Fachbereich Informatik
Universität Kaiserslautern
Postfach 3049
D-6750 Kaiserslautern

SEKI - REPORT

An Extensible Natural Calculus for Argument Presentation

Xiaorong Huang

SEKI Report SR-91-3 (SFB)

An Extensible Natural Calculus for Argument Presentation

Xiaorong Huang

Fachbereich Informatik, Universität Kaiserslautern

W-6750 Kaiserslautern, Germany

E-mail: huang@informatik.uni-kl.de

Abstract

This paper proposes the notion of an extensible natural calculus, within a general setting of a computational model for human deductive reasoning. This research is motivated by a problem almost any contemporary automated reasoning system has to confront, namely, that these systems generate proofs of increasing complexity and length, that are almost incomprehensible for a human user. Hence the necessity to transform these machine generated proofs from their internal format into a more abstract and more human oriented style.

Researchers have long been aware of the need for an intermediate representation, to spilt the entire transformation process from the machine oriented representation via this intermediate representation finally into natural language. By analyzing the blueprint of the entire transformation process we first point out that this intermediate representation should have the quality of being the direct product of a human deductive apparatus or, at least an appropriate mathematical abstraction thereof. Then, based on our preliminary empirical study, we point out the inadequateness of using proofs based on Gentzen's natural deduction formalism as such an intermediate representation and set out our own proposal, called *detailed natural proofs* (DNPs). Here the notion of a proof step is generalized, by associating it to the notion of primitive cognitive procedures, rather than restricting it solely to inference rules of *natural logic* or *mental logic*. The main part of this paper is concerned with a formalism in which the DNPs can be encoded, called an *extensible natural calculus*. This discussion is carried out within the general setting of a computational model for human formal deductive reasoning.

This work was supported by the Deutsche Forschungsgemeinschaft, SFB 314 (D2, D3).

Table of Contents

Chapter 1. Introduction.

This paper proposes a computational model for human formal deductive reasoning. Within this general setting, emphasis is laid on the notion of an extensible natural calculus, serving as a useful intermediate representation in the transformation of machine generated proofsinto a more natural argument presentation. In this introduction, we first give a brief account of traditional theories on natural logics and mental reasoning. Then, with the help of a blueprint of the entire transformation process, we point out the sort of intermediate representation we need. Since this intermediate representation should have to possess the quality of being the direct products of a human deductive apparatus, we finally try to give some rough ideas as to how the traditional theories of mental reasoning have to be extended, in order to help to pin down this intermediate representation.

The word "natural" was first used by Gerhard Gentzen for his logic motivated by the desire "to set up a formal system that came as close as possible to actual reasoning" [Gentzen 1964]. More recently, *natural logics* (also called *mental logics*) are often studied within the framework of cognitive models of deductive reasoning by cognitive psychologists [Braine 78] [Lakoff 70] [Rips 81] [Johnson-Laird 83], where they serve as a candidate for an internal structure that is responsible for the human reasoning competence. From this point of view, a model of deductive reasoning is believed to contain mainly two components: 1) A logical component containing a repertoire of deductive vocabularies available to a human being, i.e. a natural logic in form of a set of inference schemata, where the patterns of the premises and of the conclusion are specified by formula schemata. 2) A performance component that contains mainly heuristics and programs responsible for putting inference rules together to form arguments or proofs.

The original incentive for our study on human deductive reasoning lies however in another scientific endeavor, namely the transformation of machine generated proofs into readable and adequate proofs in natural language (NL), via appropriate intermediate representations. As a whole, such a system carries out two different albeit closely related cognitive skills of human beings: the ability of performing formal deductions and the ability of presenting a proof in an effective and cooperative way. To simulate such a two-stage cognitive process on a computer, two computational theories are needed to simulate each of them. Also, there must be at least one intermediate representation, as illustrated in Fig. 1.1. The transformation between these different representations can only be devised if the concept of human oriented proofs is more precisely pinned down. The second cognitive skill, on the other hand, is simulated mainly by a computational model for human proof presentation [Huang 90]. On the observation, that mainly decisions for the inclusion or omission of each reasoning step in a proof are made in the presentation process, before other more natural language specific operations take effect, the intermediate representation we are looking for must be the pure logic proofs represented as such before any operations of arrangement have ever taken effect. We may call proofs serving as our intermediate representation *detailed natural proofs* (DNPs). Under this definition, detailed

natural proofs are nothing more than proofs where all reasoning steps performed by our deductive apparatus are still present.

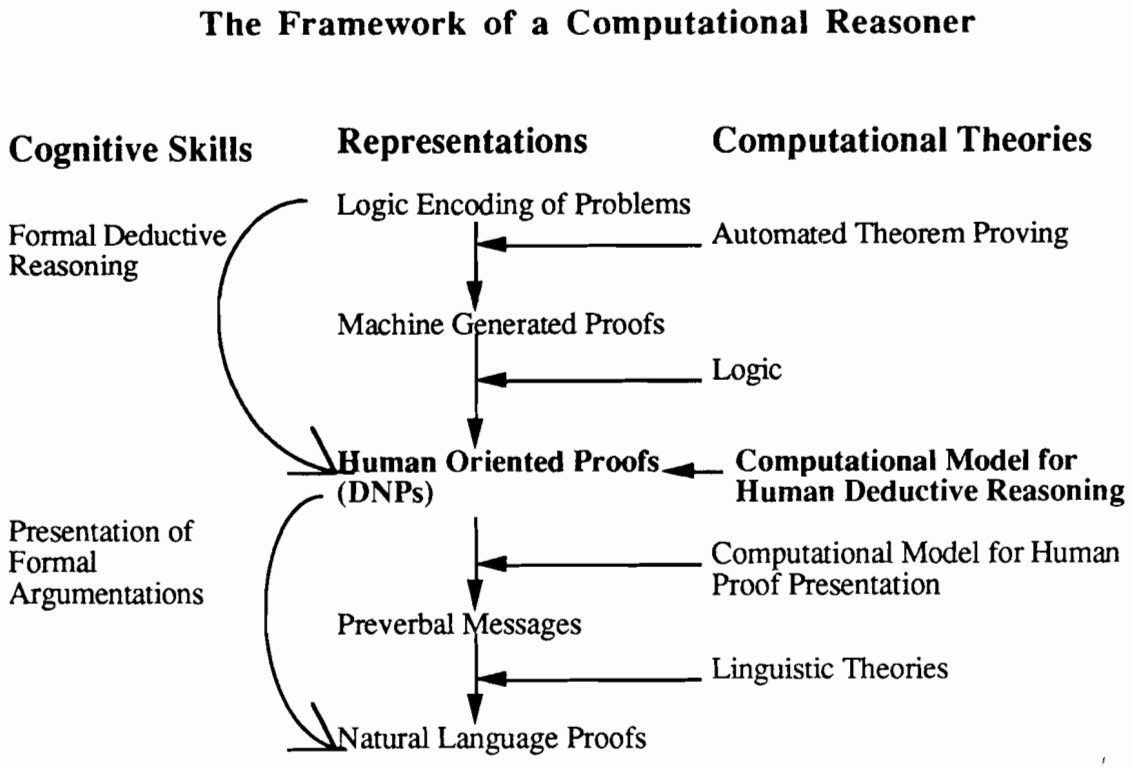

Fig. 1.1

What, however, are the direct products of a human deductive apparatus, and in particular, what are the primitive steps of such a proof? This, to the best of our knowledge, has not been investigated in any real depth. Though no explicit claims as to these direct products have been made in traditional models of human deductive reasoning, it has be taken for granted by many AI scientists, that it can be represented by a sequence of reasoning steps each justified by an inference rule of a natural (or mental) logic, with Gentzen's natural deduction system as a typical example.

Attempts have been made to transform proofs in machine oriented formalisms to natural deduction proofs [Andrews 80] [Miller 83] [Pfenning 90] [Lingenfelder 90]. Also some effort has been spent on constructing natural language generators, taking as input a natural deduction proof [Chester 76] [McDonald 83]. The experiences with these systems seem to indicate, that this is not an appropriate basis for a human oriented presentation. While proofs in various machine oriented formalisms have rather smoothly been transformed into natural deduction proofs, the latter, as a matter of fact, turn out to be an inappropriate basis for formulating pragmatic constraints concerning the presentation of arguments. The length of the natural deduction proofs alone, even of the best proofs for some simple problems encoded in this formalism, indicates that surely they can not be natural products of the human deductive apparatus, even not possibly mathematical approximations thereof [Huang 89]. To summarize,

we need a computational model of human deductive reasoning, or at least a framework of it, that tells us what DNPs are.

In order to gain more reliable experience with DNPs, in particular to clarify what kind of *reasoning steps* humans being perform, we have conducted some preliminary empirical studies: we took a mathematical textbook for undergraduates [Deussen 71] and encoded the theorems of the first chapter into predicate logic. Then we tried to make DNPs out of the proofs given in the book, by filling back all reasoning steps we believe to have been omitted. At the same time, we tried to make explicit the justification for each reasoning step. The same procedure was later carried out on some non-mathematical examples as well. To our surprise, only a small amount of reasoning steps are found to be justified by inference rules of Gentzen's natural deduction. Most, are either justified by the application of some new rules associated to Gentzen's rules in a particular way (they together are called inference rules at the logic level), or are believed to be justified by applying an axiom, a definition, or a theorem. On the account that mathematical axioms, definitions and theorems are nothing more than assertions in a logical system, these justifications will henceforth be called uniformly the *application of an assertion.* Later, it turned out that assertions in non-mathematical reasoning are often applied in a similar way. Hence, to summarize, a DNP can be conceived as a sequence of reasoning steps each justified by either the application of an inference rule or by the application of an assertion. However the latter, is still to be accounted for appropriately in a computational model.

Although we will show later, that an extensible natural calculus will suffice to support a mathematical abstraction of DNPs, the DNPs themselves and especially the notion of an extensible calculus goes beyond the scope of the traditional models of deductive reasoning. The central problem is how to account for the notion of a reasoning step, which is the elementary unit of a DNP, within a model of human reasoning. We first point out that although we agree with the assumption that inference rules of a natural logic are responsible for human deductive competence, it does not follow that there is a one-to-one relation between inference rules and reasoning steps. Instead, we suggest a one-to-one relation between reasoning steps and the notion of a primitive cognitive procedure, which we shall define later on.

Now let us examine how this change of perspective provides an account for our observations on DNPs. What do we mean when we say drawing a conclusion by the application of an inference rule? From the viewpoint of a computational model, we mean that a primitive cognitive procedure is called that draws a conclusion by matching premises against a specific inference rule. From such a procedural point of view, we find it plausible to suggest a more general structure for the so called logical component: there is some knowledge such as the inference rules, and there are some primitive procedures drawing conclusions based on this knowledge, such as the matching procedure. From this point of view, we only need to add a primitive procedure drawing conclusions based on assertions. As the matching procedure, this procedure differs from other procedures of the performance component in that it runs according to a simple and fixed algorithm, that does not need any heuristic knowledge, and does not have a problem space to search in.

Our definition of such a procedure is based on a crucial observation in our empirical studies: although these reasoning steps are always considered as atomic steps, people give explanations for them, when forced or motivated, and these explanations amount to a proof segment, where reasoning steps are justified solely by elementary logic level inference rules. Let us illustrate this by the following example:

Example 1.

Given the definition of subset as the following assertion

$\forall F,U\ U\subseteq F \Leftrightarrow \forall x\ x\in U \Rightarrow x\in F$

a human being will infer $a_1\in F_1$ from $U_1\subseteq F_1$ and $a_1\in U_1$ in one step, by applying this assertion. However, when he is forced to explain why this reasoning step follows from the definition above, (for instance, by a student with no training in logic and mathematics), his explanation will usually be something based on the following natural deduction proof, in a linearized version of Gentzen's natural deduction, first proposed by Andrews [Andrews 80]. Most important are the derived formulae in the middle and the inference rules used in the last column, with preconditions in parenthesis:

1.	1	$\vdash$	$U_1\subseteq F_1$	HYP
2.	2	$\vdash$	$a_1\in U_1$	HYP
3.	3	$\vdash$	$\forall F,U\ U\subseteq F \Leftrightarrow \forall x\ x\in U \Rightarrow x\in F$	HYP
4.	3	$\vdash$	$U_1\subseteq F_1 \Leftrightarrow \forall x\ x\in U_1 \Rightarrow x\in F_1$	$\forall$D(3)
5.	3	$\vdash$	$U_1\subseteq F_1 \Rightarrow \forall x\ x\in U_1 \Rightarrow x\in F_1$	$\Leftrightarrow$D(4)
6.	1, 3	$\vdash$	$\forall x\ x\in U_1 \Rightarrow x\in F_1$	$\Rightarrow$D(1, 5)
7.	1, 3	$\vdash$	$a_1\in U_1 \Rightarrow a_1\in F_1$	$\forall$D(6)
8.	1, 2, 3	$\vdash$	$a_1\in F_1$	$\Rightarrow$D(2,7)

Of course, it will be given in a much more informal way, and with some steps omitted. Based on this observation, we have come up with a definition suggests that this procedure draws conclusions by constructing logic level proofs, which are however, structurally strongly restricted. These logic level proofs will henceforth be referred to as the *natural expansions* (NE) of the assertion level reasoning steps. That the reasoning performed by this procedure can intuitively be seen as one step in a formal proof, rather than a proof segment, lies on its special cognitive characteristics. Namely that it is conceived as being very simple by a mathematician with some conpetence.

Assertion level inference steps may be justified by inference rules as well. These are however usually acquired during reasoning task. This leads to the notion of an extensible calculus, that corresponds to a learning mechanism of the computational model. This is the second divergence from traditional theories, which assume the existence of psychologically elementary rules that are fixed once and for all. Let us illustrate the way such a rule may be formed and entered into our rule set by continuing with the example above. It is not difficult to

imagine, after drawing the conclusion $a_1 \in F_1$ by constructing the proof segment above, an inference rule as follows that is acquired, as a side-product:

$$\frac{U_1 \subseteq F_1,\ a_1 \in U_1}{a_1 \in F_1}$$ where U_1, F_1 and a_1 are object constant

This, in turn, may be generalized into another more general inference rule by replacing constants U_1, F_1 and a_1 by meta-variables U, F and a. This, now, may be memorized and added to the natural calculus:

$$\frac{U \subseteq F,\ a \in U}{a \in F}$$ where U, F and a are meta-variables for object constant

The generalization involves only variables that do not originally occur in the assertion. According to our model, it is now important to point out that assertion level reasoning steps can be carried out in two distinct ways: either by a procedure constructing a logical level proof segment, or more efficiently, by matching against an assertion level inference rule, acquired in a previous context.

With the actual form of DNP established, we may now design a mathematical abstraction of it, which can be encoded in a extensible natural calculus and which can serve the goal of proof transformation as well. For each proof step in DNP, following the tradition, we may also write down the knowledge used in the justification process, rather than the procedures. Within the framework described above, the knowledge concerned may in the first place be an inference rule at the logic level. In the second place, for assertion level steps, it may either be the assertion itself, or an inference rule derived from it, depending on context. On the ground that a rule is simply more specific, anyway, we have chosen to give a derived rule at the assertion level as the justification. To conclude, the mathematical abstraction of DNPs will be encoded in an extensible natural calculus including inference rules at the logical level as well as at the assertion level. We will talk about DNPs from now on, referring to DNPs themselves and there mathematical abstractions interchangeably.

In the rest of this paper, we proceed gradually to our goal: the notion of an extensible natural calculus. The whole discussion is carried out in the framework of a computational model of formal deductive reasoning. Although the psychological realness is not our main concern, much value is still put on the cognitive adequateness of the model set up. It should shed more light on the internal structure underlying the cognitive skill of formal deductive reasoning, besides serving as an environment to discuss the natural calculus. Of course, emphasis is put on the procedure applying assertions as well as on the mechanism accounting for the formation of new inference rules. The rest of the paper is therefore straightforward as follows: Chapter 2 provides an overall structure of the computational model. Chapter 3 to 6 are each dedicated to one element related to the building up of the natural calculus. Related works and possible applications of the natural calculus are discussed finally in Chapter 7.

Chapter 2. General Framework of a Computational Model

In this chapter, we first set up an overall framework of a computational model for human formal deductive reasoning. The structure we are going to propose is very similar to an abstract data type. There are in the first place memory spaces holding data, and in the second place, operators manipulating the data. Finally, there are effective procedures carrying out the cognitive skills concerning formal deductive reasoning. They carry out their tasks either by applying some operators or by calling some other procedures. We shall now provide a short description of each of them:

2.1. Memory Spaces:

As conventionally, we assume the existence of a *long term memory* (LTM) and a short term or *working memory* (WM), where the latter can be intuitively viewed as the focused part of the entire memory space. It contains only objects a human being is attending to at present, and is supposed to be limited in space.

2.2. Data in Memory Spaces

For the sake of simplification, we assume that the only content stored in both of the memory spaces are of the following three sorts:

a) inference rule schemata in the form of $\frac{\mathcal{A}_1 \vdash P_1, \ldots, \mathcal{A}_n \vdash P_n}{\mathcal{A} \vdash Q}$, where P1,...,Pn are formula schemata specifying premises, and Q is a formula schema specifying the conclusion. $\mathcal{A}_1, \ldots, \mathcal{A}_n$, and $\mathcal{A}'$ are the corresponding assumptions (see below). When no manipulations on assumptions are needed, we also use a simplified form $\frac{P_1, \ldots, P_n}{Q}$.

Postulate: There is a fixed set of primitive inference rule schemata, which reside permanently in WM. They possess the cognitive status as of being *elementary* and *innate*, as apposed to *compound* and *acquired*. In terms of the computation model, they are not generated by any operators or procedures.

b) Assertions encoded in first-order predicate logic.

c) Detailed natural proofs (DNPs) and the unfinished fragments thereof. Since a proof or an argument is generally believed to be a chain of reasoning steps, we have adopted the linearized version of natural deduction proofs used in [Andrews 80], slightly modified in order to accommodate assertion level inference steps. Formally, every natural deduction proof is a finite sequence of proof lines each of the form:

line-No $\mathcal{A} \vdash \mathcal{F}\ \mathcal{R}$(reason-pointers)

where line-No is a natural number, $\mathcal{A}$ is a finite, possibly empty set of formulas, which are called assumptions. And $\mathcal{F}$ is the derived formula. The reason pointers, finally, are a set of previous proof lines, which are taken by the primitive procedure $\mathcal{R}$ justifying this proof line as preconditions. The reason pointers and the preconditions are usually replaced by their line numbers. Every proof line in a valid proof is justified in one of the following ways:

a) by a matching procedure that matches against a cognitively elementary inference rule,

b) by a matching procedure that matches against a cognitively compound inference rule generated by one of the rule generating operators,

c) by a primitive procedure applying an assertion in WM.

No commitment is made as to the form of unfinished segments of DNPs.

2.3. Operators and Primitive Procedures:

a) bookkeeping operators moving objects of various data type out of or into memory spaces, and bookkeeping procedures

b) a procedure drawing conclusions by matching premises against inference rule schemas,

c) a procedure drawing conclusions, by constructing logic level proofs with certain structural constraints,

d) an operator generating *associated* inference rules from existing rules residing in WM.

e) an operator generating new inference rules from assertions in WM, hand in hand with the assertion application procedure c), in a way illustrated in the introduction. The rules generated possess the cognitive status as being *compound* and *acquired.*

Different from autonomous procedures to be introduced below, the primitive procedures all follow a fixed algorithm without any heuristic search, and thus can be performed by reasoners without undue attention. They correspond to the effective procedures discussed by cognitive psychologists [see e.g. John-Laird 83].

4. Autonomous Procedures

Complex cognitive skills, in particular those which are observable by introspection, are carried out by the autonomous procedures in our computational model. Some examples are:

a) an interface procedure

b) complex reasoning procedures such as: a proof construction procedure and a proof checking procedure

Now where is the place for an extensible natural calculus within such a model? It should apparently be the sum total of inference rules which may once be generated and thus be at the disposal of the complex autonomous reasoning procedures. In our computational model, an assumption is made that the complex reasoning procedures are both able to move each assertion or inference rule in LTM into WM, and able to generate every possible inference rule by applying the two rule generating operators. Although a natural calculus is by nature a dynamic entity,it follows apparently that given a collection of assertions known (in LTM or WM), the corresponding natural calculus is uniquely determined by the following three elements: a fixed set of elementary rules and the two rule generating operators.

Before proceeding to the chapters that elaborate each of these three elements, it is worth examining the inference rules from the perspective of their cognitive status: On the ground that all assertion level rules are acquired by running a procedure constructing a proof segments justified by elementary rules, they can logically be thought of as macros which are compound in nature. Thus it is the set of cognitively elementary or innate inference rules that accounts for the ultimate *logical competence* of a human being. To account for the *effective competence*, nevertheless, it has to be enriched by those macros, which can be momenterily at the disposal of a reasoner as well. Whether an assertion level reasoning step is drawn by simply matching or by a procedure constructing a low level proof segment, depends on whether the corresponding assertion level rule is acquired in a previous context.

Chapter 3. Cognitively Elementary Inference Rules.

As the foundation of the entire discussion, we first introduce in this chapter the set of cognitively elementary inference rules. It was no doubt a significant achievement of Gentzen to have identified a group of primitive inference rules in 1930, which comprise his natural deduction calculus [Gentzen 30]. His work is supported by our empirical studies once again showing that the main part of the cognitively elementary rules with respect to formal deductive reasoning are really already suggested by him. The following is a listing of the inference rules presented in his calculus NK:

Structural Gentzen Rules:

$$\frac{}{\mathcal{A},F \vdash F} \qquad \frac{\mathcal{A},F \vdash G}{\mathcal{A} \vdash F \Rightarrow G} \qquad \frac{\mathcal{A} \vdash F \vee G,\ \mathcal{A},F \vdash H,\ \mathcal{A},G \vdash H}{\mathcal{A} \vdash H} \qquad \frac{\mathcal{A} \vdash \exists x\, F_x,\ \mathcal{A},F_a \vdash H}{\mathcal{A} \vdash H}$$

HYPothesis **DED**uction **CASE** **CHOICE**

$$\frac{\mathcal{A},G \vdash \bot}{\mathcal{A} \vdash \neg\neg G} \qquad \frac{\mathcal{A},\neg G \vdash \bot}{\mathcal{A} \vdash G}$$

IP(Indirect Proof)

Non-Structural Gentzen Rules

$$\frac{\mathcal{A} \vdash F,\ \mathcal{A} \vdash G}{\mathcal{A} \vdash F \wedge G} \qquad \frac{\mathcal{A} \vdash F}{\mathcal{A} \vdash F \vee G} \qquad \frac{\mathcal{A} \vdash G}{\mathcal{A} \vdash F \vee G} \qquad \frac{\mathcal{A} \vdash F_a}{\mathcal{A} \vdash \forall_x F_x} \qquad \frac{\mathcal{A} \vdash F_a}{\mathcal{A} \vdash \exists_x F_x}$$

$\wedge$**I** $\vee$**I** $\forall$**I** $\exists$**I**

$$\frac{\mathcal{A} \vdash F \wedge G}{\mathcal{A} \vdash F} \qquad \frac{\mathcal{A} \vdash F \wedge G}{\mathcal{A} \vdash G} \qquad \frac{\mathcal{A} \vdash F,\ \mathcal{A} \vdash F \Rightarrow G}{\mathcal{A} \vdash G} \qquad \frac{\mathcal{A} \vdash \forall_x F_x}{\mathcal{A} \vdash F_a}$$

$\wedge$**D** $\Rightarrow$**D** $\forall$**D**

$$\frac{\mathcal{A} \vdash F,\ \mathcal{A} \vdash \neg F}{\mathcal{A} \vdash \bot} \qquad \frac{\mathcal{A} \vdash \bot}{\mathcal{A} \vdash D} \qquad \frac{\mathcal{A} \vdash \neg(\neg F)}{\mathcal{A} \vdash F}$$

$\neg$**D** $\bot$ $\neg\neg$

Every figure above shows an inference rule. Formula schemata separated by commas above the bar represent preconditions (also called reasons). Meta-variables F, G and H can be substituted by any formula, $\forall_x F_x$, $\exists x\, F_x$ by any formula with $\forall$ or $\exists$ as the top symbol, where x denotes the corresponding bound variable. F_a denotes the formula achieved by replacing all occurrences of the bound variable "x" in F_x by an individual constant "a". The

Meta-variables "a" in ∀D can be substituted by an arbitrary term. For rule ∀I and CHOICE, in addition, the following variable conditions must be checked respectively:

The variable Condition for ∀I: meta-variable "a" (Eigenvariable in [Gentzen 30]) may not occur in $\forall_x F_x$ or in any formula in the assumption set $\mathcal{A}$.

The variable Condition for Choice: meta-variable "a" may not occur in H, or in any formula in the assumption set $\mathcal{A}$.

Only one symmetric pair of additional cognitively elementary rules was identified during our empirical studies. They were given the name ∨D:

$$\frac{\mathcal{A} \vdash P \vee Q,\ \mathcal{A} \vdash \neg P}{\mathcal{A} \vdash Q}\ ,\ \frac{\mathcal{A} \vdash P \vee Q,\ \mathcal{A} \vdash \neg Q}{\mathcal{A} \vdash P} \qquad \vee D.$$

We also make an extension to the condition, under which a rule can be applied, to enhance the naturalness. In the traditional natural deduction systems, to apply a rule

$$\frac{P_1,\ldots,P_n}{Q},$$

for each precondition P_i, a previous proof line must be found, whose derived formula F is the corresponding instance of P_i. Based on our empirical study we have extended it in the following two ways:

i) for each premise P_i, a previous proof line with a derived formula F of the form ...∧P´i∧... must be found, where P´i is the corresponding instance of P_i.

ii) for each premise P_i of the form A⇒B, a previous proof line with a derived formula F of the form

$\mathcal{A}$,A´ ⊢ B´ $\mathcal{R}$

must be found, where $\mathcal{A}$ and $\mathcal{R}$ are the other assumptions involved and the inference rule justifying this proof line, respectively. A´ and B´ are the corresponding instances of A and B.

One thing that is worth being mentioned is that the naturalness of the natural deduction system, with respect to formal reasoning, is largely due to the inference rules under the category of *structural rules*, where the usual ways of mathematical reasoning are simulated. For example, assumptions are introduced or discharged, problems are divided into cases, etc. The non-structural rules, on the contrary, deal mostly with primitive manipulations of logical connectives and quantifiers and are rarely found in DNPs. They are used, nearly exclusively as building blocks for constructing natural expansions by the procedure applying assertions. However, we do not want to make any commitment as to which rules are exclusively used in DNPs by human beings and which are, on the contrary, only used in natural expansions.

Furthermore, it is certainly an oversimplification to claim that all the rules listed in this chapter are not only cognitively elementary but also innate. While the non-structural ones are similar to those included in natural calculus set up by cognitive psychologists [Braine 78] (see also chapter 7) and are therefore more likely to be innate, the structural ones require probably a

general training in formal reasoning, such as the training provided in mathematical courses. From the perspective of cognition, therefore, these are the rules assumed for everyone with a formal training. From the computational perspective, they are elementary in the sense that they are not generated by any operators or procedures.

Chapter 4. Deriving Associated Rules

In this chapter, we introduce the first of the two operators generating inference rules, responsible for accounting for all the inference rules identified in DNPs, apart from those elementary ones. We first give a formal definition and then some examples. The operator is specified by the following schema:

if R is a rule in WM and it is of the form

$$R = \frac{A\vdash p_1,\ldots,A\vdash p_n}{A\vdash q} \qquad 4.1$$

then R´ of the form below is a rule associated with R:

$$R' = \frac{A\vdash p_1,\ldots,A\vdash p_{i-1},A\vdash p_{i+1},A\vdash p_n,A\vdash\neg q}{A\vdash\neg p_i} \qquad 4.2$$

This implies, intuitively, that the rule r and its associated rules are memorized more or less as one unit of the form

$\{p_1,\ldots,p_n,\neg q\}\vdash\perp$,

which means that at least one element of the set $\{p_1,\ldots,p_n,\neg q\}$must be false. This derivation, in addition, is carried out in conjunction with the cancellation of negations. This means if p_i is of the form $\neg p_i'$, then the conclusion schema in 4.2. will be p_i', instead of $\neg\neg p_i'$. The same holds for formula schema $\neg q$ in 4.2.

Let us now examine some of the rules generated by this operator. The two pairs of rules in 4.3 and 4.4, as identified in our empirical studies can be derived from $\Rightarrow$D and $\wedge$D, respectively.

$\Rightarrow$D´: $\quad \frac{a,\neg b}{\neg(a\Rightarrow b)}, \quad \frac{a\Rightarrow b,\neg b}{\neg a}$ $\qquad 4.3$

$\wedge$D´: $\quad \frac{\mathcal{A}\vdash\neg F}{\mathcal{A}\vdash\neg(F\wedge G)} \quad \frac{\mathcal{A}\vdash\neg G}{\mathcal{A}\vdash\neg(F\wedge G)}$ $\qquad 4.4$

The elementary inference rules introduced in chapter 3, together with their associated rules, are called the rules at the *logic level*, as apposed to rules generated by the operator to be introduced in the next chapter, which we will refer to as being at the *assertion level.*

Although not really effecting the final logical power of the natural calculus, we believe it is worthwhile to point out the existence of various other ways to account for these rules, and compare them with the solution we choose from a cognitive perspective. First, to account for the logic level inference rules, for instance those in 2.3 and 2.4, we could simply include them

into the set of elementary rules. It is nevertheless quite counter-intuitive to assume a large set of interrelated inference rules as cognitive elementary. Another argument against their being elementary, we believe more mental effort is needed to carry out deductions justified by the associated rules. A second proposal seems more plausible, at least at first sight. We may loosen the application condition of inference rules so that an inference rule will already cover all the deductions now to be covered by rules associated with it, without assuming their explicit existence. This, however, brings two drawbacks: In the first place this implies that the rule applications must involve search, before matching, though in a small space. In the second place, no distinction can be made between rules which are more central, and rules associated with them that play a comparatively secondary role. Second, since less mental effort is needed by this operator, (as compared with the operator generating assertion level inference rules directly from assertions in WM), it again seems plausible to grant the reasoner this ability.

The idea of associated rules can be generalized quite straightforwardly by taking variable conditions into account. The following is thus an associated rule of $\forall D$:

$$\frac{\neg P_a}{\neg \forall_x P_x} \qquad \forall D', \text{ which is a associated with } \forall D.$$

Notice that the derivation operator may be applied on inference rules both at the logic level and at the assertion level. And the derived rules will remain at the same level of abstraction. For example, given the assertion level inference rule introduced in the example in chapter 1:

$$\frac{U \subseteq F,\ a \in U}{a \in F}$$ where U, F and a are meta-variables for object constant

there are two associated rules which can be generated:

$$\frac{U \subseteq F,\ a \notin F}{a \notin U} \qquad \frac{a \in U, a \notin F}{U \not\subseteq F}$$

Before we are going to prove two properties of this operator, some new notation first. If an inference rule r can be derived from another rule r', then we call r and r' are associated with each other. Now, let R be an arbitrary set of inference rules, and Assoc(R) stand for the set of rules associated with at least one rule in R, then:

$$Assoc(R1 \cup R2) = Assoc(R1) \cup Assoc(R2) \qquad 4.5$$

$$Assoc(Assoc(R)) \subseteq R \cup Assoc(R) \qquad 4.6$$

While 4.5 is quite self-evident, 4.6 can be proved easily by cancelling of negation. Suppose

$$r = \frac{p_1, \ldots, p_n}{q} \in Assoc(Assoc(R)),$$

then without loss of generality,

$$r' = \frac{q', p_2, \ldots, p_n}{p'_1} \in Assoc(R),$$

where q' equals q'', if q is in form of $\neg q''$; otherwise q' equals $\neg q$. The same holds for p'_1. Now there are two cases: either

$$r = \frac{p_1,\ldots,p_n}{q} \in R$$

which proves our goal, or, again without loss of generality,

$$r = \frac{q', p_1, p_3,\ldots,p_n}{p'_2} \in \text{Assoc}(R)$$

which as well proves our goal, since p'_2 is defined similarly as q'.

Chapter 5. A Primitive Procedure Applying Assertions

In this chapter, a primitive reasoning procedure central to our model will be introduced, i.e., a procedure drawing conclusions by applying assertions. Before going into details, it is worthwhile to pause briefly and make clear the actual situation and the simplifications we have made. Let's first consider the situations within which the procedure under concern may be called. The first case is usually during the active proof construction, as already indicated in the introduction. It may as well by called under a more passive circumstance. For instance, during the reading of a mathematical text book, if a reader can not follow an assertion level reasoning step by resorting to a corresponding inference rule acquired previously, it is normal that he comes back to the assertion itself and checks this step by reading the assertion again. This checking process is always accompanied by a reasoning process at the logic level. This logic level proof segment, as already indicated, is called a *natural expansion* (NE) of the corresponding assertion level step.

Although it is possible that the applications of assertions under the two different circumstances are actually carried out by two procedures different from each other in their run-time behavior, a simplification to abstract them into one procedure is still viable, since we can not at the moment provide any run-time details anyway. What we can provide in this chapter is simply an input-output relation of the procedure because of lack of more precise knowledge. This input-output relation is deduced in section 5.1 from properties identified in the natural expansions, which appear to characterize them. Only a tentative psychologically real run-time depiction is attempted in section 5.2.

5.1. A Characterization of Natural Expansions.

In this section, structural constraints of syntactic nature on natural expansions will be emploied to define the input-output relation of this primitive reasoning procedure. Even intuitively it is obvious, that not all logic level proof segments, in which an assertion is in some way involved, embody an *application* of this assertion. A trivial example will illustrate this: suppose A is an assertion, B a second arbitrary formula, the valid derivation of $A \vee B$ do not go in line with our intuition of applying A, no matter what formula A is.

Along with the reconstruction of DNPs for problems in [Deussen 71], we have also expanded the reasoning steps at the assertion level to proof segments at the logic level, as natural as possible, with the hope that they will be the natural expansions. We will now in two steps introduce the characterizations identified thus far in the following two sub-sections.

5.1.1. Composition and Decomposition Constraints.

Before discussing the characteristics in more formal terms, let us first return to the subset example again, used at the outset.

Example 1. (continued)

In the introduction, a proof segment at the logic level is given serving as the NE of a reasoning step at the assertion level, i.e., inferring $a_1 \in F_1$ from $U_1 \subseteq F_1$ and $a_1 \in U_1$ by *applying* the following assertion:

$\forall F,U\ U \subseteq F \Leftrightarrow \forall x\ x \in U \Rightarrow x \in F$

The NEs are from now on represented in another formalism originally used by Gentzen himself [Gentzen 35], called proof trees. This, compared with the linearized version we normally use to represent proofs, is more suitable for the study of the structure of proofs. For instance, structural characteristics can be more conveniently formulated. The NE of our example is illustrated in Fig. 5.1. Some informal explanations first: In this formalism, every bar represents a reasoning step. The formulas above it, disconnected by commas, and the formula under it are the premises and the conclusion of this step, respectively. It can therefore be concluded that the leaves and the root of such a proof tree are the preconditions and the conclusion of the entire proof.

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\forall F,U\ U \subseteq F \Leftrightarrow \forall x\ x \in U \Rightarrow x \in F}{U1 \subseteq F1 \Leftrightarrow \forall x\ x \in U1 \Rightarrow x \in F1}\ \text{step 1}}{U1 \subseteq F1 \Rightarrow \forall x\ x \in U1 \Rightarrow x \in F1}\ \text{step 2} \quad , \quad U1 \subseteq F}{\forall x\ x \in U1 \Rightarrow x \in F1}\ \text{step 3}}{a1 \in U1 \Rightarrow a1 \in F1}\ \text{step 4} \quad , \quad a1 \in U1}{a1 \in F1}\ \text{step 5}$$

Fig. 5.1

The following properties can be observed in the particular proof above:

1. At every step, there is exactly one premise depending on the assertion being applied (the subset definition in this example). This means that there is a certain *linear quality* in the proof, along the shadowed branch in Fig. 5.1.

2. The premise depending on the assertion being applied, in addition, plays a special role at each step, in that all other premises (if there is any, such as in step 3 and 5), and the conclusion of this step are all either a subformula of it, or an instance thereof. This is henceforth referred to as the *subformula property*.

So the first observation in our empirical studies are that the NE of the application of an assertion is basically a linear sequence of decompositions. This is, however, not the whole cake. In other examples, there is also a composition process produces the other premises (if there are any) required by the linear decomposition along the shadowed branch. For instance, to apply an assertion of the form $\forall x\ P_x \wedge Q_x \Rightarrow R_x$, a composition of $P_a \wedge Q_a$ from the two separate premises P_a and Q_a has to be performed before deducing R_a. Let us now define two new concepts, that are used in the formulation of the constraints.

Definition 5.1

An inference rule of the form $\frac{\mathcal{A}\vdash A, \mathcal{A}\vdash p_1, \ldots, \mathcal{A}\vdash p_n}{\mathcal{A}\vdash q}$ is a *decomposition rule* with respect to formula schema A, if all applications of it, written as $\frac{\mathcal{A}\vdash A', \mathcal{A}\vdash p'_1, \ldots, \mathcal{A}\vdash p'_n}{\mathcal{A}\vdash q'}$, satisfy the following condition: $p'_1, \ldots, p'_n$ and q' are all either

1) a proper subformula of A′, or

2) an instance of A′ or one of its proper subformula, or, in the third case

3) a negation of one of the first two cases.

Under this definition, $\wedge$D, $\Rightarrow$D, $\forall$D are the only elementary decomposition rules (see Chapter 3).

Definition 5.2

An inference rule of the form $\frac{\mathcal{A}\vdash p_1, \ldots, \mathcal{A}\vdash p_n}{\mathcal{A}\vdash q}$ is called a *composition rule* with respect to q if all applications of it, written as $\frac{\mathcal{A}\vdash p'_1, \ldots, \mathcal{A}\vdash p'_n}{\mathcal{A}\vdash q'}$, satisfy the following condition:

$p'_1, \ldots p'_n$ are all either

1) a proper subformula of q′, or

2) an instance of q′ or one of its proper subformula,

(where F_a is an instance of both $\forall_x F_x$ and $\exists x F_x$) or, in the third case

3) a negation of one of the first two cases.

Now we again introduce some new terminologies: in a segment of a natural deduction proof serving as a NE of the *application of an assertion* A, the conclusion formula of the last line (the root in the corresponding proof tree) is called the *conclusion* of the application and all premises introduced by the Hyp rule (the leaves of the corresponding proof tree), except for the assertion A itself, are called the *premises* of the application. Now we can state the complete constraints observed during our preliminary empirical studies in more formal terms.

Composition and Decomposition Constraints:

A logic level proof segment serves as an NE of the application of an assertion A, if its proof tree satisfies the following constraints:

1. (*quasi-linearity property*) At every proof step, there is at most one precondition depending on A.

It can easily be concluded that all proof lines depending on A together form a branch in the proof tree, from the assertion A to the conclusion. The branch is called the *main branch*. Nodes along this branch are now called the *main intermediate conclusions*. The general structure of a proof tree is illustrated in Fig. 5.2, where the main branch is the branch from A to A_n. The formula A, A_1,...,An denote the main intermediate conclusions, and $P_{i,1}$, ..., P_{i,m_i} are the main preconditions for the decomposition step from A_i to A_{i+1}. In other word, the main intermediate conclusions are the only intermediate conclusions depending on A. Furthermore, exactly one of their preconditions depends on A. We are referring to this linear order when we later talk about the previous or subsequent main intermediate conclusions.

2. (*decomposition property*) Main intermediate conclusions are justified by decomposition rules with respect to the previous main intermediate conclusion. The inference along the main branch is therefore a linear process of decomposition of the assertion A.

3. (*composition property*) Other intermediate conclusions are justified by composition rules with respect to the corresponding intermediate conclusion.

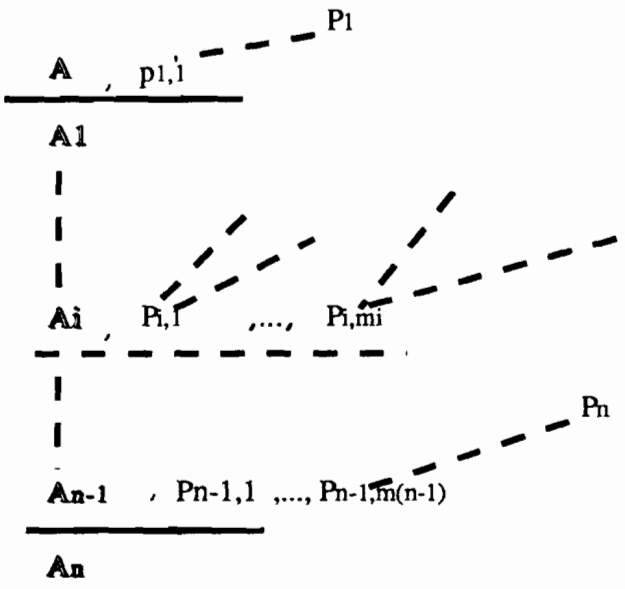

Fig. 5.2

Some very desirable properties can be concluded from the constraints above. Firstly, the derived formula of all proof lines is a subformula of the assertion being applied, or an instance thereof, or, in the third case, the negation of one of the first two cases. We will come back to this in Section 5.2. Secondly, if we construct schemata of proof trees by replacing all the constant symbols in a proof tree, not originally occurring in the assertion being applied, by meta-variables, the total number of the possible tree schemata is finite. This can be proved by the definitions of composition and decomposition rules. The constants for objects a_1, U_1 and F_1 in Fig. 2.1, for instance, can be abstracted to meta-variables a, U and F.

5.1.2. Rewriting towards more Natural Formulas.

The composition and decomposition characteristics alone,unfortunately, are still not enough to cover all the cases encountered in our empirical studies. An exception is showed in the following example:

Example 2:

Suppose we have an assertion of the form $A \vee B \Rightarrow C$. Although the rule $\frac{\neg C}{\neg B}$ fits fully our intuition of its application, the most natural tree we have constructed in Fig. 5.3 does not satisfy the composition and decomposition constraints. One of the "main intermediate conclusions", namely $\neg A \wedge \neg B$, is justified by a rule $\frac{\neg(A \vee B)}{\neg A \wedge \neg B}$, which is neither included as a logic level inference rule nor a decomposition rule.

$$\frac{A \vee B \Rightarrow C \qquad , \qquad \neg C}{\neg(A \vee B)}$$

$$\frac{\neg(A \vee B)}{\neg A \wedge \neg B}$$

$$\frac{\neg A \wedge \neg B}{\neg B}$$

Fig. 5.3

The explanation we have come up with for this phenomenon rests upon the following observation: for formulas of some special patterns there are some other formulas that are logically equivalent but *more natural* for a human. Once a less natural one is derived in a proof process as an intermediate conclusion, it is often transformed (especially when motivated by, for instance, the need of a further composition or decomposition) automatically and implicitly into its more natural equivalent, before the proof proceeds. Therefore, the constraints listed in the previous section should be loosened to allow this kind of inferences on intermediate conclusions. The following is a list of some such rewriting inference rules we have found. The first three of them are quite obvious whereas the last three might be controversial. The criterion to test the naturalness is the principle of linguistic parsimony, emploied by cognitive psychologist studying human daily casual reasoning [Johnson-Laird 83]. The negation of the compound expressions in the first two pairs, for instance, are found difficult to translate in an efficient and straightforward manner. Compared with composition and decomposition constraints discussed in the last sub-section, more empirical studies are needed to set up a more complete set of rewriting inference rules. In addition, from the perspective of a cognitive theory, we are also still confronted with the problem whether to characterize the rewriting rules as elementary.

Less Natural	More Natural	Natural Rewriting Rule
$\neg(A \vee B)$	$\neg A \wedge \neg B$	$\dfrac{\neg(A \vee B)}{\neg A \wedge \neg B}$
$\neg(A \wedge B)$	$\neg A \vee \neg B$	$\dfrac{\neg(A \wedge B)}{\neg A \vee \neg B}$
$\neg\neg A$	A	$\dfrac{\neg\neg A}{A}$
$\neg(A \Rightarrow B)$	$A \wedge \neg B$	$\dfrac{\neg(A \Rightarrow B)}{A \wedge \neg B}$
$\neg \forall x\ Px$	$\exists x\ \neg Px$	$\dfrac{\neg \forall x\ Px}{\exists x\ \neg Px}$

$\neg\exists x\ Px$	$\forall x\ \neg Px$	$\dfrac{\neg\exists x\ Px}{\forall x\ \neg Px}$

Table 5.1

5.2. A Tentative Description of the Run-time Behavior

What we have proposed thus far is essentially merely a combination of constraints, which appears to characterize the set of deductions usually falling under the intuitive concept of the application of a certain assertion. A precise and detailed run time description of the procedures is nevertheless still beyond our present knowledge and far beyond the scope of this paper. We are going, all the same, to at least provide some rough ideas of the possible run time behavior, gained by solely resorting to introspection.

As indicated at the outset of this chapter, the application of an assertion by resorting to a logic level NE may happen during both active proof constructions or passive proof checking. In fact, whenever the corresponding assertion level rule does not reside in WM and is therefore be at the disposal of the autonomous reasoning procedures, a logic level proof segment is usually constructed, with variation in degrees of detail and explicitness. Let us illustrate this by Schubert's well known steam-roller problem [Stickel 86], under the circumstance of proof checking.

Example 3.

The entire problem is given in natural language:

Wolves, foxes, birds, caterpillars, and snails are animals, and there are some of each of them. Also there are grains, and grains are plants. Every animal either likes to eat all plants or all animals much smaller than itself that like to eat some plants (Axiom). Caterpillar and snails are much smaller than birds, which are much smaller than foxes, which in turn are much smaller than wolves. Wolves do not like to eat foxes or grains, while birds like to eat caterpillars and snails like to eat some plants. Therefore there is an animal that likes to eat a grain-eating animal.

The following is the axiom in our encoding, where "a", "a´", and "p", "p´"are variables of sorts, animal and plant respectively:

Axiom 1. a,a´ :Animal, p,p´:Plant

$$\forall a\ (\forall p\ \mathrm{Eats}(a, p) \vee$$

$$(\forall a'\ a'<a \wedge \exists p'\ \mathrm{Eats}(a', p') \Rightarrow \mathrm{Eats}(a, a')) \qquad 5.1$$

Suppose the derivation of Eats(a1, p1) from a1'<a1, Eats(a1', p1') and ¬Eats(a1, a1') is claimed to be justified by applying axiom 5.1. And suppose further, that the corresponding assertion level inference rule

$$\frac{a'<a,\ Eats(a', p'),\ \neg Eats(a, a')}{Eats(a, p)}$$

where "a", "a´", and "p", "p´"are meta-variables 5.2

is either not yet acquired at all or not residing in WM. The following logic level proof may then be constructed, with the corresponding proof tree shown in Fig. 5.4:

(1)	1	$\vdash \forall a\ (\forall p\ Eats(a, p) \vee$	
		$(\forall a'\ a'<a \wedge \exists p'\ Eats(a', p') \Rightarrow Eats(a, a'))$	Hyp
(2)	1	$\vdash \forall p\ Eats(a1, p) \vee$	
		$(\forall a'\ a'<a1 \wedge \exists p'\ Eats(a', p') \Rightarrow Eats(a1, a'))$	$\forall D(1)$
(3)	2	$\vdash a1'<a1 \wedge Eats(a1', p1') \wedge \neg Eats(a1, a1')$	Hyp
(4)	2	$\vdash \neg(a1'<a1 \wedge \exists p'\ Eats(a1', p') \Rightarrow Eats(a1, a1'))$	$\Rightarrow D'(3)$
(5)	2	$\vdash \neg(\forall a'\ a'<a1 \wedge \exists p'\ Eats(a', p') \Rightarrow Eats(a1, a'))$	$\forall D'(4)$
(6)	1,2	$\vdash \forall p\ Eats(a1, p)$	$\vee D(2,5)$
(7)	1,2	$\vdash Eats(a1, p1)$	$\forall I(6)$

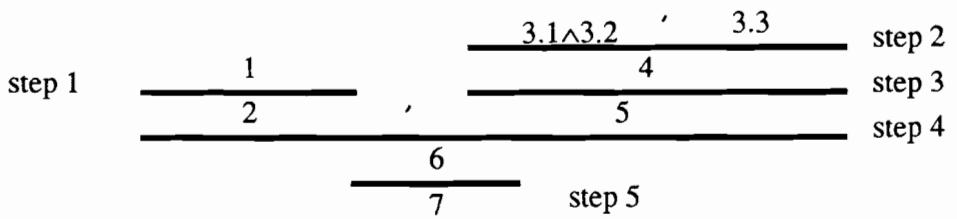

Fig. 5.4

Some explanations may be necessary: The rule $\Rightarrow D'$ and $\forall D'$ used in step 4 and 5 are associated with rule $\Rightarrow D$ and rule $\forall D$ respectively. Extended premise conditions are used, too.

Most importantly, by no means do we claim that this logic level NE is actually carried out in run-time by literally writing down the proof above. This proof is much often or even exclusively followed by reading the assertion being applied by pointing to the corresponding subformula, whose instance or negation thereof is the intermediate conclusion at this point. It can be easily performed since all the intermediate conclusions of the expanded proof are either instances of subformulas of the assertion or negations thereof, as is proved in Section 5.1. The process is made still easier, since it can be performed in a quasi-linear way, i.e., along the main branch, on the ground that in most of the practical cases, proofs to establish main premises in a composition manner (represented by subtrees rooted at a main premise) are quite simple. We might even predict the order of this implicit reasoning process underlying the check of an

assertion level step. A plausible conjecture is that, given the general schema of proof trees in Fig. 5.2, the implicit reasoning proceeds in the following order: the composition of the main preconditions $P'_{1,1}$, ..., P'_{1,m_1}, the decomposition from A to A_1; the composition of the preconditions $P'_{2,1}$, ..., P'_{2,m_2}, the decomposition from A_1 to A_2; and so forth. Fig. 5.4 is an example. In order to provide a really procedural description, however, we must still first resolve problems as to how a reasoner chooses from parallel executable steps (such as step 1 and step 2 in Fig. 5.4), as well as how he comes to the appropriate preconditions at each step.

Before leaving this chapter, now there is the question if the procedure defined above really all the covers cases a human will intuitively accept as the application of some assertion? This, is obviously an empirical issue. What we may claim is only, that all the reasoning carried out by this procedure seems to be fully in accord with our intuition. No claims, nonetheless, can be made vice versa, since very possibly our intuitive notion is not all-or-none. For instance, while one may think the derivation of $a \in F \vee b \in F$ from $a \in U \vee b \in U$ and $U \subseteq F$ is an application of the subset definition, we might as well accept this as a reasoning process involving firstly a splitting into two cases and then applying the subset definition twice. Our hope is therefore, that we have at least captured the kernel of this notion, although it might have a hazy border.

Chapter 6. Compound Inference Rules at the Assertion Level

In this chapter we introduce a structured collection of assertion level inference rule schemata, which on the one hand cover the complete set of inferences carried out by the primitive procedure described in the last chapter, and on the other hand, really have the chance to enter WM and thus become available to the reasoner. In sec. 6.1,we first set up a compact but logically complete structure to organize the rule schemata, and then elaborate on its cognitive import in sec. 6.3. Some more examples can be found in sec. 6.2.

6.1. A Structured Representation

We shall first try to pin down the set of assertion level inference rules which may enter the WM. At the same time we will show that for each assertion, the corresponding sum total of inference rule schemata is finite. From the perspective of our computational model, this set of inference rules is exactly the set of possible outputs of the operator generating inference rules from assertions. It is of course again a simplification to ascribe the generation of inference rules from assertions to a single operator. Apart from the possibility of deriving new rules after calling the assertion application procedure, as illustrated in Chapter 1, there are certainly some other occasions where inference rules are generated directly from assertions. As will be discussed in section 6.3, however, we believe it is reasonable to assume that they are all logically equivalent. The entire discussion of this chapter is based on this assumption. We will also prove, that the operator generating new rules associated to existing rules (see Chapter 4) will not generate any really new rules at the assertion level. Let's now examine our subset example in more detail.

Example 1. (continued)

Suppose that the assertion application procedure has just constructed a proof presented in the tree form in Fig. 5.1. Our assumption is that possibly the reasoner learns the following inference rule schema as well, apart from merely drawing a concrete conclusion $a1 \in F1$ from premises $a1 \in U1$ and $U1 \subseteq F1$:

$$\frac{a \in U,\ U \subseteq F}{a \in F} \qquad 6.1$$

where a, U and F are meta-variables for object variables. More generally, hand in hand with deductive steps supported by proof trees represented in Fig. 5.2, a corresponding inference rule taking the form of 6.2 may be acquired:

$$\frac{P'_1,\ldots,P'_m}{A'_n} \qquad 6.2$$

where $P'_1,...,P'_n$ are formula schemata obtained from $P_1,...,P_n$ (the formulas attached to the leaves, except the assertion A itself) and A_n' is the formula schemata obtained from A_n, attached to the root. The formula schemata are obtained from the corresponding formulas by replacing constant symbols not originally occurring in A by new meta-variables. Obviously, these constant symbols must occur in formulas serving as premises, such as $a \in U$ and $U \subseteq F$ in our example. This procedure produces schemata of proof trees. And there is obviously a one-to-one correspondence between assertion level inference rules and tree schemata, under the above mentioned assumption that all other rule acquiring procedures are equivalent to this reasoning-and-abstraction procedure described here, with respect to the input-cutput relation. We may henceforth talk about proof trees and proof tree schemata interchangeably. In the following, therefore, we are going to establish a more structured organization of inference rules, based on this correspondence.

Let us first introduce some further terminologies. Let $\mathcal{A}$ be an assertion and $\mathcal{B}$ a set of logic level inference rules. Now let $R_{\mathcal{A},\mathcal{B}}$ designate the set of inference rules acquired by constructing NEs using exclusively logical level rules in $\mathcal{B}$. It is especially interesting to examine the set of proof tree schemata designated by $Tree_{\mathcal{A},\mathcal{B}}$ such that for all $r \in R_{\mathcal{A},\mathcal{B}}$, there is a tree schema $t \in Tree_{\mathcal{A},\mathcal{B}}$, such that r can be accounted for by a subtree of t. Rule r is called *terminal* if it is accounted for by a tree $t \in Tree_{\mathcal{A},\mathcal{B}}$, rather than a proper subtree. Apparently $Tree_{\mathcal{A},\mathcal{B}}$ is unique. Below is a constructive definition:

i) Start with the tree in Fig.6.1.a, which corresponds to the rule $\frac{}{A \vdash \mathcal{A}'}$

ii) If there is a tree t in the form of Fig. 6.1.b, and

a) $r=\frac{A \vdash a,\ A \vdash p_1,...,A \vdash p_n}{A \vdash Q} \in \mathcal{B}$ is a decomposition rule with respect to a, now if there exists a substitution σ, such that $A'\sigma=a\sigma$, then extend t to a tree t' in form of Fig. 6.1.c.

b) If $\frac{A \vdash P}{A \vdash Q}$ is a natural rewriting rule, a similar extension is made (n=0 in Fig. 6.1.c)

iii) If there is a tree t in the form of Fig. 6.1.b, and $r=\frac{A \vdash p'_1,...,A \vdash p'_n}{A \vdash Q} \in \mathcal{B}$ is a composition rule with respect to Q, now if there exists a substitution σ, such that $p\sigma=Q\sigma$, then extend t to a tree t' in form of Fig. 6.1.d.

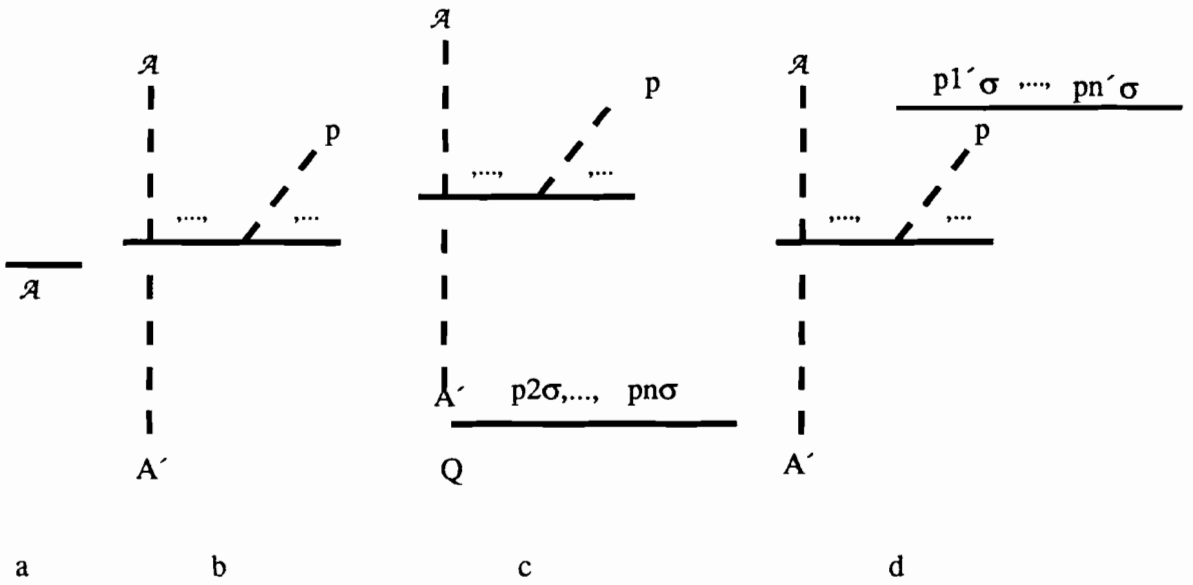

Fig. 6.1

Some explanations: i) is an initialization, where a tree with only one node is introduced, corresponding to the initial inference rule $\frac{}{A \vdash \mathcal{A}}$ ii) and iii) extend existing trees by decomposing the root or the leaves.

The informations contained in this set is in fact rather redundant, since many rules accounted for by one tree schema are often associated to rules accounted for by another tree schema, in the way described in Chapter 4. This, reflects the fact that a rule can either be acquired directly, in a reasoning-and-abstraction manner, or it can be derived as an associated rule of another acquired rule. Let us illustrated this in the example below:

Example 1. (Continued)

With a rule $\frac{a1 \in U1,\ U1 \subseteq F1}{a1 \in F1}$ already acquired from the subset definition, supported by the NE illustrated in Fig. 5.1, it is only natural for a human to be able to apply the following associated rule:

$$\frac{a1 \in U1,\ a1 \notin F1}{U1 \not\subseteq F1}$$

This, however, has as a matter of fact a corresponding NE of its own:

$$\dfrac{\dfrac{\forall F,U\ U\subseteq F \Leftrightarrow \forall x\ x\in U \Rightarrow x\in F}{U1\subseteq F1 \Rightarrow \forall x\ x\in U1 \Rightarrow x\in F1} \qquad \dfrac{\dfrac{a1\in U1 \quad , \quad a1\notin F1}{\neg(a1\in U1 \Rightarrow a1\in F1)}}{\neg(\forall x\ x\in U1 \Rightarrow x\in F1)}}{U1 \not\subseteq F1}$$

Fig. 6.2

In general, if Fig 6.3.a is the corresponding tree schema for a rule $\dfrac{c1,c2,b1}{b2}$, acquired from assertion A, the corresponding tree schema for the associated rule $\dfrac{b1,\neg b2,\ c1}{\neg c2}$ can certainly be constructed, using corresponding associated logic level rules, as shown in Fig. 6.3.b.

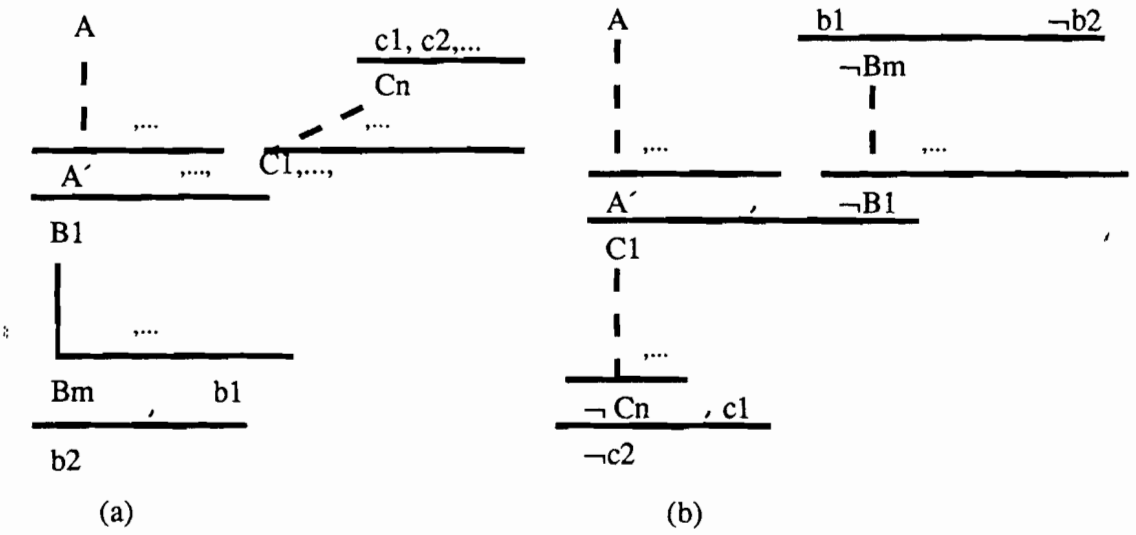

Fig. 6.3

This argument leads exactly to the following property, that makes a more succinct representation possible:

$$R_{\mathcal{A},Assoc(\mathcal{B})\cup\mathcal{B}} = R_{\mathcal{A},\mathcal{B}} \cup Assoc(R_{\mathcal{A},\mathcal{B}}) \qquad 6.3$$

where $\mathcal{B}$ is an arbitrary set of logic level inference rules. A natural corollary is obtained in conjunction with properties 4.5 and 4.6:

$$Assoc(R_{\mathcal{A},Assoc(\mathcal{B})\cup\mathcal{B}}) \subseteq R_{\mathcal{A},Assoc(\mathcal{B})\cup\mathcal{B}} \qquad 6.4$$

From a static perspective, this means if all rules associated to elementary rules are already at the disposal of the operator generating assertion level inference rules from assertions, the derivation of associated rules will not bring rules really new at the assertion level. Now let $Rules_{\mathcal{A}}$

designate the complete set of inference rules acquirable from an assertion $\mathcal{A}$, based exclusively on the set of cognitively elementary inference rules Elementary , then

$$\begin{aligned}\text{Rules}_{\mathcal{A}} &= R_{\mathcal{A},\text{Elementary}\cup\text{assoc(Elementary)}} \cup \text{Assoc}(R_{\mathcal{A},\text{Elementary}\cup\text{assoc(Elementary)}}) \\ &= R_{\mathcal{A},\text{Elementary}\cup\text{assoc(Elementary)}} && \text{by 6.4} \\ &= R_{\mathcal{A},\text{Elementary}} \cup \text{Assoc}(R_{\mathcal{A},\text{Elementary}}) && \text{by 6.3}\end{aligned}$$

Now we have isolated a subset of the rules which can be represented by $\text{Tree}_{\mathcal{A},\text{Elementary}}$ in an very compact way, that also plays an epistemologically more central role. Thanks to the compactness of the tree form, for almost all examples, $\text{Tree}_{\mathcal{A},\text{Elementary}}$ consists usually of only one or two trees. The special cognitive import of these more kernel rules will be discussed in section 6.3. It is to be indicated here, that some logical redundancy still remains in this representation, since there are elementary rules associated with each other. Within the elementary rules we have proposed, for instance, the two $\vee$D rules

$$\frac{\mathcal{A}\vdash P\vee Q,\ \mathcal{A}\vdash\neg P}{\mathcal{A}\vdash Q}\ ,\quad \frac{\mathcal{A}\vdash P\vee Q,\ \mathcal{A}\vdash\neg Q}{\mathcal{A}\vdash P}$$

are associated with each other.

Now we finish the discussion of the subset example used throughout this paper by showing its $\text{Tree}_{\mathcal{A},\text{Elementary}}$ and a listing of some of the rules contained in $\text{Rules}_{\mathcal{A}}$.

Example 1. (Continued)

Two trees are needed since the equivalence ($\Leftrightarrow$) is understood as the short hand of the conjunction of two implications and therefore can be decomposed in two different ways. The subset definition is repeated below:

$$\forall F,U\ U\subseteq F\Leftrightarrow\forall x\ x\in U\Rightarrow x\in F$$

The following is a list of some of the rules in $R_{\mathcal{A}}$, and their corresponding tree schemata.

	Inference Rule	**Derivation(Tree or Association)**
(1)	$\dfrac{A\vdash a1\in U1,\ A\vdash U1\subseteq F1}{A\vdash a1\in F1}$	Tree 6.4.a
(2)	$\dfrac{A\vdash a1\notin F1,\ A\vdash U1\subseteq F1}{A\vdash a1\notin U1}$	Associated with (1)
(3)	$\dfrac{A\vdash a1\in U1,\ A\vdash a1\notin F1}{A\vdash U1\not\subseteq F1}$	Associated with (1)
(4)	$\dfrac{A\vdash a1\in U1\Rightarrow a1\in F1}{A\vdash U1\subseteq F1}$ where a1 does not occur in A	Tree 6.4.b

(5) $\dfrac{A\vdash\forall x\ x\in U1\Rightarrow x\in F1}{A\vdash U1\subseteq F1}$ Subtree of 6.4.b

(6) $\dfrac{A\vdash U1\not\subseteq F1}{A\vdash\neg\forall x\ x\in U1\Rightarrow x\in F1}$ Associated to (5)

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall F,U\ U\subseteq F\Leftrightarrow\forall x\ x\in U\Rightarrow x\in F}{U1\subseteq F1\Leftrightarrow\forall x\ x\in U1\Rightarrow x\in F1}}{U1\subseteq F1\Rightarrow\forall x\ x\in U1\Rightarrow x\in F1}}{\forall x\ x\in U1\Rightarrow x\in F1}}{a1\in U1\Rightarrow a1\in F1}\quad,\quad a1\in U1}{a1\in F1}\qquad U1\subseteq F1$$

Fig. 6.4.a

$$\dfrac{\dfrac{\dfrac{\forall F,U\ U\subseteq F\Leftrightarrow\forall x\ x\in U\Rightarrow x\in F}{U1\subseteq F1\Leftrightarrow\forall x\ x\in U1\Rightarrow x\in F1}}{(\forall x\ x\in U1\Rightarrow x\in F1)\Rightarrow U1\subseteq F}\quad,\quad\dfrac{a1\in U1\Rightarrow a1\in F1}{\forall x\ x\in U1\Rightarrow x\in F1}}{U1\subseteq F1}$$

Fig. 6.4.b

Apparently, rules (1),(2),(3) are associated with each other, albeit independently derivable. Rule (4) has no associated rules because of its variable condition. Rule 5, the associated rule of the terminal rule 6, is non-terminal. Finally, we still want to use this example to illustrate how our extended conditions for applications of inference rules enhances the naturalness of the rules. Rule (4), for instance, allows an extra class of derivations, namely inferring $A\vdash U'1\subseteq F'1$ from A, $a'1\in U'1\vdash a'1\in F'1$, as far as a'1,U'1 and F'1 do not occur in A.

6.2. Examples

Knowledge represented in a logic based language can usually be conceived of a as system of assertions. According to the theory proposed in this paper, these assertions should directly increase the deductive power of a reasoner. Although in most of the current languages no modularity is supported, assertions can usually be grouped together into theories. And theories, in turn, form a hierarchical structure reflecting the dependency relations [Kerber 89.b]. We are going to provide concrete examples of assertion level inference rules in several different theories, to demonstrate the growth of the natural calculus, hand in hand with the growth of a knowledge base.

A. Assertions at the Logic Level.

The growth of the natural calculus can best be observed in mathematical reasoning. Mathematical theories are in general set up hierarchically, and even within a theory, subordinate theorems and lemmas are also first proved, thus paving additional routes leading to a main theorem. In terms of our computational model, this means that along with the development of a mathematical theory, a human being working with it is provided with new, more abstract means of drawing conclusions, i.e., applying intermediate theorems. To prove a theorem in group theory, for instance, a human may use some theorems of set theory.

On the bottom of the hierarchy of mathematical knowledge, we can usually find a layer of assertions pertaining directly to logic per se. To extend their reasoning repertoire beyond the basic inference rules directly supported by the calculus like that of Gentzen, and to provide the layers above with more direct routes, logicians first prove some new assertions.

For example, once the formula

$$\exists_x P_x \vee \exists_x Q_x \Leftrightarrow \exists_x P_x \vee Q_x$$

is proved as a theorem, the following two inference rules may be added to the collection of inference rules at our disposal:

$$\frac{\exists_x P_x \vee \exists_x Q_x}{\exists_x P_x \vee Q_x} \quad , \quad \frac{\exists_x P_x \vee Q_x}{\exists_x P_x \vee \exists_x Q_x}$$

The following are some more examples:

Logic Tautologies	Inference Rules
$\neg\forall_x P_x \Rightarrow \exists_x \neg P_x$	$\frac{\neg\forall_x P_x}{\exists_x \neg P_x} \quad , \quad \frac{\neg\exists_x \neg P_x}{\forall_x P_x}$

$\forall_x \neg P_x \Rightarrow \neg\forall_x P_x$	$\frac{\forall_x \neg P_x}{\neg\forall_x P_x}$, $\frac{P_a}{\neg\forall_x \neg P_x}$
$\neg P_a \Rightarrow \neg\forall_x P_x$	$\frac{\neg P_a}{\neg\forall_x P_x}$
$\forall_x P_x \wedge Q_x \Leftrightarrow \forall_x P_x \wedge \forall_x Q_x$	$\frac{\forall_x P_x \wedge Q_x}{\forall_x P_x}$,...
$\forall_x P_x \vee \forall_x Q_x \Rightarrow \forall_x P_x \vee Q_x$	$\frac{\forall_x P_x \vee \forall_x Q_x}{\forall_x P_x \vee Q_x}$, $\frac{\forall x\ \forall_x P_x \vee \forall_x Q_x}{P_a \vee Q_a}$,...
$\exists_x P_x \wedge Q_x \Rightarrow \exists_x P_x \wedge \exists_x Q_x$	$\frac{\exists_x P_x \wedge Q_x}{\exists_x P_x}$,...

This splitting of inference rules pertaining to logic itself resolves a problem troubling cognitive psychologists designing natural logics [Braine 78]. To cover their empirical data, their natural logics have to be enriched by inference rules which are by nature obviously compound. In our model, these data can be explained by assuming a knowledge base containing assertional knowledge. It is nevertheless worth noting that it is difficult to set up a criterion to distinguish the elementary rules from logic level rules derived from logic theorems, the border between the two may again be hazy.

B. Assertions at the Epistemological Level.

There is an epistemological level in concept description languages like for example the KL-One family [Brachmann 78], that provides a hopefully adequate basis for the construction of conceptual level objects. These, in a logic based representation language are all hidden implicitly in the way of encoding concepts as well as the various relations among them into logical formulas [Hayes 79]. The usual reasoning inherent to those concept constructing primitives can consequently be carried out by applying the corresponding assertions.

Let us look at some simple examples. If we have the full power of higher order logic, and two predicates Property(A, P) and Subsume(A,B) with the intended meaning that P is a property of concept A and concept A subsumes concept B, a possible axiomatization may be:

$\forall$A, P Property(A P) $\Leftrightarrow$ ($\forall$x Element(x A) $\Rightarrow$ P (x))

$\forall$A, B Subsume(A, B) $\Rightarrow$ ($\forall$x Element(x B) $\Rightarrow$ Element(x A))

Now the well known inheritance relation can be expressed by the following assertion derivable from the two axioms above:

$\forall$A, B, P, x (Property(A P) $\wedge$ Subsume(A, B) $\wedge$ Element(x B)) $\Rightarrow$ P (x).

which enables us to apply a new inference rule:

$$\frac{\text{Property(A P), Subsume(A, B), Element(x B)}}{\text{P(x)}}.$$

Another form of inheritance might be well captured by an assertion as well:

$$\forall A, B, P\ (\text{Property}(A\ P) \wedge \text{Subsume}(A, B)) \Rightarrow \text{Property}(B\ P)$$

which would account for the derivation of "*all students are mortal*" from "*all human beings are mortal*" and "*all students are human beings*".

C. A Puzzle Example

Thus far, we have discussed examples of logical assertions, fundamental for mathematical reasoning, and assertions related to epistemological constructs, fundamental to common sense reasoning. And throughout the paper, we have used a subset example considered typical for mathematical reasoning. Now we are going to continue with Schubert's steam-roller problem, to show that similar observations can be made in deductive common sense reasoning as well.

Example 3. (continued)

For the convenience of discussion, the encoding of the axiom is repeated below.

Axiom 1. a,a´ :Animal, p,p´:Plant

$$\forall a\ (\forall p\ \text{Eats}(a, p) \vee$$

$$(\forall a'\ a'<a \wedge \exists p'\ \text{Eats}(a', p') \Rightarrow \text{Eats}(a, a')) \qquad 5.1$$

Although $\text{Tree}_{\mathcal{A},\text{Elementary}}$ consists of two trees, due to the redundancy of the two symmetric $\vee$D rules, only one is shown in Fig. 6.5. Since the two $\vee$D rules are associated with each other, this tree is complete by itself, as discussed in the last section. Some inference rules are listed below:

Inference Rules	**Derivation(Tree or Association)**
(1) $\dfrac{\neg\text{Eats}(a, p),\ a'<a,\ \text{Eats}(a', p')}{\text{Eats}(a, a')}$	Tree in Fig. 6.5
(2) $\dfrac{a'<a,\ \text{Eats}(a', p'),\ \neg\text{Eats}(a, a')}{\text{Eats}(a, p)}$	Associated to (1)
(3) $\dfrac{\neg\forall p\ \text{Eats}(a, p),\ a'<a,\ \text{Eats}(a', p')}{\text{Eats}(a, a')}$	Subtree of Fig. 6.5
(4) $\dfrac{a'<a,\ \text{Eats}(a', p'),\ \neg\text{Eats}(a, a')}{\forall p\ \text{Eats}(a, p)}$	Associated to (3)

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall a\,(\forall p\ \mathrm{Eats}(a,p)\vee(\forall a'\ a'<a\wedge\exists p'\ \mathrm{Eats}(a',p')\Rightarrow\mathrm{Eats}(a,a'))}{(\forall p\ \mathrm{Eats}(a1,p)\vee(\forall a'\ a'<a1\wedge\exists p'\ \mathrm{Eats}(a',p')\Rightarrow\mathrm{Eats}(a1,a'))}\quad \dfrac{\neg\,\mathrm{Eats}(a1,p2)}{\neg\forall p\ \mathrm{Eats}(a1,p)}}{\forall a'\ a'<a1\wedge\exists p'\ \mathrm{Eats}(a',p')\Rightarrow\mathrm{Eats}(a1,a')}\quad \mathrm{Eats}(a1',p1)}{a1'<a1\wedge\exists p'\ \mathrm{Eats}(a1',p)}}{a1'<a1\wedge\exists p'\ \mathrm{Eats}(a1',p)\Rightarrow\mathrm{Eats}(a1,a1')}\quad a1'<a1\wedge\exists p'\ \mathrm{Eats}(a1',p')}{\mathrm{Eats}(a1,a1')}$$

Fig. 6.5

6.3. Cognitive Status of Assertion Level Inference Rules

In this section, we are going to discuss briefly the questions as to whether every assertion level rules plays the same role. Do some of them possess a more significant cognitive status? Are rules stored in reality as an unorganized congregation, or does the tree form structure suggested in section 6.1 have a cognitive import?

If restricted to the reasoning-and-abstraction view of rule acquisition, most of the answers for the above questions are negative. As already indicated, however, there is at least another procedure introducing inference rules into WM, the one carrying out the task of comprehending assertions. The effect of the understanding of an assertion must include the introduction of at least some of the corresponding rules, on the ground that a natural expansion is not always involved in assertion level proof steps, even when an assertion is applied for the first time in a context. It is therefore a plausible conjecture that what happens in the understanding process of assertions is a natural expansion process at the meta-level, involving meta-variables later appearing in assertion level inference rules. On the ground that only elementary rules are residing permanently in WM, in contrast to rules associated to them, which are not derived unless motivated, it can be further concluded that rules accounted for by trees in $\mathrm{Tree}_{\mathcal{A}\mathit{Elementary}}$ are really learned first in this comprehension process, since the elementary rules alone are involved in their NEs. Furthermore, since NEs of rules accounted for by subtrees are part of the NEs of the terminal rules, there is really a reason to assume a structure similar to $\mathrm{Tree}_{\mathcal{A}\mathit{Elementary}}$. Up to now it should be clear that the division of assertion level rules into those accounted for by $\mathrm{Tree}_{\mathcal{A}\mathit{Elementary}}$ and those associated to them is nor entirely arbitrary nor is it only for the sake of mathematical clarity. Although no precise prediction is possible in our theory, it is very likely that the first group of rules are acquired during the comprehension process, while members of the latter are derived form members of the former, when motivated in a reasoning process.

It might be an over-generalization to claim that the comprehension process will always result in the memorization of $Tree_{\mathcal{AE}lementary}$, since the situation can be more complicated sometimes. Take the $Tree_{\mathcal{AE}lementary}$ of the two assertions in Fig. 6.6 for example, both of them consist of two trees. What is special here ist that, the two trees have in both of the cases a common subtree. It is possible that these common subtrees have a better chance to be memorized, while a derivation of the terminal rules needs to be motivated.

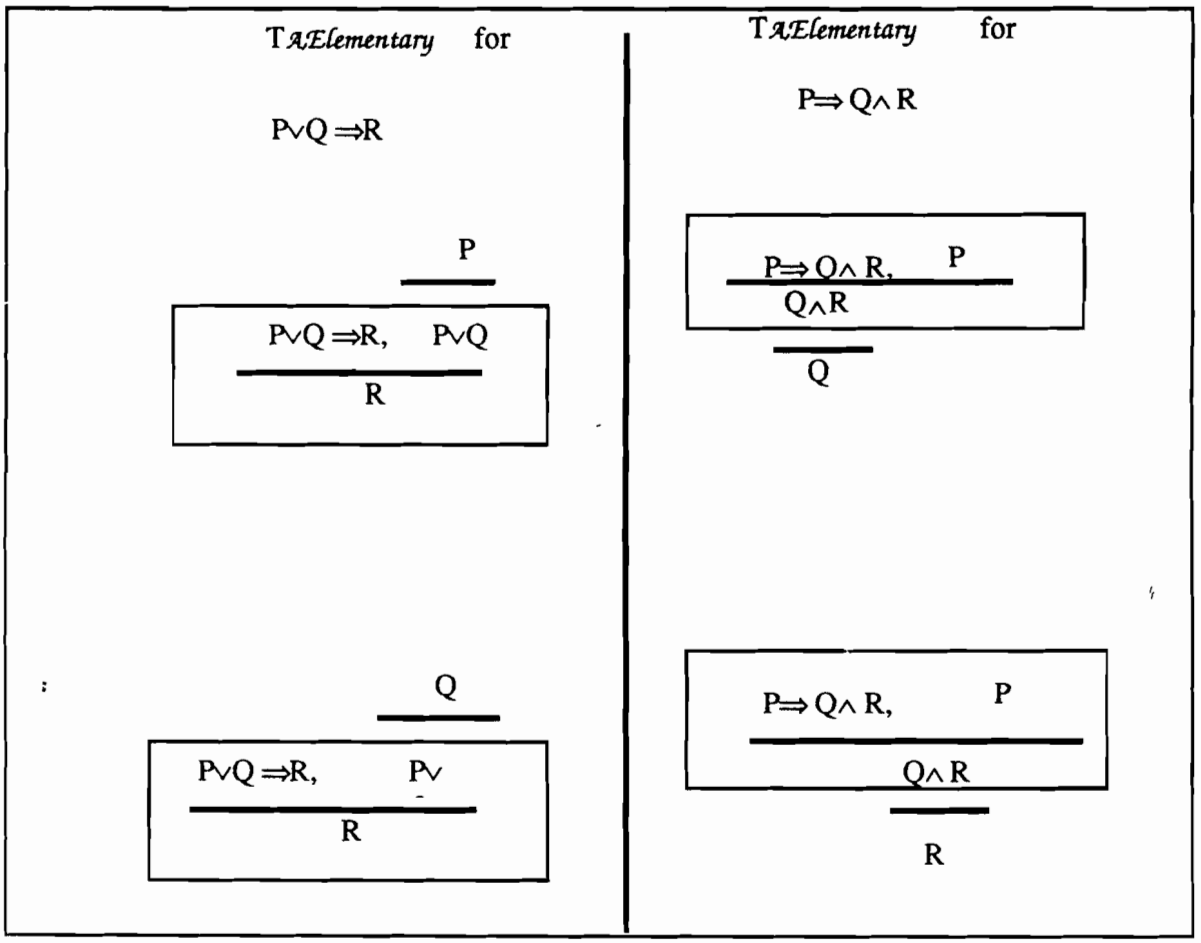

Fig. 6.6

Chapter 7. Related Works and Applications

In the following sections we want to discuss two sorts of related works: theories dealing with related issues such as natural logics or mental reasoning, and AI applications where these concepts play or will play a major role.

7.1. Related Works

Our set of cognitively elementary inference rules originates directly from the long tradition of the study on natural logics, carried out both by logicians and later by cognitive psychologists. The word "natural" was first used by Gerhard Gentzen [Gentzen 30] to describe his logic. He studied the ways in which mathematical inferences are drawn and decided to design a calculus containing a relatively large number of inference rules which "came as close as possible to actual reasoning", and thus took an important departure from the calculi in the Churchean tradition. And more recently, natural logic became an active research topic of cognitive scientists, aiming at discovering internal structures that could account for human reasoning competence. Granted the problems that arise from the assumption of a mental logic [Johnson-Laird 83] [Holland et al 86] [Lakoff 87], it remains an appealing proposal. Various versions of natural logics are consequently developed for this purpose. While Gentzen's logic can be seen as aiming at a characterization of human formal reasoning, with mathematics as a typical example; the second group of natural logics are either designed to captured the nature of casual daily reasoning [John-Laird 72, 83], reasoning with English connectives [Braine 78] or to serve as a semantic interpretation of natural languages [Lakoff 70]. Since we are primarily interested in formal reasoning, our set of elementary inference rules is essentially adopted from Gentzen, in a linearized version first described in Andrews [Andrews 80].

Secondly, the central issue of this paper, i.e. the characterization of reasoning intuitively called the application of assertions, is closely related to one of John-Laird's "effective procedure" [John-Laird 83], aimed at accounting for spontaneous daily reasoning. In spite of the difference of task domains and the basic theoretical assumptions, both studies are confronted with a similar problem and have opted for a similar solution. Given a set of premises, the problem confronts John-Laird as a cognitive psychologist is how to explain that only a small subset of all the valid conclusions are actually drawn by a human being under ordinary circumstances. We, on the other hand, are required to delineate the subset of all the valid inferences intuitively understood as the application of an assertion, given an assertion and some other premises. To this end, both approaches have tried to find constraints that appear to characterize the subset in concern. The composition and decomposition constraints proposed here are of purely syntactical nature, as for a psychological explanation we can only refer to the primitiveness of the procedure carrying out this reasoning (see section 5.2). In comparison, John-Laird's characterization is more of a semantic nature and it may be more superior as a

psychological explanation. He first proposed a measure of the information content of formulas, and then postulates that "no conclusion contains less semantic information than the premises on which it is based or fails to express that information more parsimoniously". This, establishes to a greater extent the meaningfulness of the conclusions drawn. Unfortunately, neither can his measure be extended to predicate logic [John-Laird 83] [Hintika 73], nor can his criterion split deductions into units called the applications of single assertions. Comparing the two approaches mathematically, within the realm of propositional logic, the set of *meaningful* reasoning carried out by John-Laird's procedure is in general more complex, i.e., involving more than one application of an assertion. Finally, we want to point out our difference with respect to the general theoretical assumptions. While our discussion is entirely built on top of the assumption of the existence of mental inference rule schemata as an internal structure responsible for deductive competence, this assumption is disputed by John-Laird in general. Despite of his success in explaining daily casual reasoning by means of other internal representations, we believe it remains an appropriate assumption in regards to formal deductive reasoning, where the individuals are usually assumed to have received training in formal logic. Furthermore, our framework makes possible the accommodation of the compound rules,which help to explain some other observations (see section 6.3).

7.2. Possible Applications in AI

Although not directly on the subject of natural logic and mental reasoning, there is enough work showing the need of such concepts in various AI applications. We hope that our new model of deductive reasoning, particularly the notion of an extensible natural calculus, will give some weight to these applications.

First there is the problem of transforming machine generated proofs or arguments into natural language, which in effect set us on the line of this research originally. It has long been a problem for natural language generation systems to find heuristic strategies capable to make decisions on the identification of important argument steps and on the deleting of others [Chester 76] [McDonald 83]. And it comes at no suprise that the context dependent strategies that are borrowed from traditional discourse theories do not fare well, since the raw data (proofs written in Gentzen's natural deduction) is not the product of a human deductive apparatus, as was always implicitly taken for granted. In earlier papers, we have shown that DNPs encoded in a natural calculus including assertion level inference rules have proved to be an appropriate basis for further transformation [Huang 89, 90]. Since many results have already been achieved in the transformation of machine generated proofs into well structured natural deduction proofs [Lingenfelder 90] [Pfenning 90], we have designed algorithms to transform natural deduction proofs into DNPs. Moreover, on the ground of the observation that the programs producing natural deduction proofs have all extended the natural deduction calculus, by including some assertion level inference rule pertaining to the logic layer, we believe it is very likely that we can perform the entire transformation in one step. This, will be treated in a separate paper.

A second application area concerns human-machine communication. There is a trend in developing knowledge based systems, that are required to support queries of the type "Why...". To this end, obviously, a simple answer of the traditional form "Yes" or "No" will not suffice. What is needed is exactly an argumentation in a natural calculus.

Finally, the procedure applying assertions will enable a cognitively more adequate interface for inter-active theorem provers or proof checkers along the tradition of Nuprl [Constable et al 86] [Felty et al 88] [Guttman et al 90] [Paulson 89] [Pfenning et al 90]. Apart from matching against elementary inference rules, the user should have the choice of drawing new conclusions by calling a procedure which applies a known assertion to them in the context (normally an axiom, a theorem or a lemma) on some other assertions, serving as premises.

Chapter 8. Conclusion and Future Work

This paper deals with issues concerning natural logic and cognitive models for formal deductive reasoning. Different from similar research carried out in cognitive psychology, where psychological explanations are the main concern, our study is directly motivated by the practical need of presenting proofs or arguments found by machines to a human user in an appropriate way. By analyzing the blueprint of the entire transformation process, we have isolated explicitly an intermediate representation, called detailed natural proofs (DNP), that possesses the quality of a mathematical abstraction of the actual output of the human deductive apparatus. In order to find a formalism suitable for the encoding of DNPs, preliminary empirical studies on human mathematical proofs are carried out, leading to a new computational model for human deductive reasoning. Further sophistication is obtained by adding to the traditional models a primitive procedure carrying out the application of assertions. Hand in hand with this procedure, secondly, an operator is added to account for the acquisition of logically compound inference rules. In this model therefore, the output of a reasoning process is a proof consisting of a sequence of proof steps each justified either by an inference rule, or by an assertion. And the latter, can be replaced in DNPs by acquired assertion level inference rules, without effecting serving as an intermediate representation for further transformation. Now the extensible set of inference rule, including cognitively elementary inference rules and acquired compound rules alike, is called the natural calculus.

The topic of human deductive reasoning however is so rich that our study only indicates a beginning. Among the issues to be addressed in the future, we want to mention first that we need a more semantic explanation for our constraints, probably along the line of Johnson-Laird. Secondly, refinements with respect to run-time behavior must be made. In particular, an *user model* has to be constructed for someone reading a proof, that provides predictions about whether an assertion level step will be understood by applying the special procedure or by matching against an assertion level inference rule schema, acquired in a previous context. This information will provide more guidance to the decision making procedures in the text planners [Huang 90]. Finally, it would certainly be interesting to integrate results on proof planning into such a model [Bundy 87] [Bundy et al 88], and thereby provide new means for structuring DNPs.

Finally, we hope this model may be extended to the sort of reasoning that is beyond the realm of first order predicate logic. An important future research topic is therefore to test the ideas presented in this paper in domains covered by logics of stronger expressive power, for instance modal logics or logics with sort structures [Prior 67] [Rescher 71] [Gabbay 84] [Schmidt-Schauß 89], as well as logics of higher order [Andrews 86].

Acknowledgement

I would like to thank Christoph Lingenfelder who carefully read an earlier version of this paper, and Dan Nesmith for many discussions on the initial ideas which led to this paper. Thanks are especially due to Jörg Siekmann, my advisor, for many inspiring discussions and constructive suggestions.

References

[Andrews 80] P.B.Andrews, Transforming Matings into Natural Deduction Proofs, Lecture Notes in Computer Science 87 (CADE 1980).

[Andrews 81] P. B. Andrews, Theorem Proving via General Matings, Journal of the ACM 28, 1981.

[Andrews 86] P. B. Andrews, An Introduction to Mathematical Logic and Type Theory: to Truth Through Proof, Academic Press, 1986

[Braine 78] M. D. S. Braine, On the Relation between the Natural Logic of Reasoning and Standard Logic, psychology review 1978.

[Benthem 88] J. v. Benthem, A Manual of Intensional Logic.CSLI 1988.

[Bibel 83/87] W. Bibel, Automated Theorem Proving, Vieweg, Braunschweig(1983/1987).

[Brachman 79] R. J. Brachman, On the Epistemological Status of Semantic Networks, in Associative Networks, Ed. N. V. Findler, Academic Press, 1979.

[Bundy 87] A. Bundy(1987), The Use of Explicit Plans to Guide Inductive Proofs. Dai Research Paper No. 349, Dept. of AI, Univ. of Edinburgh.

[Bundy et al 88] A. Bundy, F.van Harmelen, J. Hesketh, A. Smaill (1988), Experiments with Proof Plans for Induction. Dai Research Paper No. 413.

[Chester 76] D. Chester, The Translation of Formal Proofs into English. AI 7, 1976.

[Constable et al 86] R. L. Constable, et al, Implementing Mathematics with the Nuprl Proof Development System. Prentice Hall, Inc., 1986.

[Deussen 71] P. Deussen, Halbgruppen und Automaten. Springer-Verlag, 1971.

[Dowek 90] G. Dowek, A Proof Synthesis Algorithm for a Mathematical Vernacular, to appear in Proc. of the first Workshop on Logical Frameworks, Sophia-Antipolis, France, 1990.

[Felty & Miller 88] Amy Felty, Dale Miller, Specifying Theorem Provers in a Higher-Order Logic Programming Language, MS-CIS-88-12, LINC LAB 99, Univ. of Pennsylvania.

[Gabbay 84] D. Gabbay and F. Guenthner, Handbook of Philosophical Logic, D.Reidel Publishing Company, 1984.

[Genesereth 87] M. R. Genesereth & N. J. Nilsson, Logical Foundations of Artificial Intelligence, Morgan Kaufmann Publishers Inc, 1987.

[Gentzen 35] G. Gentzen, Untersuchungen über das logische Schließen I, Math. Zeitschrift 39, 1935.

[Gentzen 64] G. Gentzen, Investigation into Logical Deduction, American Philosophcal Quarterly, 1964, 1.

[Guttman et al 90] Joshua D. Guttman, William M. Farmer, F. Javier Thayer, IMPs: A Proof System for a Generic Logic, The MITRE Corporation, 1990.

[Hayes 79] Patrick J. Hayes, The Logic of Frames, in Fram Conceptions and Text Understanding, edited by D. Metzing, 1979.

[Hintikka 73] J. Hintikka, Logic, Language-games, and Information. Oxford: Oxford University Press, 1973.

[Holland et al 86] H. Holland, K. J. Holyoak, R. E. Nisbett, P. R. Thagard, Induction, Processes of Inference, Learning and Discovery. MIT Press, 1986.

[Huang 89.a] X.R. Huang, A Human Oriented Proof Presentation Model, SEKI Report SR-89-11.

[Huang 89.b] X.R. Huang, Proof Transformation Towards Human Reasoning Style, Proc. of GWAI-89.

[Huang 90] X.R. Huang, Reference Choices in Mathematical Proofs, to appear in Proc. of ECAI-90.

[Johnson-Laird 83] P.N. Johnson-Laird, Mental Models. Cambridge, Massachussetts: Harvard Univ. Press, 1983.

[Kerber 89.a] M. Kerber, Some Aspects of Analogy in Mathematical Reasoning. In: Jantke, K. (Ed.): *Proc. of Int. Workshop on Analogical and Inductive Inference*, Reinhardsbrunn, GDR. Springer LNAI 397. 231 - 242. Also available as SEKI REPORT SR-89-12

[Kerber 89.b] M. Kerber, A Frame Based approach to Representing Mathematical Concept. SEKI-Report SR-89-20.

[Lakoff 70] G. Lakoff, Linguistics and Natural Logic. Syntheses, 1970, 22.

[Lakoff 87] G. Lakoff, Women, Fire, and Dangerous Things, What Categories Reveal about the Mind, The University of Chicago Press, 1987.

[Lingenfelder 88] C. Lingenfelder, Structuring Computer Generated Proofs. SEKI-Report SR-88-19.

[Lingenfelder 89] C. Lingenfelder, Structuring Computer Generated Proofs. Proc.IJCAI-89

[Lingenfelder 90] Ch. Lingenfelder, Transformation and Structuring of Computer Generated Proofs. Ph.D. Thesis, University of Kaiserslautern, 1990.

[McDonald 83] D.D.McDonald, Natural Language Generation as a Computational Problem. In: Brady/Berwick: Computational Models of Discourse, MIT Press, Cambridge, MA, 1983.

[Miller 83]D. Miller, Proofs in Higher Order Logic, Ph.D Thesis, Carnegie Mellon University 1983.

[McKeown 85] K. R. McKeown, Text Generation. Cambridge Univ. Press, 1985.

[Paulson 89] Lawrence C. Paulson, The Foundation of a Generic Theorem Prover, Journal of Automated Reasoning 5, 1989.

[Pfenning 87]F. Pfenning, Proof Transformation in Higher-Order Logic, Ph.D. Theses, Carnegie Mellon University 1987.

[Pfenning & Nesmith90] F. Pfenning, D. Nesmith, Presenting Intuitive Deductions via Symmetric Simplification, Proc. CADE-90.

[Pfenning et al 90] F. Pfenning, Sunil Issar, Dan Nesmith Peter B. Andrews, ETPS User's Manual, CMU, 1990

[Prior 67] A. Prior, Past, Present and Future, Oxford University Press, 1967.

[Rescher 71] N. Rescher and A Urquhart, Temporal Logic, Sringer-Verlag, 1971.

[Rips 83] Lance J. Rips, Cognitive Processes in Propositional Reasoning, Psychological Review Vol. 90, 38-71, 1983.

[Robinson 65] J. A. Robinson, A Machine-Oriented Logic Based on the Resolution Principle, J.ACM 12, 1965.

[Schmidt-Schauß 89] M. Schmidt-Schauß, Computational Aspects of an Order-Sorted Logic with Term Declarations, Springer-Verlag, 1989.

[Stickel 86] M.E. Stickel, Schubert's Steam-roller Problem:Formulations and Solutions, J. of Automated Reasoning, Vol. 2, No. 1, 1986.