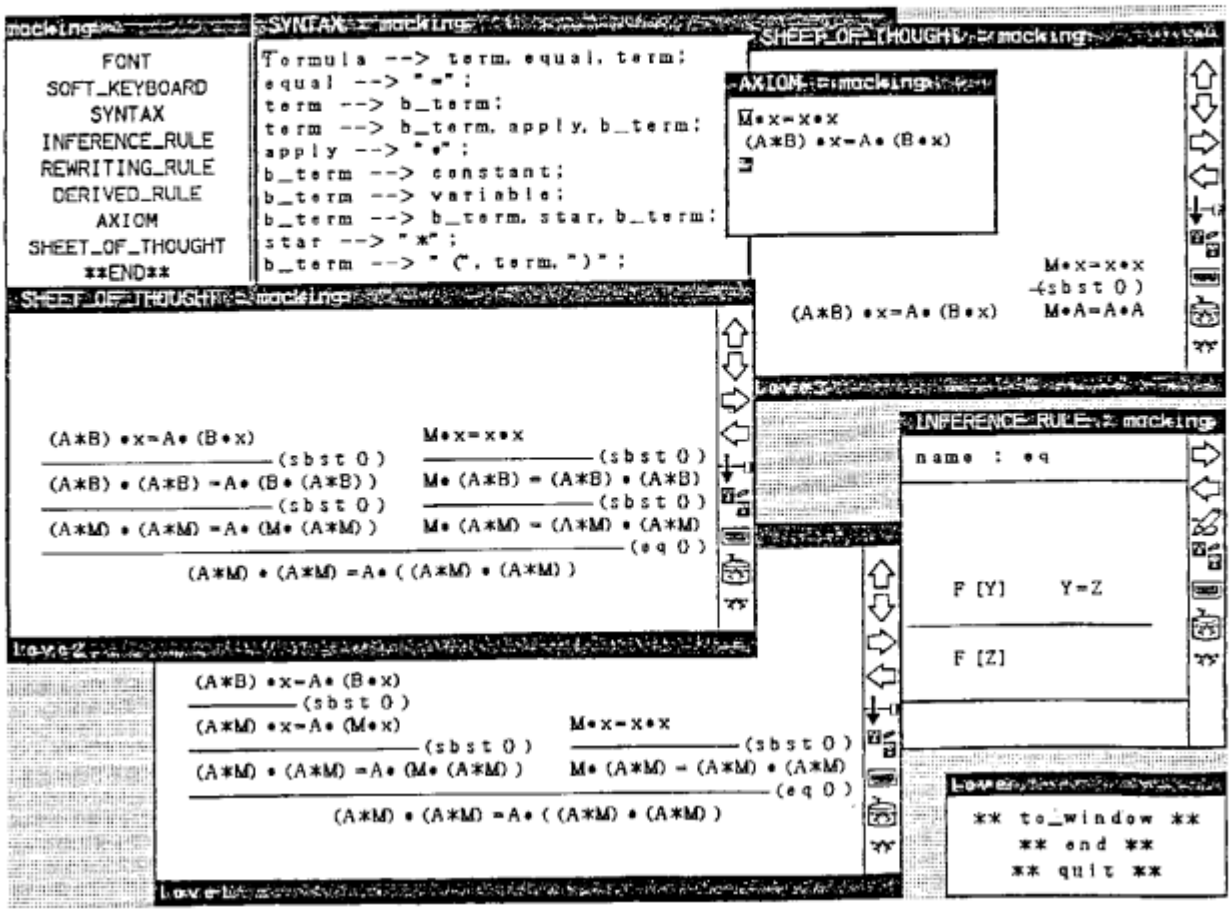

Figure 4: Proof Construction on the Sheet of Thought

rem at the beginning. A human proof may contain some careless mistakes including small gaps in a proof. The checker provides a language for describing human proofs. The user describes his or her proof by this language and gives it to the system. The system checks the proof. If the checker finds errors in the proof, it shows them to the user. When a user has a proof and wants to verify its correctness, a proof checker is one of the best tools for him. But when one begins to find a proof for some formula, the system such as RAS which assists to construct a proof is better than the proof checker.

Many proof checkers have been developed up to now. AUTOMATH [1] is one in which the user specifies the construction of proofs. PL/CV2 [2] is used for proving the correctness of PL/I like programs. CAPLA [8] deals with the proofs on linear algebra.

A proof constructor(e.g. LCF [5], FOL [15], EKL [9], and Nuprl [3]) is a system which supports a user to construct proofs as well as theorems through the interaction between the user and the system.

EUODHILOS may be seen as a proof constructor. The important thing is that it is general-purpose, which deduces some significant difference to other constructors in those underlying logics are fixed. Since EUODHILOS is general, it can be applied to any cases of human reasoning, the fixation of logic may restrict the reasonings under consideration. The representation of proof fragments is another significant difference from other proof constructors. We use the natural tree format of representation, so we can reason comparatively easier, and this difference, though it seems a liitle one, induces much difference in constructing difficult and complex proofs.

3. Concluding Remarks

The first version of EUODHILOS is now available and the next version is being improved by reflecting the experience of using the current one.

So far, we have dealt with logics, such as first-order logic (NK), propositional modal logic (T), intensional logic (IL), combinatory logic, Martin-Löf's type theory, and category theory. Many logics can be treated in the current version. Some logic such as tableau

method seems impossible to be treated in the current framework. We plan to extend the framework so that logics given in other formulations can be treated in the system.

From the experiments so far in EUODHILOS, we are convinced of the followings: (i) Describing the syntax of logical expressions is difficult at first. But, after defining several logics, we can define a new logic in a few hours. If the system keeps descriptions for typical logics as a library, the description of a new logic would be an easy task even for beginners. (ii) On a sheet of thought, users are free from deduction errors. On the paper, they may make mistakes in deriving a new formula when deduction rules are applied. The difference is important, because the users have to pay attentions only to the decisions how to proceed the proof on the sheet of thought. (iii) The reasoning assistant system can be used as a tool for CAI. In the system, users can deal with a variety of logics.

The current state is the first step towards the realization of a practical reasoning assistant system. To put the step forward, we have to investigate various subjects including the followings: (1) Treatment of relationships between meta and object theories, (2) Maintaining dependency relations among various theories, (3) Opening up various new application fields of reasoning, and (4) Improvement and refinement of human-computer interface for the reasoning system.

We believe the reasoning assistant system EUODHILOS will open up a new dimension of mechanized reasoning for practical use.

References

[1] N.G. de Bruijn: The Mathematical Language AUTOMATH, its Usage, and some of its Extensions, In M. Laudet et al. (eds.), *Sympoosium on Automated Demonstration*, Springer-Verlag, pp.29–61, 1970.

[2] R.L. Constable, S.D. Johnson & C.D. Eichenlaub: An Introduction to the PL/CV2 Programming Logics, *LNCS* **135**, Springer-Verlag, 1982.

[3] R.L. Constable et al.: Implementing Mathematics with the Nuprl Proof Development System, *Prentice-Hall*, 1986.

[4] J.A. Goguen & R.M. Burstall: Introducing Institutions, *LNCS* **164**, Springer-Verlag, 1983.

[5] M.J. Gordon et al.: Edinburgh LCF,*LNCS* **78**, Springer-Verlag, pp.221–270, 1979.

[6] T.G. Griffin: An Environment for Formal Systems, *ECS-LFCS-87-34*, Univ. of Edinburgh, 1987.

[7] R. Harper, F. Honsell & G. Plotkin: A Framework for Defining Logics, *ECS-LFCS-87-23*, Univ. of Edinburgh, 1987.

[8] ICOT: The CAP Project (1)–(6), *Proc. 32nd Annual Conv. IPS Japan*, 1986. (in Japanese)

[9] J. Ketonen & J.S. Weening: EKL—An Interactive Proof Checker, User's Reference Manual, *Dept. of Computer Science*, Stanford Univ., 1984.

[10] S.K. Langer: A Set of Postulates for the Logical Structure of Music, *Monist* **39**, pp.561–570, 1925.

[11] Y. Matsumoto et al.: BUP:A Bottom-Up Parser Embedded in Prolog, *New Generation Computing* **1**, pp.145–158, 1983.

[12] T. Minami et al.: EUODHILOS: A General-Purpose Reasoning Assistant System – Concept and Implementation –, IIAS-SIS Research Report No. 84, *FUJITSU LIMITED*, 1988.

[13] F.C.N. Pereira et al.: Definite Clause Grammars for Language Analysis—A Survey of the Formalism and a Comparison with Augmented Transition Networks, *AI Journal* **13**, pp.231–278, 1980.

[14] R. Smullyan: To Mock a Mockingbird, *Alfred A. Knopf Inc.*, 1985.

[15] R.W. Weyhrauch: Prolegomena to a Theory of Mechanized Formal Reasoning, *AI Journal* **13**, pp.133-179, 1980.