STATE UNIVERSITY OF NEW YORK AT BUFFALO

Department of Computer Science

A Belief Revision System Based on Relevance Logic and Heterarchical Contexts

João P. Martins & Stuart C. Shapiro

March 1981

Technical Report Number 175

This work was supported in part by the National Science Foundation under Grant MCS80-06314 and by the Instituto Nacional de Investigação Cientifica (Portugal) under Grant No. 20536.

Abstract

This paper describes the underlying theory of a Belief Revision System based on Relevance Logic and Heterarchical Contexts. In our system each statement is indexed by the set of basic (i.e., non-derived) assumptions used in its derivation and by the set of basic assumptions with which it is incompatible. A context is a set of basic assumptions and contains all the statements whose first index is a subset of the context and whose second index is disjoint from the context. This allows straightforward switching between contexts and the possibility of efficiently performing hypothetical reasoning.

1. Introduction

During the last 15 years researchers in Artificial Intelligence (AI) have worried about the problem of representing and manipulating knowledge in a computer program [Mylopoulos 81]. Most of this work focused in the epistemological side of the problem, i.e., in the structuring of knowledge, and not until recently have AI researchers started to work seriously on the problem of revising and modifying knowledge as new facts are known -- the so-called Belief Revision problem [Doyle and London 80].

Belief Revision is important in systems which have to make conclusions based on partial evidence and later on may have to revise such conclusions if an unexpected condition occurs, eg. a contradiction or the impossiblity of carrying out some action.

One of the most primitive forms of Belief Revision and also, until recently, the only one used in AI, is chronological backtracking, which consists in changing the most recent decision taken.

An alternative solution, the so called dependency-directed backtracking was proposed by [Stallman and Sussman 77]. They created a system in which data-dependencies are explicitly represented (they store, along with each derived fact, the set of all facts used in its derivation -- its dependency record). When a contradiction is found dependency records are traced backwards to find all the hypotheses assumed in the context of the contradiction and then heuristics are used to rule out one of them. In this way the assumption which is changed during backtracking does not necessarily correspond to the last choice made but rather to the assumption which <u>most likely</u> caused the contradiction to occur.

Based on the dependency-directed backtracking mechanism Doyle [Doyle 79; 80] designed a sytem for automatic Belief Revision (the Truth-Maintenance System - TMS). Doyle stores along with each fact in the system not only the set of all facts used in its derivation but also the set of all facts which are incompatible with it. When the presence of two incompatible facts is detected in the system TMS uses the dependency-directed backtracking to trace backwards the chain of dependency records and rule out one of the assumptions assumed in the context of the contradiction.

Doyle's work has been quite influential in the Belief

Revision domain and a number of systems were designed based on the TMS [Friedman 80; McAllester 78; Shrobe 79; Thompson 79].

All these systems assume that at any point there is one set of statements which is currently believed, which we will refer to as the current context, and these systems allow only one context to be represented at a time. This fact causes a big overhead when switching between contexts since the system has to search through a dependency graph marking and unmarking facts and therefore rules out the possibility of performing hypothetical reasoning efficiently.

Our system is an attempt to rectify this situation by allowing a flexible notion of context coupled with explicit record of data-dependencies and automatic revision of beliefs (truth maintenance). In our system each statement is indexed by the set of basic (i.e., non-derived) assumptions used in its derivation and by the set of assumptions with which it is incompatible. A context is a set of basic assumptions and contains all derived statements whose first index is a subset of the context and whose second index is disjoint from the context. This allows straigthforward switching between contexts and the possibility of efficiently performing hypothetical reasoning.

2. The SW system

We have worked out the underlying theory of a Belief Revision system based on relevance logic that is more flexible and efficient than similar systems built so far. The system will be built and tested within the paradigm of a Knowledge Engineering approach to Natural Language Understanding (NLU) [Shapiro and Neal 80] using the SNePS semantic network [Shapiro 79] and the MULTI system [McKay and Shapiro 80]. As a relevance logic formalism we use an expanded and modified version of the system proposed by [Shapiro and Wand 76] which is called the <u>SW system</u>, after <u>S</u>hapiro and <u>W</u>and.

A question that may be raised in the first place is what is the reason to base such a system on relevance logic? Relevance logic (refer to Appendix 1 for an introduction) relies on the fact that if "A implies B" then surely A and B must have something in common, i.e., A must be relevant to the deduction of B. One way to account for the relevance of well formed formulas (wffs) (used by [Anderson and Belnap 75] in the FR system) is to tag each wff with the set of hypotheses used in its derivation (dependencies of wffs are explicitly stored along with the wffs themselves) and to restrict the rules of inference so that only hypotheses really used to derive the wff are allowed in the tag. Since the problem of data-dependencies is of fundamental importance in Belief Revision systems we feel that relevance logic is better suited than ordinary logic to study these systems.

Each formula in the SW system is of the form <u>A,t,c,r</u> where <u>A</u> is some wff, t∈{hyp,der,ext}, and <u>c</u> and <u>r</u> are sets of <u>indices</u>. An index can be any object distinguishable from other indices. We refer to the quadruple A,t,c,r as an <u>assertion</u> or <u>formula</u>, to <u>A</u> as a <u>wff</u>, to <u>t</u> as the <u>origin tag</u>, to the set <u>c</u> as the <u>core context</u> (CC) of the wff A and to the set <u>r</u> as the <u>restriction set</u>

(RS) of the wff A. The purpose of t is to distinguish between the different kinds of formulas in the system: all assertions of the form A,hyp,c,r are hypotheses; all assertions of the form A,der,c,r are derived results, which have been deduced under the set of hypotheses $\{B,hyp,\{k\},r_B|k\in c\}$ and all assertions of the form A,ext,c,r are derived results whose CC has been extended (see below). Each hypothesis in the system is uniquely identified by an index. The sets of indices c and r in A,t,c,r identify those hypotheses which are related to the wff A: the set c identifies all the hypotheses under which the formula has been derived; the set r identifies all the hypotheses which are incompatible with the formula.

The program using the SW system interacts with a user that may either enter assertions or ask for certain kinds of reasoning to be performed. The reasoning of the system is justified by a set of rules of inference. Before we present some of these rules we will try to motivate the necessity for the origin tag and CC. Later in the paper we will give reasons and applications for the RS. The examples that support the discussion use Anderson and Belnap's FR system, which, for the purposes of the examples, can be viewed as a Fitch style system [Fitch 52] with CCs added to the wffs. In this system there is a hierarchical containment of subproofs which restricts the use one can make of a given formula, i.e., a formula derived in a subproof can not be used in another subproof not contained in the one in which the formula was derived. A new subproof is initiated every time a new hypothesis is introduced and every newly entered hypothesis is tagged by a unique index (corresponding in our system to the CC of an hypothesis -- see below). As new formulas are derived through the use of rules of inference their tag (CC) will identify the set of hypotheses used in their derivation.

One extremely important rule of inference in the FR system is the implication introduction (->I) which enables the derivation of implications: when in a subproof we have a wff, say B, which depends on the hypothesis which initiated the subproof, say A, we can derive the wff A->B in the subproof immediately containing the subproof under consideration. In our system there is no hierarchical containment of subproofs, any formula which has been derived or introduced as an hypothesis is a potential candidate to be used in the derivation of new formulas through the use of rules of inference, so, if we want to use the ->I rule, we need to have some way to know whether or not a given formula is an hypothesis. The way to solve this problem is to add something to the formula: the origin tag. Without the origin tag we would be able to derive strange results, for example A->B from A&B (Fig.1).

1	A&B,{1},{}	Hyp
2	A,{1},{}	&E,1
3	B,{1},{}	&E,1
4	A->B,{},{}	->I,2,3 ??

Figure 1
false "proof"

In the FR system the assumed goal, is to be able to obtain a formula with an empty CC (theorems in the FR system are wffs with empty CC) and so the policy in manipulating formulas in such system seems to be "decrease as much as you can the CCs of the formulas". The SW system was designed to be used in a NLU environment and not as a theorem prover. Under such assumption one may wonder whether it would be useful to increase the CC of formulas. To answer such question, let us consider the partial proof in the FR system presented in figure 2a. One of the

```
| P,{1}            Hyp
|-----------
| | Q,{2}          Hyp
| |---------
| | P,{1}          Reit
| | Q,{2}          Rep

        a
```

```
| P,{1}            Hyp
|------------
| | Q,{2}          Hyp
| |----------
| | P,{1}          Reit
| | Q,{2}          Rep
| | P&Q,{1,2}      &I ??
| Q->P&Q,{1}       ->I

        b
```

Figure 2
Reasoning in the FR system

reasons why the FR system does not allow the application of the &I rule between the formulas P,{1} and Q,{2} (in the FR system the &I rule requires the equality of the CCs of the wffs to be conjoined) is to prevent the derivation of the "irrelevant" implication Q->P&Q,{1} (Fig.2b). In the inner subproof, however, we should be able to say that P&Q holds under the CC {1,2}, provided that we don't use this fact to apply the ->I rule. Within the inner subproof "world" (or context) it is clear that P&Q,{1,2} holds but this is not valid outside that world. This latter case in which one wants to do further reasoning under some assumed world would certainly be useful in a NLU system. Therefore, in our system, we allow the conjunction of two formulas with different CCs but we will "mark" the resulting formula with a special tag: the <u>ext</u> tag. The purpose of the <u>ext</u> tag is to state that a given wff has had its context extended and therefore neither it nor the formulas derived from it are eligible for the application of the ->I rule. To justify this, consider again the proof presented in figure 2a. As we have seen, in this proof it is not possible to assert the conjunction P&Q,{1,2}. Now, suppose that we re-do the proof by changing the hypotheses as shown in figure 3. By introducing on each

```
| P & (Q->Q),{1}      Hyp
|--------------
| P,{1}               &E
| Q->Q,{1}            &E
| | Q & (P->P),{2}    Hyp
| |--------------
| | Q,{2}             &E
| | Q->Q,{1}          Reit
| | Q,{1,2}           ->E
| | P,{1}             Reit
| | P->P,{2}          &E
| | P,{1,2}           ->E
| | P&Q,{1,2}         &I
```

Figure 3
Partial proof in the FR system

hypothesis of the proof a term which is a theorem in the FR system (A->A) we are able to derive P&Q,{1,2} within the inner subproof. The ->I rule allow us to bring P&Q out of the inner subproof, resulting in the implication (Q & (P->P))->P&Q,{1}. But we are not able to derive Q->P&Q,{1} since this is not a relevant implication.

We present next the motivations for building a new system (SW) instead of using an already worked out formalism (FR). In the FR system there is a hierarchical containment of subproofs. Every new hypothesis which is introduced is within some set of subproofs and this set is rigid and inflexible -- it can not change unless the whole proof is re-done. In the SW system when a new hypothesis is introduced it is independent of all the other formulas already extant. As new formulas are being derived they are associated with a set of hypotheses, their CC, which can be calculated from the CCs of their parent formulas. Each formula depends only on the hypotheses defined by its CC. Having in mind that a context is a set of hypotheses we can easily see that, at any time, the set of contexts represented in the network is the power set of the set of hypotheses introduced and also that any formula is simultaneously in every context which is a superset of the formula's CC. This fact leads to a great flexibility in dealing with contexts since it permits straightforward switching between contexts (one has only to consider a different set of hypotheses) and ease of delineation of formulas within some context. Furthermore, the SW system allows non-monotonic reasoning which is not possible in the FR system and this is another enhancement introduced by our system.

2.1. Inference rules of SW

To make the SW rules of inference easier to state, we will use the function $\uparrow$ defined on origin tags as shown in Figure 4.

$\uparrow$	hyp	der	ext
hyp	der	der	ext
der	der	der	ext
ext	ext	ext	ext

Figure 4
Definition of $\uparrow$

We present next the rules of inference allowed in the SW system.

The statement of the rule follows an informal motivation for the rule.

1. Hypothesis (Hyp): This rule allows us to introduce at any stage any proposition we choose as an hypothesis. There are two different ways in which we can introduce an hypothesis corresponding, respectively, to the two parts of this rule: an hypothesis may be introduced assuming that it is compatible with all the other formulas in the system; or an hypothesis may be introduced assuming its incompatibility with a given set of formulas in the system, namely all the formulas derived from a pre-determined set of hypotheses. The rule of Hypothesis states that at any point we may add A,hyp,{k},{} to the set of existing formulas, provided that no formula of the form B,hyp,{k},r already exists (we will refer to <u>k</u> as the <u>index</u> of the wff A); under the same restriction as above we can add A,hyp,{k},{k_1,...,k_n} to the set of existing formulas where $k_1 \ldots k_n$ are indices of existing hypotheses, i.e., for each <u>i</u>, H_i,hyp,{k_i},r_i is a formula in the system;
2. Hypothesis updating (Hup): This rule tries to model a form of reasoning which is well known to humans. Suppose that someone is trying to prove some formula. He assumes a set of working hypotheses and starts applying rules of inference to generate new formulas. It might be the case that during this process one realizes that some additional hypothesis is needed and in this case there are two options available to the reasoner: he can introduce a brand new hypothesis (which corresponds to the rule of hypothesis above) or he can

realize that one of the hypotheses previously introduced was incomplete, i.e., he should have not only assumed what he did but that and something else. In this case one does not re-do the proof but rather says something like "well ... when I assumed A I really should have assumed A and B so let me just add B to the hypothesis A and go on with the proof". The rule of Hypothesis updating corresponds to the formalization of this second type of reasoning. It states that an hypothesis A,hyp,{k},r may be updated by replacing it by A&B,hyp,{k},r without invalidating any formula derived from the original hypothesis, provided that if B,hyp,$\{k_b\}$,r_b exists in the system $\{k,k_b\}\cap(r\cup r_b)=\emptyset$. For a proof of this claim refer to [Shapiro and Wand 76, pp.15-16];

3. Implication introduction (->I): This rule corresponds to the fact that if one assumes some hypothesis A and if under such assumption B can be derived then it is clear that "B follows from A". This means that if one wants to derive an implication the most natural way to do it is to take the antecedent(s) of the implication one wishes to prove as an extra hypothesis and aim to derive its consequent as a conclusion, if one succeeds he may take this as the proof of the original implication. In standard natural deduction systems [Fitch 52], where there is a containment of subproofs the ->I rule introduces only one hypothesis at a time as an antecedent of the implication, namely the hypothesis which initiated the the subproof in which the formula is contained. In our system there is no subproof containment and the ->I rule allows us to deduce that the conjunction of any number of the hypotheses used in the derivation of a given formula

implies it in the CC defined by the remainding hypotheses. The rule of ->I states that if B,t,c,r is a formula and $H=\{C,hyp,\{k\},r_C|k\in c\} = \{C_1,hyp,\{k_1\},r_1,...,C_n,hyp,\{k_n\},r_n\}$ and if $\{k_{j+1}..k_n\}\cap(r_1\cup..\cup r_j)=\emptyset$ then we can deduce $(C_1\&..\&C_j)\text{->}B,\uparrow(t,der),\{k_{j+1}..k_n\},r_1\cup..\cup r_j\cup r$, where $1\leq j\leq n$. Note that the rule corresponding to the standard ->I, if $A,hyp,\{k\},r_a$ and $B,der,c\cup\{k\},r_b$ are existing formulas and if $r_a\cap c=\emptyset$ then we can deduce $A\text{->}B,c,r_a\cup r_b$, is a particular case of this one;

4. Implication elimination (->E): This rule corresponds to the well known modus ponens rule of inference. It states that if A,t_1,c_1,r_1 and $A\text{->}B,t_2,c_2,r_2$ are existing formulas and $(c_1\cup c_2)\cap(r_1\cup r_2)=\emptyset$ then we can deduce $B,\uparrow(t_1,t_2),c_1\cup c_2,r_1\cup r_2$;
5. Negation introduction (~I): This rule corresponds to the idea behind the method of proof by reductio ad absurdum. In standard natural deduction systems where there is a containment of subproofs the rule of ~I enables the derivation of the negation of the hypothesis of a subproof in which a contradiction has been found in the subproof immediately containing it. Since in our system there is no subproof containment the rule of ~I allows the derivation of the negation of the conjunction of any number of the hypotheses assumed in the CC of a contradiction in the CC defined by the remaining hypotheses. Furthermore we allow, in the second part of this rule, an extended form of contradiction originated by contradictory wffs defined in two diferent CCs. The rule of ~I states that if A,t_1,c,r_a and $\sim A,t_2,c,r_b$ are both existing formulas and if

$H=\{C,hyp,\{k\},r \mid k \in c\} = \{C_1,hyp,\{k_1\},r_1,..,C_n,hyp,\{k_n\},r_n\}$ then we can deduce $\sim(C_1\&..\&C_j),\uparrow(t_1,t_2),\{k_{j+1}..k_n\},r_1\cup..\cup r_j\cup r_a\cup r_b$, where $1\leq j\leq n$. Notice that the rule corresponding to the standard ~I, if A,t_1,c,r_1 and $\sim A,t_2,c,r_2$ are both existing formulas then we can deduce $\sim C,\uparrow(t_1,t_2),c-\{k\},\{k\}\cup r_1\cup r_2$ for each $k\in c$ and $C,hyp,\{k\},r$, is a particular case of this rule. The second part of ~I states that if A,t_1,c_1,r_a and $\sim A,t_2,c_2,r_b$ are both existing formulas and if $c_1\neq c_2$ and $(c_1\cup c_2)\cap(r_1\cup r_2)=\emptyset$, then, if H is the set defined as above, with $c=c_1\cup c_2$, we can deduce $\sim(C_1\&..\&C_j),ext,\{k_{j+1},..,k_n\},r_1\cup..\cup r_j\cup r_a\cup r_b$, where $1\leq j\leq n$;

6. <u>Negation elimination</u> (~E): This rule corresponds to the well known modus tollens rule of inference. It states that if $\sim B,t_1,c_1,r_1$ and $A\text{->}B,t_2,c_2,r_2$ are existing formulas and $(c_1\cup c_2)\cap(r_1\cup r_2)=\emptyset$ then we can deduce $\sim A,\uparrow(t_1,t_2),c_1\cup c_2,r_1\cup r_2$;
7. <u>Hypotheses contradiction</u> (Hypcon): This rule is related to the rule of ~I. It handles the case where a contradiction is found in a context defined only by two hypotheses and it results in the explicit statement that the two hypotheses are mutualy exclusive: if $A,t_1,\{k_i,k_j\},r_1$ and $\sim A,t_2,\{k_i,k_j\},r_2$ (or if $A,t_1,\{k_i\},r_1$ and $\sim A,t_2,\{k_j\},r_2$) are both existing formulas and if $H_i,hyp,\{k_i\},r_i$ and $H_j,hyp,\{k_j\},r_j$ are the hypotheses in their CC then we can replace those hypotheses by $H_i,hyp,\{k_i\},r_i\cup\{k_j\}$ and $H_j,hyp,\{k_j\},r_j\cup\{k_i\}$;
8. <u>Double negation introduction</u> (~~I): This rule formalizes the idea that if one affirms that a given proposition holds it is the same as saying that it is not the case that it is not the case that the proposition holds. It states that if A,t,c,r is an existing formula then we can deduce $\sim\sim A,\uparrow(t,der),c,r$;

9. <u>Double negation elimination</u> (~~E): This rule formalizes the idea that affirming that it is not the case that it is not the case that some proposition holds is the same as saying that the proposition holds. It states that if ~~A,t,c,r is a formula then we can deduce A,↑(t,der),c,r;

10. <u>And introduction</u> (&I): This rule corresponds to the principle of reasoning that says that if some formula A holds and some other formula B also holds then it is clear that both of them hold. It states that if A,t_1,c_1,r_1 and B,t_2,c_1,r_2 are existing formulas then we can deduce $A\&B,\uparrow(t_1,t_2),c_1,r_1 \cup r_2$; if A,t_1,c_1,r_1 and B,t_2,c_2,r_2 are existing formulas and $c_1 \neq c_2$ and $(c_1 \cup c_2) \cap (r_1 \cup r_2) = \emptyset$ then we can deduce $A\&B,ext,c_1 \cup c_2,r_1 \cup r_2$;

11. <u>And elimination</u> (&E): This rule corresponds to the idea that if A&B holds then certainly A holds and B holds. It states that if A&B,t,c,r is an existing formula and $t \neq ext$ then we can either deduce A,↑(t,der),c,r or B,↑(t,der),c,r or both;

12. <u>Or introduction</u> (vI): This rule formalizes the idea that if some proposition holds then the proposition resulting from disjuncting any other proposition to it also holds. It states that if A,t,c,r is an existing formula then we can either deduce AvB,↑(t,der),c,r, or BvA,↑(t,der),c,r or both, where B is any wff, provided that if $B,hyp,\{k\},r_b$ exists in the system $(c \cup \{k\}) \cap (r \cup r_b) = \emptyset$.

13. <u>Or elimination</u> (vE): This rule enables us to use disjunctions to derive new formulas. Suppose that we have a disjunction <u>AvB</u> and we wish to derive a certain conclusion <u>C</u>. Suppose also that we were able to derive <u>C</u> from either

disjunct, i.e., we were able to derive that A->C and that B->C. Under such circumstances, the rule of vE allows the introduction of <u>C</u> as a formula in the system. It states that if AvB,t_1,c_1,r_1, A->C,t_2,c_2,r_2 and B->C,t_3,c_2,r_3 are existing formulas and $(c_1 \cup c_2) \cap (r_1 \cup r_2 \cup r_3) = \emptyset$ then we can deduce C,$\uparrow(t_1,\uparrow(t_2,t_3))$,$c_1 \cup c_2$,$r_1 \cup r_2 \cup r_3$;

As an example we show in figure 5 the proof of the formula (P->(Q->R)) -> ((P&Q)->R) in the SW system. Note that although

1	P->(Q->R),hyp,{1},{}	Hyp
2	P&Q,hyp,{2},{}	Hyp
3	P,der,{2},{}	&E,2
4	Q->R,der,{1,2},{}	->E,3,1
5	Q,der,{2},{}	&E,2
6	R,der,{1,2},{}	->E,5,4
7	(P&Q)->R,der,{1},{}	->I,2,6
8	(P->(Q->R)) -> ((P&Q)->R),der,{},{}	->I,1,7

Figure 5
proof of (P->(Q->R)) -> ((P&Q)->R)

theorem proving was not the goal of the design of our system it can be used to prove theorems.

Up to now our discussion has ignored the RSs and in the example presented all the formulas had empty RSs. RSs were introduced to allow one assumption to supersede a set of other assumptions. Suppose that we want the hypothesis H to supersede the hypotheses $H_1 \ldots H_n$ (H_1,hyp,{k_1},r_1, .., H_n,hyp,{k_n},r_n are formulas in the system). All we have to do is to introduce H as an hypothesis whose RS contains the indices $k_1 \ldots k_n$, i.e., the indices of the hypotheses it supersedes. The way the rules of inference are stated prevents the application of any rule among two formulas such that the CC of one has an index contained in the RS of the other. This means that introducing the hypothesis H with RS $\{k_1 \ldots k_n\}$ would prohibit the use of any formula derived from one (or more) of the hypotheses $H_1 \ldots H_n$ in a context containing the hypothesis H. This superseding aspect of the SW system is used when performing non-monotonic reasoning as showed in the next sections.

The SW system will be implemented in a fully indexed database (SNePS Semantic Network) and, following a suggestion of [Shapiro and Wand 76], each index will be its hypothesis formula. Thus, in the actual implementation of the SW system, the CC and RS of each wff will be a set of pointers to the hypotheses themselves. This makes extremelly easy the computation of which formulas are accessable within a given context and which are excluded.

Furthermore, after a suggestion of Gerald Sussman, each formula will also be indexed by the parent formulas and by the rule of inference used in its derivation. This will enable the recording of the story of the derivation of the formulas which may be particularly important during explanation (refer to section 3.4).

In this way we can look at the network of formulas as being composed by two different but superimposed graphs: one containing the basic dependencies of the formulas and the other the story of their derivation.

3. Applications of the SW system

3.1. Truth-Maintenance

Truth-Maintenance systems, in Doyle's sense, are systems which maintain a database of statements, each of them having some indication of whether it is believed or not-believed, and as new statements are added to the system it automatically revises its current set of beliefs in such a way that no inconsistent statements are simultaneously believed.

In our system, statements will not be explicitly marked as believed or not believed, since the belief or disbelief in a statement depends on the context which is being considered, but rather the statements will be indexed by their CC. This fact enables efficiently switching between contexts since the system does not have to search through a dependency graph marking and unmarking facts but it has rather to consider only the formulas indexed by the new context. The way contexts are defined and the way the system will be implemented also allows the quick retrieval of all the formulas in a given context.

In our system there are three different ways of handling contradictions (explained in detail in the following sections) which correspond to a backtracking mechanism which seems to be more comprehensive than the one proposed by Stallman and Sussman.

3.2. Hypothetical Reasoning

Hypothetical reasoning can be defined as the kind of reasoning made from one or more hypotheses whose truth value is unknown, doubtful or even known to be false. Based on these three possibilities for the truth value of the hypotheses, Rescher defines three different kinds of hypothetical reasoning:

> " ... hypothetical inference is reasoning which derives a conclusion from premises one or more of which is _problematic_ (of unknown truth-status) or _belief-contravening_ (negating some accepted belief and thus taken to be false) or outright _counterfactual_ (i.e., actually known to be false) ... " [Rescher 64, p.1], italics in the original.

Let us now look at the possible applications of these kinds of reasoning and how our system will handle them.

Problematic Reasoning is reasoning in which one (or more) of the premises have an unknown truth value, i.e., they are possibly true or possibly false with no definite view being held. Such type of reasoning is usefull in contingency-planning, i.e., look ahead reasoning with hypotheses that may or may not happen so that the reasoner can be prepared to handle possible future situations.

Belief-contravening reasoning is reasoning done from premisses which "conflict with accepted beliefs upon grounds that are inductive or probabilistic rather than logico-deductive" [Rescher 64, p.4]. The premises are "believed" to be false although there are no logical grounds yet to show their falsehood. This is the kind of reasoning used in the well known reductio ad absurdum proof method.

Counterfactual reasoning is reasoning made from premises which are known to be false. This is the kind of reasoning usually known as "reasoning for the sake of argument".

Our system deals in a similar way with the different kinds of hypothetical reasoning: when an hypothesis which originates hypothetical reasoning is introduced it is treated as a regular hypothesis and the inference system proceeds without worrying about it. All the formulas derived from this hypothesis include in their CC a reference to it. In the case of contingency planning and reasoning for the sake of argument the system does whatever inferences it wants to do and when the hypothesis which originated such kind of reasoning is droped all the formulas that were derived from it will be ignored by the system. The word ignored seems to be the most appropriate since those formulas will not be erased or marked as disbelieved, they simply will not be considered for the application of further rules of inference. If later on that hypothesis is raised again all the formulas which were derived from it are immediately available for further deductions. In the case of a proof by reductio ad absurdum, when a contradiction is found and the system is faced with the task of negating one of the hypothesis it will negate the hypothesis which originated the belief-contravening reasoning using the ~I.

3.3. Non-monotonic Reasoning

One of the properties of classical logic is the so called monotonicity property: let P and Q be sets of sentences and c be a single sentence, if $P \subseteq Q$ and if $P \vdash c$ then $Q \vdash c$. This means that if a statement is derivable from a set of statements it is also derivable from any set of statements containing the original set. This principle does not conform with the laws of human thought in which the learning of a new fact may invalidate several other facts previously believed to be true.

Several attempts have been made in AI to incorporate non-monotonic reasoning in a computer program [McCarthy 80; McDermott and Doyle 80; Reiter 80; Shapiro 79b; Winograd 80]. The SNePS deduction system allows two non-monotonic operators:

1. The non-derivable operator ($\nvdash$): $\nvdash$A means that A is not derivable in the current network;
2. The default operator (Δ): ΔA means "assume A is true unless it is provably false", ΔA is equivalent to ($\nvdash$ ~A)->A.

In our system non-monotonic reasoning can be implemented in the following way: Rules with the non-monotonic operators initiate subproofs with the non-monotonic assertion as an hypothesis and, as we know, all the statements derived from this hypothesis include in their CC a reference to it. If later on the non-monotonic assertion is proved to be false, i.e., if we can derive the assumption assumed to be non-derivable, then all the statements that depend on such assumption can be easily identified by its index. Furthermore by not considering the non-monotonic assumption as part of the current context all the facts derived from it will not be considered by the system.

3.4. Explanation Capabilities

As programs become bigger and more complex the task of debugging them becomes more and more difficult. The difficulty in debugging a program can be considerably decreased if the program can explain its line of reasoning, i.e., supply reasons for its inferences and actions, since in this case the programmer will be concerned with the debugging of reasoning rather than code. Besides being easier to debug, systems which explain their reasoning seem more convincing to the users. This has been one of the major reasons for the incorporation of explanation capabilities in MYCIN.

MYCIN [Davis et al. 77; Shortliffe 76; 80], PROSPECTOR [Duda et al. 77; 78] and TEIRESIAS [Davis 77] are examples of programs which explain their line of reasoning. They approach the problem of explanation in a quite different way from one another: MYCIN, which is based on a production system, records instances of execution of rules in a tree which is then read by the explanation system; PROSPECTOR stores its knowledge in a partitioned semantic network which is read by the explanation system; TEIRESIAS uses the concept of meta-level knowledge to read the production rules of the system and explain its behavior.

Explanation capabilities can easily be introduced in our system:

1. The fact that each formula is indexed with the set of hypothesis used in its derivation would enable the immediate justification of the assumptions which support the formula, i.e., based on the context of the formula the system can justify its belief or disbelief in some formula;
2. During the derivation of a formula the system will not only be able to explain the rules of inference that are being

used but also the hypotheses which are being introduced or withdrawn at every step;

3. After the derivation of a formula the system can trace down the graph containing the story of its derivation and thereby present to the user the complete proof of the derivation.

3.5. Estimate of the cost of a change

A system which has a complete record of data-dependencies may compute the effect that the change of some hypothesis will have in the system, i.e., when some change is proposed to the system it can find out all those statements which will be affected by the change and compute how costly the change will be. This feature, which was first proposed by Stallman and Sussman as a potential application of their system [Stallman and Sussman 77, p.165], can also be used to implement the principle of minimum mutilation stated by [Quine and Ullian 78]: when dealing with an inconsistency and faced with the problem of giving up some beliefs the principle of minimum mutilation consists of discarding those beliefs that make the smallest change in the current set of beliefs but get rid of the contradiction.

To see how this can be done in our system let us consider again the rule of negation introduction presented before: suppose that A,t_1,c,r_1 and $\sim A,t_2,c,r_2$ are both existing formulas (i.e., an inconsistency is detected in CC c) and that the set of hypotheses defining the CC c is $H=\{C \mid C,hyp,\{k\},r \ \& \ k \in c\} = \{C_1,\ldots,C_n\}$, then, unless we are willing to do hypothetical reasoning whithin the context c, we should, at least, invalidate one of the hypothesis $C_1 \ldots C_n$. But which one? The rule of ~I allow us to negate any of them, i.e., we can deduce $\sim C, \uparrow(t_1, t_2), c-\{k\}, \{k\} \cup r_1 \cup r_2$ for each $k \in c$ and $C, hyp, \{k\}, r$. If we don't have any good clue about which hypothesis to change we can use Quine's principle of minimum mutilation instead of picking up any of them at random. This is done in the following way: for each of the hypothesis in H compute the number of formulas in the system which depend on that hypothesis (this can be easily be done due to our way of indexing formulas) and invalidate the hypothesis which has the smallest number of formulas depending upon it. It is important to clarify what we mean by invalidate one of the hypothesis. Let us assume that C_i is the hypothesis in H (i.e., the formula $C_i, hyp, \{k_i\}, r_i$ exists in the system and $k_i \in c$) which has the smallest number of formulas depending upon it. By "invalidate C_i" we mean "deduce $\sim C_i, der, c-\{k_i\}, \{k_i\} \cup r_1 \cup r_2$ and consider $c-\{k_i\}$ as the working context".

3.6. Enhancement of the SNePS deduction system

The current implementation of the SNePS deduction system [McKay and Shapiro 80; Shapiro 77; 79; Shapiro and McKay 80] is able to handle deduction rules which may have any other deduction rules in consequent position but does not allow arbitrary deduction rules to be in antecedent position of other rules, i.e., the antecedents of rules currently allowed must be either simple propositions or formulae with ${}_n\mathfrak{X}_j^i$ or $\not\vdash$ as main connectives. To explain the reason for this restriction let us, very briefly, take a look of how the present deduction system works.

The SNePS deduction system works by creating a set of processes, each one of them with certain duties, that run under the MULTI multiprocessing system [McKay and Shapiro 80]. There are basically three different kinds of processes:

1. Pattern matchers (INFER and F-INFER): these processes match a given node (or network structure) against the current network;

2. Data-collectors (ANS-CATCH, TOPINF and TOPMOST-TOPINF): these processes receive answers and remember each answer received. Everytime they receive an answer not previously received the answer is sent to all the processes to which the data-collector reports, all the processes which are interested in such an answer;

3. Active connectives (CH-processes and IMPLY): each of these processes is responsible for a given rule node. When answers are received concerning the antecedents of the rule node they decide what to do with the consequents of the rule.

The SNePS deduction system can work in both the forward and backward directions. The same basic deduction procedure is used in both modes of inference: to use some rule of inference the system tries to find an instance of its antecedents, which is done both through a network match of the antecedents of the rule and through a check for other rules which may, in turn, enable the deduction of such antecedents. If such an instance of the antecedents of the rule is found (or derived) a message stating this fact is sent through a data-collector to the CH-process which corresponds to the active (or procedural) representation of the rule. Such a process will decide whether or not the data received is enough for the deduction of the consequents of the rule. In case of a deduction of the consequents it sends messages stating this fact to all processes which are intersted in such results.

Let us now see why this approach does not work in the case of a rule being in antecedent position of other rule. With such a goal in mind let us consider the rule (A->B)->C, where A, B and C are expressions which may contain variables, and let us suppose that we want to derive an instance of C. The fact that we have in the network an instance of A, an instance of B, or both would not help us in the derivation of C since the existence of such instances <u>does not</u> entail that A->B. The process of network matches fails to work in this case since we are not interested in A or B "per se" but we are rather intersted in whether <u>B follows from A</u>. To show this fact we would have to assume A and, based on that, show that B holds. In the SW system this would correspond to the introduction of A as an hypothesis and using only the allowed rules of inference derive B under such hypothesis, i.e., the context of B would contain the index of A.

The implementation of the proposed system would enable the SNePS deduction system to allow any kind of rule nodes either in antecedent or consequent position of rules.

4. Example of Belief Revision

In this section we show how our system would handle the example presented in page 252 of [Doyle 79] in which the system is facing the task of scheduling a meeting to be held preferably at 10 a.m., in either room 813 or 801. Part of the reasoning followed by the SW system is shown in Figure 6.

1.	(TIME=10 v ~TIME=10),hyp,{1},{}	Hyp
2.	TIME=10,hyp,{2},{3}	Hyp,Hypcon,2,3
3.	~TIME=10,hyp,{3},{2}	Hyp,Hypcon,2,3
4.	(ROOM=813 v ROOM=801),hyp,{4},{}	Hyp
5.	ROOM=813,hyp,{5},{}	Hyp
6.	ROOM=801,hyp,{6},{}	Hyp
7.	(TIME=10 & ROOM=813),ext,{2,5},{3}	&I,2,5
8.	~(TIME=10 & ROOM=813),hyp,{8},{}	Hyp
9.	~ROOM=813,ext,{2,8},{3,5}	~I,7,8
10.	(TIME=10 & ROOM=801),ext,{2,6},{3}	&I,2,6
11.	~ROOM=801,hyp,{11},{}	Hyp
12.	(ROOM=813 & ~TIME=10),ext,{3,5},{2}	&I,3,5

Figure 6
Reasoning of SW system

A person wants the system to suggest a time and room for a meeting. The person prefers 10:00 and mentions no other options. The system establishes the hypothesis of line 1 and two competing hypotheses to consider (lines 2 and 3). Since 10:00 is prefered the current context is now {1,2}. The person says that the room must be 813 or 801. The system again establishes three new hypotheses (lines 4,5 and 6). Picking 813 to try first, the context of interest is {1,2,4,5} and an answer formula is derived (line 7). However the person finds that room 813 is busy at 10:00 (line 8) and wants an answer taking this information into consideration. Since lines 7 and 8 are contradictory, the context {2,5,8} is untenable (and therefore so is the current context). Hypothesis 8 reflects the user's assertion and hypothesis 2 reflects the user's preference, so hypothesis 5 is droped from the current context and negated in context {2,8} (line 9). We still need a room and although no room has been derived for the current context, hypothesis 6 is not inconsistent with the current context ({1,2,4,8}), so the system expands the current context to {1,2,4,6,8} and suggests room 801 at 10:00 (line 10). Now, however, the person discovers that room 801 is not available (line 11). This contradicts formula 6 which is in the current context ({1,2,4,6,8,11}), but line 6 was neither a user assertion nor a user preference, so it can be droped from the current context. Also hypothesis 6 and 11 are updated to record this incompatibility (Fig.7). Neither room 813 nor room

6.	ROOM=801,hyp,{6},{11}	Hypcon,6,11
11.	~ROOM=801,hyp,{11},{6}	Hypcon,6,11

Figure 7
hypotheses updated

801 is possible in the current context of formulas {1,2,4,8,11} (see formulas 9 and 11), so in order to get a room to meet, we must change the context. Hypotheses 1,4,8 and 11 are all user assertions, so the system drops hypothesis 2. Neither time nor room has been derived in the context {1,4,8,11}, but if we

include hypothesis 5 (and there is no reason not to), we have formula 5 holding in context {1,4,5,8,11}. Now, contemplating the times, we already know that formula 2 is incompatible with the current context (since {2,5,8} was untenable), so we can conclude that we will meet in room 813 but not at 10:00 (Figure 6, line 12).

5. Concluding Remarks

The novelty of the SW system consists in the way the current set of beliefs (and disbeliefs) is defined and handled. Each formula in the SW system is indexed by the indices of the hypotheses used in its derivation and by the indices of the hypotheses with which it is incompatible.

Our approach to the definition of context and the implementation of CC represent one of the most crucial points in our system. By defining context as a set of hypotheses and indexing each formula with all the hypotheses used in its derivation we have a system which is flexible in two different ways:

1. Given any formula we can immediately compute the set of all hypotheses used it its derivation. All we have to do is to follow the arcs connecting the formula with the hypotheses in its context;
2. Given any context (set of hypotheses) we can find all the formulas in the context by following the arcs connecting the hypotheses with the formulas derived from them and checking

whether the RS of the formula is compatible with the context.

Our approach avoids searching through a dependency graph to find the hypotheses used in the derivation of some formula and the need to mark formulas as believed or not-believed.

The discussion presented here only focused in propositional calculus. We are working towards a predicate calculus implementation of the rules of inference using the non-standard connectives available in SNePS [Shapiro 79b].

6. References

1. Anderson A. and Belnap N., "The pure calculus of entailment", The Journal of Symbolic Logic, Vol.27, pp.19-52, 1962.
2. Anderson A. and Belnap N., "First degree entailments", Mathematische Annalen, Vol.149, pp.302-319, 1963.
3. Anderson A. and Belnap N., Entailment, Princeton University Press, 1975.
4. Davis R., "Interactive Transfer of Expertise: Acquisition of New Inference Rules", Proc. IJCAI-77, pp.321-328.
5. Davis R., Buchanan B. and Shortliffe E., "Production Rules as Representation for a Knowledge-Based Consultant Program", Artificial Intelligence, Vol.8,No.1, pp.15-45, 1977.
6. Doyle J., "A Truth Maintenance System", in Artificial Intelligence, Vol.12 No.3, 1979.
7. Doyle J., "A model for deliberation, action and introspection", Technical Report No.581, Massachussets Institute of Technology, AI Lab., 1980.
8. Doyle J. and London P., "A selected descriptor-indexed bibliography to the literature on Belief Revision", SIGART Newsletter 71, pp.7-23, 1980.

9. Duda R., Hart P., Nilsson N., Reboh R., Slocum J. and Sutherland G., "Development of a Computer-Based Consultant for Mineral Exploration", Annual Report, SRI International, 1977.

10. Duda R., Hart P., Nilsson N. and Sutherland G., "Semantic Network Representations in Rule-Based Inference Systems", in Pattern-Directed Inference Systems, Waterman and Hayes-Roth (eds.), Academic Press, 1978.

11. Fitch F., Symbolic Logic: an introduction, Ronald Press, 1952.

12. Friedman L., "Trouble-Shooting by Plausible Inference", Proc. First AAAI Conference, pp.292-294, 1980.

13. McAllester D., "A three-valued truth maintenance system", MIT AI-Lab, Technical Report Memo 473, 1978.

14. McCarthy J., "Circumscription - A Form of Non-Monotonic Reasoning", in Artificial Intelligence, Vol.13 No.1-2, pp.27-39, 1980.

15. McDermott D. and Doyle J., "Non-monotonic logic I", Artificial Intelligence, Vol.13 No.1-2, pp.41-72, 1980.

16. McKay D. and Shapiro S., "MULTI - A LISP Based Multiprocessing System", Proc. of the 1980 LISP Conference, pp.29-37, 1980.

17. Mylopoulos J., "An overview of Knowledge Representation", in Proc. of the Workshop on Data Abstraction, Databases and Conceptual Modelling, SIGART Newsletter, No.74, pp.5-21, January 1981.

18. Plaisted D., "An efficient relevance criterion for mechanical theorem pproving", Proc. First AAAI Conference, pp.79-83, 1980.

19. Quine W. and Ullian J., The Web of Belief, Random House, 2nd Edition, 1978.

20. Reiter R., "A logic for default reasoning", Artificial Intelligence, Vol.13 No.1-2, pp.81-132, 1980.

21. Rescher N., Hypothetical Reasoning, North-Holland, 1964.

22. Shapiro S., "Representing and Locating Deduction Rules in a Semantic Network", in Proc. Workshop on Pattern-Directed Inference Systems, SIGART Newsletter, No.63, pp.14-18, June 1977.

23. Shapiro S., "The SNePS semantic network processing system", in Associative Networks, N.V. Findler (ed.), Academic Press, 1979a.

24. Shapiro S., "Using Non-standard Connectives and Quantifiers

for Representing Deduction Rules in a Semantic Network", presented at "Current Aspects of AI research", a seminar held at the Electrotechnical Laboratory, Tokyo, August 27-28, 1979b.

25. Shapiro S. and McKay D., "Inference with Recursive Rules", Proc. First AAAI Conference, pp.151-153, 1980.

26. Shapiro S. and Neal J., "A Knowledge Engineering appproach to Natural Language Understanding", SNeRG Technical Note 2, Department of Computer Science, SUNY at Buffalo, 1980.

27. Shapiro S. and Wand M., "The relevance of relevance", Technical Report No.46, Computer Science Department, Indiana University, Indiana, 1976.

28. Shortliffe E., Computer-Based Medical Consultations: MYCIN, Elsevier, 1976.

29. Shortliffe E., "CONSULTATION SYSTEMS FOR PHYSICIANS: The Role of Artificial Intelligence Techniques", Proc. of the Third Bienal Conference of the CSCSI, pp.1-11., 1980.

30. Shrobe H., "Dependency-Directed Reasoning in the Analysis of Programs which Modify Complex Data Structures", in Proc. IJCAI-79, pp.829-835.

31. Stallman R. and Sussman G., "Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Circuit Analysis", Artificial Intelligence, Vol.9 No.2, pp.135-196, 1977.

32. Tarski A., Introduction to logic and to the methodology of deductive sciences, Oxford University Press, 3rd Edition, 1965.

33. Thompson A., "Network Truth-Maintenance for Deduction and Modelling", Proc. IJCAI-79, pp.877-879.

34. Winograd T., "Extended inference modes in reasoning by computer systems", Artificial Intelligence, Vol.13 No.1-2, pp.5-26, 1980.

APPENDIX 1: <u>Introduction to Relevance Logic</u>

One of the objectives of logic is to establish the precise meaning of everyday terms such as "or", "and", "not", "every", "some". The meaning of some of these terms is sometimes vague and cases exist where there is a conflict between the intuitive everyday usage of the term and the precise meaning assigned by logic. As Tarski says,

> " ... If we combine two sentences by the words "if...then..." we obtain a compound sentence which is denoted as an IMPLICATION or a CONDITIONAL SENTENCE ... considerable differences between the usages of implication in logic and everyday language manifest themselves. Again, in ordinary language, we tend to join two sentences by the words "if...then..." only when there is some connection between their forms and contents ... we often associate with this connection the conviction that the consequent follows necessarily from the antecedent ... they [the logicians] extended the usage of this phrase ["if...then..."], considering an implication a meaningful sentence even if no connection whatsoever exists between its two members and they made the truth or falsity of an implication depending exclusively upon the truth or falsity of the antecedent and consequent ... It has been objected that the logicians, on account of their employment of material implication, arrived at paradoxes and even plain nonsenses. This has resulted in an outcry for a reform of logic to the effect of bringing about a far

> reaching rapprochement between logic and ordinary language regarding the use of implication ... some logicians have undertaken attempts to reform the theory of implication ... they are anxious to find also a place for another concept of implication, for instance, of such a kind that the possibility of deducing the consequent from the antecedent constitutes a necessary condition for the truth of an implication ... "
> [Tarski 65, pp.23-28]

One of such attempts is the work of Anderson and Belnap [Anderson and Belnap 62; 63; 75] on the logic of relevant implication or relevance logic.

In relevance logic proofs are restricted to ensure the relevance of antecedent to consequent and to avoid the paradoxes of implication:

1. A -> (B v ~B) : anything implies something that is true;

2. (A & ~A) -> B : a false statement implies anything at all.

Relevance logic is based on the fact that if "A implies B" then surely A and B must have something in common, i.e., A must be relevant to the deduction of B.

In this appendix we present an introduction to one of the formalisms of relevance logic (system FR of [Anderson and Belnap 75]) and compare it with standard logic, using the natural deduction notation of [Fitch 52] (standard Fitch system FH).

On the Fitch system there are no axioms, only rules of inference. A proof is a nested set of subproofs. A subproof is a list of either formulas or subproofs. There is one outermost subproof, called categorical, and the remaining subproofs are called hypothetical. Theorems are formulas in the categorical subproof.

In the system FR each wff has a set associated with it. We will use lowercase letters to represent these sets. If A is a wff with associated set a we write A,a to represent it. It is standard to use natural numbers to represent elements of a. The last element of a categorical subproof will be a wff which will be thereby proved to be a theorem of the system. In FR the set associated with this wff will be the empty set.

We present below the rules of inference used in the FR system and show, in paralel, the corresponding rules in the FH system. We will group the rules according to the classification of [Anderson and Belnap 75, pp.346-348].

I. Structural Rules

1.Hypothesis (Hyp): This rule permits us to introduce an hypothesis at any stage of a proof. This hypothesis receives a unit class {k} of numerical subscripts, where {k} is a singleton set whose element k has never before appeared in the proof and initiates a new subproof. (Fig.A.1)

2.Repetition (Rep): Any formula may be repeated within a subproof maintaining the associated set. (Fig.A.2)

```
FH      | A     Hyp          FR       | A,{k}     Hyp
        |----                         |------
        |                             |
```

Figure 1
hypothesis

```
FH      | A                       FR      | A,a
        | .                               | .
        | .                               | .
        | A     Rep                       | A,a        Rep
        |                                 |
```

Figure 2
repetition

3.Reiteration (Reit): Any formula may be reiterated (repeated) into any subproof contained in the proof in which the formula is, maintaining the associated set. (Fig.A.3)

```
FH    | A                      FR    | A,a
      | .                            | .
      | .                            | .
      | |                            | |
      | |  .                         | |  .
      | |  .                         | |  .
      | |  A      Reit               | |  A,a       Reit
      | |                            | |
```

Figure 3
reiteration

II - Intensional rules (involving only the intensional connective '->')

4.Implication introduction (->I): If in a subproof with hypothesis A,{k} we derive B,a∪{k} then we can infer A->B,a in the proof containing the subproof. (Fig.A.4)

5.Implication elimination (->E): If in a subproof we have

```
FH   | .                            FR   | .
     | .                                 | .
     | | A                               | | A,{k}
     | |----                             | |--------
     | | .                               | |    .
     | | .                               | |    .
     | | B                               | | B,au{k}
     | A->B        ->I                   | A->B,a        ->I
     |                                   |
```

Figure 4
-> introduction

both A,a and A->B,b then we can infer B,aUb. (Fig.A.5)

```
FH  | A                      FR   | A,a
    | .                           | .
    | .                           | .
    | A->B                        | A->B,b
    | B          ->E              | B,aUb      ->E
    |                             |
```

Figure 5
-> elimination

III - <u>Mixed rules</u> (intensional and extensional connectives)

6. <u>Negation introduction</u> (~I): From an hypothesis, A,{k}, implying a contradiction, i.e., implying both B,aU{k} and ~B,aU{k}, infer ~A,a in the proof containing the contradictory subproof. (Fig.A.6)

7. <u>Negation elimination</u> (~E): from ~B,a and A->B,b infer ~A,aUb. (Fig.A.7)

8. <u>Or elimination</u> (vE): If in a subproof we have the following three things: the assertion AvB,a; a subproof with hypothesis A,{k} and C,bU{k} as one of its subformulas; a subproof with hypothesis B,{k+1} and C,bU{k+1} as one of its subformulas, then we can infer C,aUb in the subproof. (Fig.A.8)

```
FH    | .                          FR    | .
      | .                                | .
      | | A                              | | A,{k}
      | |----                            | |--------
      | | .                              | | .
      | | .                              | | .
      | | B                              | | B,a∪{k}
      | | .                              | | .
      | | .                              | | .
      | | ~B                             | | ~B,a∪{k}
      | ~A          ~I                   | ~A,a            ~I
      |                                  |
```

Figure 6
~ introduction

```
FH    | .                          FR    | .
      | .                                | .
      | A->B                             | A->B,b
      | .                                | .
      | .                                | .
      | ~B                               | ~B,a
      | ~A          ~E                   | ~A,a∪b          ~E
      |                                  |
```

Figure 7
~ elimination

```
FH    | .                          FR    | .
      | .                                | .
      | AvB                              | AvB,a
      | | A                              | | A,{k}
      | |----                            | |--------
      | | .                              | | .
      | | .                              | | .
      | | C                              | | C,b∪{k}
      |                                  |
      | | B                              | | B,{k+1}
      | |----                            | |--------
      | | .                              | | .
      | | .                              | | .
      | | C                              | | C,b∪{k+1}
      | C             vE                 | C,a∪b         vE
      |                                  |
```

Figure 8
v elimination

IV - <u>Extensional rules</u>

9. <u>Double negation elimination</u> (~~E): If in a subproof we have ~~A,a then we can infer A,a. (Fig.A.9)

```
FH      | .                       FR      | .
        | .                               | .
        | ~~A                             | ~~A,a
        | A        ~~E                    | A,a       ~~E
        |                                 |
```

Figure 9
~~ elimination

10. <u>Double negation introduction</u> (~~I): If in a subproof we have A,a then we can infer ~~A,a. (Fig.A.10)

```
FH      | .                       FR      | .
        | .                               | .
        | A                               | A,a
        | ~~A        ~~I                  | ~~A,a          ~~I
        |                                 |
```

Figure 10
~~ introduction

11. <u>And introduction</u> (&I): If in a subproof we have both A,a and B,a then we can infer A&B,a. (Fig.A.11)

```
FH      | .                       FR      | .
        | .                               | .
        | A                               | A,a
        | .                               | .
        | .                               | .
        | B                               | B,a
        | A&B        &I                   | A&B,a          &I
        |                                 |
```

Figure 11
& introduction

12. <u>And elimination</u> (&E): If in a subproof we have A&B,a then we can either infer A,a, B,a or both. (Fig.A.12)

```
FH    | .                        FR  | .
      | .                            | .
      | A&B                          | A&B,a
      | .                            | .
      | .                            | .
      | A           &E               | A,a       &E
      |                              |

      | .                            | .
      | .                            | .
      | A&B                          | A&B,a
      | .                            | .
      | .                            | .
      | B           &E               | B,a       &E
      |                              |
```

Figure 12
& elimination

13. <u>Or introduction</u> (vI): If in a subproof we have A,a then we can infer AvB,a or BvA,a or both, where B is any wff. (Fig.A.13)

```
FH    | A                        FR  | A,a
      | .                            | .
      | .                            | .
      | AvB         vI               | AvB,a     vI
      |                              |

      | A                            | A,a
      | .                            | .
      | .                            | .
      | BvA         vI               | BvA,a     vI
      |                              |
```

Figure 13
v introduction

As examples we present proofs in both systems. In figure

A.14 is presented the proof of (P->(Q->R)) -> ((P&Q)->R). In

```
FH    | P->(Q->R)             Hyp    FR   | P->(Q->R),{1}           Hyp
      |-----------                          |---------------
      | | P&Q                 Hyp           | | P&Q,{2}               Hyp
      | |--------                           | |------------
      | | P                   &E            | | P,{2}                 &E
      | | P->(Q->R)           Reit          | | P->(Q->R),{1}         Reit
      | | Q->R                ->E           | | Q->R,{1,2}            ->E
      | | Q                   &E            | | Q,{2}                 &E
      | | Q->R                Rep           | | Q->R,{1,2}            Rep
      | | R                   ->E           | | R,{1,2}               ->E
      | (P&Q)->R              ->I           | (P&Q)->R,{1}            ->I
      P->(Q->R)->P&Q->R       ->I           P->(Q->R)->P&Q->R,{}      ->I
```

Figure 14
proof of (P->(Q->R))->((P&Q)->R)

figure A.15 is presented the proof of P -> (Q->(P&Q)). The

```
FH    | P                     Hyp    FR   | P,{1}                   Hyp
      |------                               |--------
      | | Q                   Hyp           | | Q,{2}                 Hyp
      | |----                               | |------
      | | P                   Reit          | | P,{1}                 Reit
      | | Q                   Rep           | | Q,{2}                 Rep
      | | P&Q                 &I            | | ?
      | Q->(P&Q)              ->I
      P->(Q->(P&Q))           ->I
```

Figure 15
proof of P->(Q->(P&Q))

impossibility of continuing the last proof in the system FR indicates that something irrelevant is going on. In figure A.16 we show that the paradoxes of implication are derivable in the FH system but that comparable derivations can not be carried out in the FR system.

Let us note that, as can be seen in the previous examples, the set associated with each formula in the FR system indicates <u>all</u> the hypothesis actually used in deriving that formula. For

```
FH  | A                     Hyp        FR  | A,{1}                  Hyp
    |-----------                           |---------------
    | | ~(Bv~B)             Hyp            | | ~(Bv~B),{2}          Hyp
    | |---------                           | |-------------
    | | | B                 Hyp            | | | B,{3}              Hyp
    | | |------                            | | |-----------
    | | | Bv~B              vI             | | | Bv~B,{3}           vI
    | | | ~(Bv~B)           Reit           | | | ~(Bv~B),{2}        Reit
    | | ~B                  ~I             | | ?
    | | Bv~B                vI
    | | ~(Bv~B)             Rep
    | ~~(Bv~B)              ~I
    | Bv~B                  ~~E
    A->(Bv~B)               ->I
```

a. A->(Bv~B)

```
FH  | A&~A                  Hyp        FR  | A&~A,{1}               Hyp
    |---------                             |-----------
    | | ~B                  Hyp            | | ~B,{2}               Hyp
    | |------                              | |--------
    | | A&~A                Reit           | | A&~A,{1}             Reit
    | | A                   &E             | | A,{1}                &E
    | | ~A                  &E             | | ~A,{1}               &E
    | ~~B                   ~I             | ?
    | B                     ~~E
    (A&~A)->B               ->I
```

b. (A&~A)->B

Figure 16
paradoxes of implication

this reason, it is called the <u>context</u> of the formula.