A Model for Belief Revision

João P. Martins
Departamento de Engenharia Mecanica, Instituto Superior Técnico, 1000 Lisboa, Portugal

Stuart C. Shapiro
Department of Computer Science, State University of New York at Buffalo, Buffalo, NY 14260, U.S.A.

Recommended by Drew McDermott and Thomas Dean

ABSTRACT

It is generally recognized that the possibility of detecting contradictions and identifying their sources is an important feature of an intelligent system. Systems that are able to detect contradictions, identify their causes, or readjust their knowledge bases to remove the contradiction, called Belief Revision Systems, Truth Maintenance Systems, or Reason Maintenance Systems, have been studied by several researchers in Artificial Intelligence (AI).

In this paper, we present a logic suitable for supporting belief revision systems, discuss the properties that a belief revision system based on this logic will exhibit, and present a particular implementation of our model of a belief revision system.

The system we present differs from most of the systems developed so far in three respects: First, it is based on a logic that was developed to support belief revision systems. Second, it uses the rules of inference of the logic to automatically compute the dependencies among propositions rather than having to force the user to do this, as in many existing systems. Third, it was the first belief revision system whose implementation relies on the manipulation of sets of assumptions, not justifications.

1. Issues in Belief Revision

1.1. Introduction

Most computer programs constructed by researchers in AI maintain a model of their environment (external and/or internal), which is updated to reflect the perceived changes in the environment. This model is typically stored in a knowledge base, and the program draws inferences from the information in the knowledge base. All the inferences drawn are added to the knowledge base. One reason for model updating (and thus knowledge base updating) is the detection of *contradictory information*. In this case, the updating should be preceded by the decision about what proposition is the culprit for the contradiction, its removal from the knowledge base,[1] and the subsequent removal of every proposition that depends on the selected culprit.

The conventional approach to handling *contradictions* consists of blaming the contradiction on the most recent decision made (chronological backtracking). An alternative solution, dependency-directed backtracking, consists of changing, not the last choice made, but a choice that caused the unexpected condition to occur. This second approach, proposed by Stallman and Sussman [53], started a great deal of research in one area of AI, which became loosely called belief revision.[2]

Belief revision systems [8, 14, 26] are AI programs that deal with contradictions. They work with a knowledge base, performing reasoning from the propositions in the knowledge base, "filtering" those propositions so that only part of the knowledge base is perceived, namely, the propositions that are under consideration, called the set of *believed propositions*. When the belief revision system considers another one of these sets, we say that it *changes its beliefs*. Belief revision is an area of considerable interest, being both the subject of theoretical studies (e.g., [9, 18, 58]) and practical implementations (e.g., [4, 39, 41]).

Typically, a belief revision system explores alternatives, makes choices, explores the consequences of its choices, and compares results obtained when using different choices. If, during this process, a contradiction is detected, the belief revision system will revise the knowledge base, changing its beliefs in order to get rid of the contradiction.

There are several problems that researchers in belief revision have to address: the *inference* problem, which studies how do new beliefs follow from old ones; the *nonmonotonicity* problem, which studies the methods of recording that one belief depends on the absence of another; *dependency recording*, which concerns the study of the methods for recording that one belief depends on another one; *disbelief propagation*, which worries about how one fails to disbelieve all the consequences of a proposition that is disbelieved; and, finally, the *revision of beliefs*, which studies how to change beliefs in order to get rid of a contradiction.

No single system or researcher has addressed all these problems. In the remainder of this section, we will take a look at some of the issues involved in each one of these areas and discuss how they had been addressed by the following researchers: Doyle [11–13], McAllester [31–33], McDermott [34–35], and de Kleer [5–7, 10]. Most of the work on belief revision has been

[1] Or making it inaccessible to the program.

[2] The field of belief revision in AI is usually recognized to have been initiated by the work of Jon Doyle [11, 12], although a system that performs belief revision (in robot planning) was developed simultaneously by Philip London [23].

influenced by these researchers: Doyle, who started the area, led to the studies of algorithm improvement by Goodwin [16–18] and by Petrie [42] and several applications, for example, [51, 55]; McAllester's system was used in several applications, for example, [20, 43]; McDermott's work led to the development of the first commercial system with belief revision, DUCK [52]; de Kleer's work, which we believe was highly influenced by the work described in this paper (see also [25, 28–30]), was the starting point to the implementation for the KEE worlds [39, 40].

1.2. Inference

Belief revision systems have to keep a record of where each proposition in the knowledge base came from—the *support* of the proposition. This record is used both during the identification of the possible culprits for a contradiction and in the process of changing the system's beliefs. It would be desirable to put the responsibility of computing these dependencies on the system itself, so that as new beliefs are generated their dependency on old beliefs will be *automatically* computed.

This is a problem area that has been mostly ignored by researchers. The systems of Doyle, McDermott, and de Kleer do not address this issue at all: the inferences are made outside the system, which just passively records them. McAllester generates justifications for the truth values of the propositions in the knowledge base using axioms that define the rules of inference of each logical symbol. The justifications are recorded as clauses obtained by the application of the axioms.

1.3. Nonmonotonicity

In systems where we have to make decisions based on incomplete information, it is useful to be able to tell that one belief depends on the absence of another. If this latter becomes believed, then the former is disbelieved. This kind of behavior is called *nonmonotonic* (see, for example, [57]).

Nonmonotonicity was addressed both by Doyle and McDermott. The basic idea underlying their approaches is to record, along with each proposition in the knowledge base, both the set of propositions that have to be believed and the set of propositions that have to be disbelieved in order for the proposition to be believed.

1.4. Recording dependencies

There are two ways of recording the support of propositions, corresponding to justification-based and to assumption-based systems [5]. In *justification-based* systems, the support of each proposition contains the propositions that *directly* produced it. This approach was taken by Doyle, McAllester, and McDermott. In *assumption-based* systems, the support of each proposition contains the *hypotheses* (*non*derived propositions) that produced it. This approach was taken by us and de Kleer.

Let us consider the following example from [3, p. 197]: Suppose that the knowledge base contains the propositions:

$$\forall(x)\,[\mathrm{Man}(x) \rightarrow \mathrm{Person}(x)]\,,$$

$$\forall(x)\,[\mathrm{Person}(x) \rightarrow \mathrm{Human}(x)]\,,$$

$$\forall(x)\,[\mathrm{Human}(x) \rightarrow \mathrm{Person}(x)]\,.$$

Adding Man(Fred) to the knowledge base causes the derivation of Person(Fred), which, in turn, causes the derivation of Human(Fred). Furthermore, the addition of Human(Fred) to the knowledge base causes Person(Fred) to be rederived.

In justification-based systems, the support of each proposition contains the propositions that *directly* produced it. Under this approach, when Person(Fred) is derived from $\forall(x)\,[\mathrm{Man}(x) \rightarrow \mathrm{Person}(x)]$ and Man(Fred), its support will be {Man(Fred), $\forall(x)\,[\mathrm{Man}(x) \rightarrow \mathrm{Person}(x)]$}. Likewise, the support of Human(Fred) is {Person(Fred), $\forall(x)\,[\mathrm{Person}(x) \rightarrow \mathrm{Human}(x)]$}. Finally, when Person(Fred) is rederived, its support will be {Human(Fred), $\forall(x)\,[\mathrm{Human}(x) \rightarrow \mathrm{Person}(x)]$}. In Fig. 1, we represent the dependencies among these propositions. In this figure, a node pointed to by an arc labeled "do" (derivation origin) represents the support for the proposition at the tail end of the arc. The arcs labeled "pr" (for premisses) leaving that node point to the propositions that produced the proposition supported by the node. If there exists a path of arcs (alternately labeled do and pr) from the proposition A to the proposition B, then the proposition A depends on proposition B.

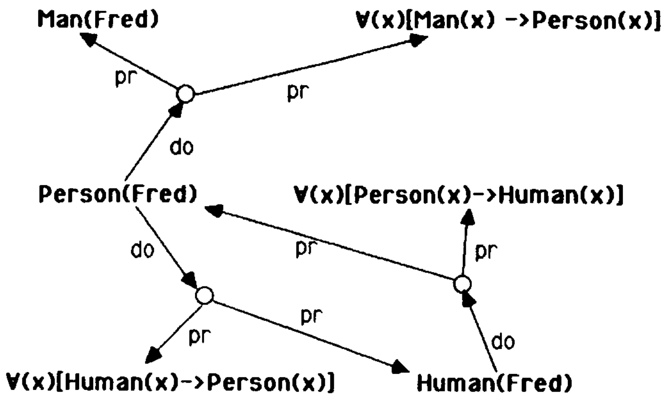

FIG. 1. Knowledge base dependencies (justification-based systems).

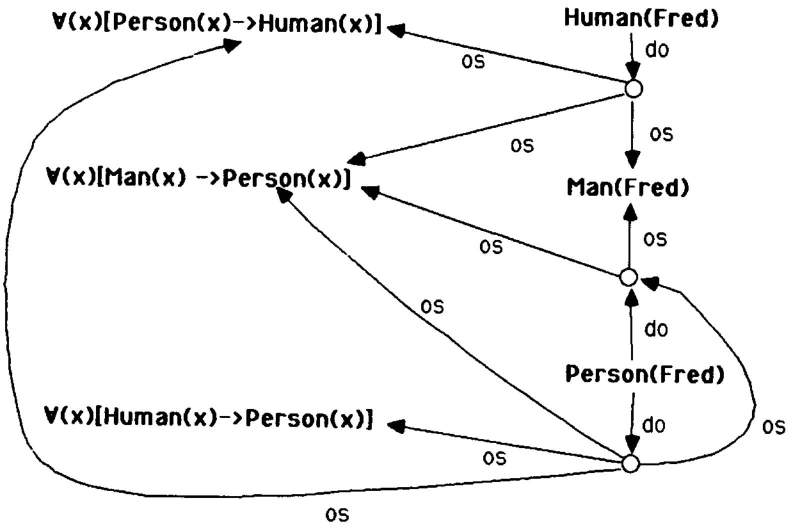

FIG. 2. Knowledge base dependencies (assumption-based systems).

In assumption-based systems, the support of each proposition contains the *hypotheses* (nonderived propositions) that produced it. Under this approach, when Person(Fred) is derived from ∀(x) [Man(x)→Person(x)] and Man(Fred), it is supported by these hypotheses;[3] i.e., its support is {Man(Fred), ∀(x) [Man(x)→Person(x)]}. When Human(Fred) is derived, it is supported by ∀(x) [Person(x)→Human(x)] and the hypotheses underlying Person(Fred); it is supported by {Man(Fred), ∀(x) [Man(x)→Person(x)], ∀(x) [Person(x)→Human(x)]}. Similarly, when Person(Fred) is re-derived it is supported by {Man(Fred), ∀(x) [Man(x)→Person(x)], ∀(x) [Person(x)→Human(x)], ∀(x) [Human(x)→Person(x)]}. Figure 2 shows the dependencies among the propositions in the knowledge base. In this figure, a node, pointed to by an arc labeled "do", represents the support for the proposition at the tail end of the arc. The arcs labeled "os" (origin set) leaving that node point to the hypotheses from which the proposition was derived.

1.5. Disbelief propagation

One important aspect of belief revision systems is the updating of the knowledge base when some proposition is disbelieved.[4] We should note that this

[3] We are assuming that the propositions Man(Fred), ∀(x) [Man(x)→Person(x)], ∀(x) [Person(x)→Human(x)], and ∀(x) [Human(x)→Person(x)] are hypotheses, i.e., they were entered into the knowledge base rather than being derived.

[4] Charniak, Riesbeck and McDermott [3] use the term "Knowledge Base Garbage Collection".

process should only be initiated by the disbelief of a hypothesis (a proposition that has no other justification than being told to the system by the user) rather than the disbelief of a derived proposition (which, without changing the underlying hypotheses, could be re-derived).

The result of this disbelieving process should be the set of propositions that had been generated if the system had started without the disbelieved proposition. Considering our previous example, let us assume that we decided that the proposition $\forall(x)$ [Person(x) $\rightarrow$ Human(x)] should be removed from the knowledge base. Clearly, this entails that Human(Fred), which was derived from it, should be removed from the knowledge base as well. This removal produces a new knowledge base in which the only propositions that should be considered are Man(Fred), $\forall(x)$ [Man(x) $\rightarrow$ Person(x)], $\forall(x)$ [Human(x) $\rightarrow$ Person(x)], and Person(Fred).

In a justification-based system, this is done by "marking" the propositions that should not be considered (marking propositions rather than erasing them permits some savings when a proposition once believed but later disbelieved is believed once more). When a proposition is removed from the knowledge base, the belief revision system has to go through the knowledge base deciding what the consequences of the removal are and "marking" propositions. A similar procedure has to take place if we decide to re-consider some proposition that is marked: the knowledge base has to be searched to decide which marked propositions should be "unmarked".

In assumption-based systems, the knowledge base retrieval function has to know which hypotheses are under consideration, whenever it performs a knowledge base retrieval operation. In assumption-based systems, there is no marking of the propositions in the knowledge base; it is the knowledge base retrieval function that decides dynamically[5] which propositions should be considered.

Concerning the implementation of the disbelieving process, researchers in belief revision have taken three different approaches:

(1) Doyle and McAllester perform two passes through the recorded proposition dependencies in the knowledge base: the first to disbelieve all the propositions that depend on the removed hypothesis, and the second to check whether each of the disbelieved propositions could be re-derived from the believed ones.

(2) McDermott uses both proposition dependencies and proposition labeling (data pools). The first is used to record the origin of the proposition, and the second to partition the propositions in the knowledge base. The algorithm for changing the labels on propositions (and thus place them in a different knowledge base partition) is very efficient and is used in disbelief propagation.

(3) De Kleer uses the notion of context, a set of hypotheses together with all

[5] Every time it performs a knowledge base retrieval.

the propositions derived from them, and the disbelief propagation is obtained by the removal of a hypothesis from the context.

1.6. The revision of beliefs

The revision of beliefs is the ultimate task for which a belief revision system is designed. It uses all the previously discussed features in deciding about the possible culprits for a contradiction, in "removing" one of them from the knowledge base, and in changing its beliefs accordingly.

No system has addressed the problem of selecting *the* culprit from the set of possible culprits for a contradiction, although some proposals have been made [2, 12, 25, 32].

The problems addressed here mainly concern the actual recording of the occurrence of the contradiction and the justification for disbelieving (or for believing) some proposition from its occurrence.

(1) Doyle creates a node recording the occurrence of a contradiction (a *contradiction* node, in Doyle's terminology) and justifies the belief or disbelief in a new proposition with this mode.

(2) McAllester uses clauses as the records of dependencies and infers new clauses to summarize a contradiction.

(3) De Kleer uses contradictions to tell that certain sets of hypotheses are inconsistent and that all propositions depending upon them are affected.

1.7. Overview of the paper

In this paper, we present a belief revision system based on a logic specifically conceived to support belief revision systems, discuss the properties of the system independently of its implementation (the abstract system), and present a particular implementation of our abstract model (SNeBR) using the SNePS Semantic Network Processing System [46].

SNeBR was the first assumption-based belief revision system implemented [25, 28, 29]; it is written in FRANZLISP and runs on VAX-11 systems at the Department of Computer Science, State University of New York at Buffalo, and at the Instituto Superior Técnico (School of Engineering of the Technical University of Lisbon, Portugal).

The paper is organized into several sections: Section 2 presents a logic, the SWM system, that relies on the notion of dependency and provides for dealing with contradictions; Section 3 extends SWM by the introduction of nonstandard connectives; Section 4 discusses the features that a computer program based on SWM will exhibit; Section 5 presents a particular implementation of our abstract model; and Section 6 presents an example illustrating some of the features of our system.

2. The SWM System

2.1. Introduction

In this section we introduce a logic, the SWM system,[6] that was developed to support belief revision systems. The interesting aspect of supporting a belief revision system in SWM is that the dependencies among propositions can be computed by the system itself rather than having to force the user to do this, as in many existing systems.

When discussing a logic, there are two aspects to consider, its syntax and its semantics. The *syntax* of a logic includes a set of formation rules and a set of rules of inference. The set of *formation rules* determines which formulas are legal in the logic. These formulas are called well-formed formulas, *wffs* for short. We assume standard formation rules for wffs with $\neg$, $\vee$, $\wedge$, $\rightarrow$ as connectives and $\forall$, $\exists$ as quantifiers. See, for example, [22, pp. 44 and 104]. The set of *rules of inference* (the deductive system) specifies which conclusions may be inferred from which premisses. Given an argument (P, c) (a premisse-conclusion argument is an ordered pair (P, c) in which P is a set of propositions, called *premisses*, and c is a single proposition, called the *conclusion*), we say that c is *deducible* from P, written $P \vdash c$, if there is a sequence of rules of inference which when applied to P produces c. The *semantics* of a logic concerns the study of the conditions under which sentences are true or false. The semantics are completely determined by the specification of two things, the *interpretations* of the language (every possible assignment of a particular object to each particular member of the language) and the *truth conditions* for it (what it means for a given sentence to have a given truth value in a given interpretation). We say that the argument (P, c) is *valid* if there is no interpretation in which each sentence in P is true and in which c is false. If (P, c) is valid, we write $P \models c$.

There is nothing about validity in the deductive system, and there is nothing about deducibility in the semantics. Although syntax and semantics are separate parts of a logical system, and thus deducibility and validity are intensionally distinct, they must fit together properly in order for the system to make any sense. A logic is said to be *sound* if and only if every argument deducible in its deductive system is valid according to its semantics. A logic is said to be *complete* if and only if every argument valid according to its semantics is deducible in its deductive system.

It is important to stress, right from the beginning, that SWM has a syntax but doesn't (yet) have a semantics. In other words, we have developed a proof technique that is suitable to support belief revision systems, but we have not formally studied the conditions under which sentences are true or false. We

[6] After Shapiro, Wand, and Martins. The SWM system is a successor of the system of Shapiro and Wand [50].

hope that this paper will generate interest towards defining a semantics for SWM.

The first step towards formally analyzing arguments consists of providing precise meaning for everyday terms like "and", "or", "if", "if . . . then . . .", "every", "some", etc. In the process of translating an informal argument into a formal one, some of the features of the informal argument are lost. As Haack says:[7]

> Some informal arguments are intuitively judged to be valid, others invalid. One then constructs a formal language in which the relevant structural features of those arguments can be schematically represented, and axioms/rules which allow the intuitively approved, and disallow the intuitively disapproved arguments . . . . However, if formal logic faithfully followed informal arguments in all their complexity and vagueness there would be little point in formalisation . . . but considerations of simplicity, precision and rigour may be expected to lead to discrepancies between informal arguments and their formal representations . . . . One should recognise, then, that a failure on the part of a formal system to represent all the knobs and bumps of the informal arguments it systematises is not necessarily objectionable. On the other hand, one must be wary of assuming that all adjustments are acceptable; one needs to ask whether the gains in simplicity and generality compensate for the discrepancy. [19, pp. 32–34].

The important point is to keep in the model those features that are of interest to the modeler. Therefore one should bear in mind which features of the informal arguments one wants to preserve in their formal counterparts. In our case, our main goal is to keep a record of propositional dependencies, and our approach builds a deductive system that blocks some unwanted deductions that are allowed in classical logic. The blocked deductions involve the introduction of irrelevancies.

2.2. Relevance logic

Here, we introduce the terminology used by Anderson and Belnap in one of their relevance logic systems and show how it is used to effectively block some of the results, obtainable in classical logic, which Anderson and Belnap consider to be irrelevant. Anderson and Belnap's relevance logic was taken as the starting point for developing the SWM system. The main features of

[7] This quote may be taken as an argument against formal systems (logic) in AI. However, we should bear in mind that models always simplify the phenomenon being modeled and therefore that some features are always lost during the modeling process. Notice that *any* programmed model is a formal model and thus subject to these considerations.

relevance logic used in SWM are the way it keeps track of which hypotheses were used in the derivation of a given wff and the way this is used to restrict the application of certain rules of inference.

Relevance logic was proposed by Anderson and Belnap [1], reacting against the lack of relevance in classical logic. Among other things, relevance logic challenges classical logic with respect to the classical concept of validity: Anderson and Belnap argue that if one proposition entails another, then there must be an element of causality that relevantly connects them, and, for that reason, they do not recognize as valid some of the arguments classified as valid by classical logic. In particular, they explicitly deny the so-called paradoxes of implication: $A \rightarrow (B \rightarrow A)$, anything implies a true proposition; and $(A \wedge \neg A) \rightarrow B$, a contradiction implies anything. To their (semantic) notion of entailment, there corresponds a (syntactic) notion of deducibility according to which B is deducible from A only if the derivation of B genuinely uses, and does not simply take a detour via, A.

We briefly describe how Anderson and Belnap define deducibility in a natural deduction system, the *FR system* [1, pp. 346–348]. A natural deduction system, e.g., [15], contains no axioms, only rules of inference. The rules of inference of a natural deduction system typically contain: (1) a rule of hypothesis, which enables one to get started without the need of axioms from which to begin, and (2) two rules of inference for each logical symbol, called the introduction and elimination rules. The introduction rule tells how to introduce an occurrence of the logical symbol (logical symbols are either logical connectives or quantifiers) and is written σI, "σ" being the logical symbol. The elimination rule tells how to eliminate an occurrence of the symbol and is written σE.

A proof is defined to be a nested set of subproofs. A subproof is a list of wffs and/or subproofs. Each wff is contained in a subproof. Subproofs are initiated every time a new hypothesis is introduced (which can be done at any point) and terminated when the hypothesis is discharged. There is one outermost subproof, called "categorical", in which no hypotheses are assumed, and the remaining subproofs are called "hypothetical". Theorems are wffs in the categorical subproof.

In FR, to ensure that B is deducible from A only if A is *used* in the derivation of B, Anderson and Belnap restrict the classical rules of natural deduction, as follows:

(1) Within a deduction, each wff is associated with a set containing references to all the hypotheses that were *really* used in its derivation. We call this set the *origin set* (OS), and we denote the fact that A is a wff with OS α by writing A,α.

(2) The rules of inference are stated taking OSs into account, blocking what are considered to be irrelevant applications of the rules, which are allowed in classical logic.

In the FR system, whenever a new hypothesis is introduced, it is associated with a singleton OS whose element is an identifier that never appeared before in the proof.[8] The rules of inference are stated so that all the wffs derived using a particular hypothesis will have its identifier in their OS. When a rule of inference is applied, the resulting wff is associated with an OS that is either the union of the OSs of the parent wffs, the OS of the parent wff(s), or the set difference of the OSs of the parent wffs. To give an idea of how the OSs can be formed, we will elaborate on two rules, $\rightarrow$I and $\wedge$I.

The rule of $\rightarrow$I (implication introduction) states that, if A is a hypothesis with OS $\{k\}$, B is a derived wff with OS $\alpha \cup \{k\}$ (meaning that A was genuinely used in the derivation of B), and they are both in the same subproof, then one can deduce $A \rightarrow B$ (in the subproof immediately containing the subproof initiated by the introduction of the hypothesis A) and associate this new wff with the OS α. This rule is schematically presented in Fig. 3.[9] Notice that $A \rightarrow B$ does not depend on hypothesis A. This is the reason for the set difference operation performed on the OS of B to obtain the OS of $A \rightarrow B$.

The rule of $\wedge$I (and-introduction) states that if A and B are wffs with the same OS, then one can deduce $A \wedge B$ and associate it with that OS. This rule, shown in Fig. 4, may seem too strongly stated, but it must be so in order to restrict the gratuitous introduction of irrelevancies. Suppose that $\wedge$I allowed the conjunction of wffs with different OSs, resulting in a wff whose OS was the union of the OSs of the parent wffs. Figure 5 shows how we could then introduce irrelevancies. The application of $\wedge$E to the wff in line 5, which resulted from such a use of $\wedge$I, allows the hypothesis of line 2 to be "smuggled into" the OS of A (line 6), thereby allowing the "proof" of $A \rightarrow (B \rightarrow A)$, one of the paradoxes of implication. The proof makes use of some rules of inference which have not been discussed, namely *reiteration* (Reit), *repetition* (Rep), *and-elimination* ($\wedge$E), and *modus ponens* (MP) (for a description of these rules refer to [25]).

m	│ $A,\{k\}$	Hyp
	│ ...	
n	│ $B,\alpha \cup \{k\}$	
	$A \rightarrow B,\alpha$	$\rightarrow \mathrm{I}(m, n)$

FIG. 3. FR system's $\rightarrow$I.

	...	
m	A,α	
	...	
n	B,α	
	$A \wedge B,\alpha$	$\wedge \mathrm{I}(m, n)$

FIG. 4. FR system's $\wedge$I.

[8] Relevance logic systems typically use natural numbers as elements of the OSs.

[9] The symbols m and n in this figure denote line numbers and $\rightarrow \mathrm{I}(m, n)$ means that the rule of $\rightarrow$I was applied to the formulas in the lines m and n.

1	$A, \{1\}$	Hyp
2	$B, \{2\}$	Hyp
3	$A, \{1\}$	Reit(1)
4	$B, \{2\}$	Rep(2)
5	$A \wedge B, \{1, 2\}$	$\wedge$I(3, 4)??
6	$A, \{1, 2\}$	$\wedge$E(5)
7	$B \rightarrow A, \{1\}$	$\rightarrow$I(2, 6)
8	$A \rightarrow (B \rightarrow A), \{\}$	MP(1, 7)

FIG. 5. "Proof" in the FR system.

2.3. Supported wffs

Before presenting the rules of inference of the SWM system, let us discuss what types of information we need in our logic.[10] During this discussion, we should bear in mind that SWM will not be used to prove theorems but will rather be involved in a sort of "perpetual proof" akin to AI systems: it will receive information and will be questioned by a user who wants to know whether a particular proposition is true or false under some set of assumptions. During this process, contradictions may be uncovered and their culprits should be identified.

One of the fundamental problems that any logic underlying a belief revision system has to address is how to keep track of and propagate propositional dependencies. This is important, because, in the event of the detection of a contradiction, we should be able to identify *exactly which* assumptions were used in the derivation of the contradictory propositions: we don't want to blame some assumption irrelevant to the occurrence of the contradiction as the culprit for the contradiction; and, when looking for the possible culprits for a contradiction, we don't want to leave out any assumption possibly responsible for the contradiction.

One way of doing this, used in the FR system and in the system of Shapiro and Wand [50], consists of associating each wff with an *origin set*, which references every hypothesis used in its derivation. Most of this mechanism was adopted in the SWM system.[11]

[10] A detailed discussion of these issues can be found in [25].

[11] Besides the dependency propagation mechanism, there is another advantage in using relevance logic to support belief revision systems. In classical logic, a contradiction implies anything; thus, in a belief revision system based on classical logic, whenever a contradiction is derived, it should be discarded immediately. In a relevance-logic-based belief revision system, we may allow the existence of a contradiction in the knowledge base without the danger of filling the knowledge base with unwanted deductions. In a relevance-logic-based belief revision system all a contradiction indicates is that any inference depending on *every* hypothesis underlying the contradiction is of no value. In this type of system we can perform reasoning in a knowledge base known to be inconsistent. A detailed discussion can be found in [25].

Another important issue in belief revision systems which will be reflected in our logic consists in the recording of the conditions under which contradictions may occur. This is important, because once we discover that a given set of hypotheses is inconsistent,[12] we may not want to consider it again, and even if we do want to consider it, we want to keep in mind that we are dealing with an inconsistent set. In the SWM system, contradictions are recorded by associating each wff (and its corresponding origin set) with a set, called the *restriction set*, that contains information about which sets unioned with the wff's origin set produce an inconsistent set. When new wffs are derived, their restriction sets are computed directly from the restriction sets of the parent wffs, and when contradictions are detected, all the wffs whose origin set references any of the contradictory hypotheses have their restriction set updated in order to record the newly discovered contradictory set.

In addition, for the proper application of some rules of inference, it is important to know whether a given wff was introduced as a hypothesis or was derived from other wffs. In order to do this, we associate with each wff an identifier, called the *origin tag*, that tells whether the wff is a hypothesis, a normally derived proposition, or a special proposition that if treated regularly would introduce irrelevancies into the knowledge base. Concerning this latter case, suppose, for example, that we have the hypotheses that "John is tall" and that "John is fat". Under these two hypotheses, the rule of and-introduction allows us to conclude that "John is tall and fat". As opposed to the FR system, SWM allows for this inference but the resulting proposition is marked as "special" (see the rule for and-introduction in Section 2.4). Notice that the proposition that "John is tall and fat" depends on the hypotheses that "John is tall" and that "John is fat". If this proposition weren't special, then we could (in SWM) apply the rule of and-elimination to conclude that "John is fat" depends on the hypotheses that "John is tall" and that "John is fat", and if, later on, a contradiction were discovered involving "John is fat", we could end up blaming the hypothesis that "John is tall" as a possible culprit.

SWM deals with objects called *supported wffs*. A supported wff consists of a wff and an associated triple, its *support*, containing an *origin tag* (OT), an *origin set* (OS), and a *restriction set* (RS). We write $\langle A, \tau, \alpha, \rho \rangle$ to denote that A is a wff with OT τ, OS α, and RS ρ, and we define the functions $\mathbf{wff}(\langle A, \tau, \alpha, \rho \rangle) = A$, $\mathbf{ot}(\langle A, \tau, \alpha, \rho \rangle) = \tau$, $\mathbf{os}(\langle A, \tau, \alpha, \rho \rangle) = \alpha$ and $\mathbf{rs}(\langle A, \tau, \alpha, \rho \rangle) = \rho$. Notice that the support is not part of the wff itself but rather associated with a *particular occurrence of the wff*. The set of all supported wffs is called the *knowledge base*.

The OS is a set of hypotheses. The OS of a supported wff contains those (and only those) hypotheses that were *actually used* in the derivation of that wff. The OTs range over the set {hyp, der, ext}: *hyp* identifies hypotheses, *der* identifies normally derived wffs within SWM, and *ext* identifies special wffs

[12] A set is *inconsistent* if a contradiction may be derived from it. A set is *consistent* just in case it is not inconsistent. We represent a contradiction by $\rightarrow\leftarrow$, thus H is inconsistent if $H \vdash \rightarrow\leftarrow$.

whose OS was extended.[13] An RS is a set of sets of wffs. A supported wff whose RS is $\{R_1, \ldots, R_n\}$ means that the hypotheses in its OS added to any of the sets $R_1, \ldots, R_n$ produce an *inconsistent set*. The RS of a supported wff will contain *every* set of hypotheses which unioned with the wff's OS will produce a set that is *known*[14] to be inconsistent. Our rules of inference guarantee that the information contained in the RS is carried over to the new wffs whenever a new proposition is derived. Furthermore, the rules of inference guarantee that RSs do not contain redundant information; i.e., given $\langle A, \tau, \alpha, \rho \rangle$, the following types of redundancy do not arise:

(1) There is no $r \in \rho$ such that $r \cap \alpha \neq \emptyset$.[15]

(2) There are no $r \in \rho$ and $s \in \rho$ $(r \neq s)$, such that $r \subset s$.[16]

We say that the supported wff $\langle A, \tau, \alpha, \rho \rangle$ has a *minimal* RS if the following two conditions are met:

(1) $\forall r \in \rho \; (r \cap \alpha) = \emptyset$;

(2) $(\forall r, s \in \rho \text{ and } (r \neq s)) \; r \not\subset s$.

In Appendix A, we prove that all the supported wffs in the knowledge base resulting from the application of the rules of inference of the SWM system have minimal RS.

To compute the RS of a supported wff resulting from the application of the rules of inference, we define the functions μ and $\int$.

The function μ is used whenever *a rule of inference that generates a supported wff whose OS is the union of the OSs of the parent wffs* is applied. It generates the RS of the resulting wff by unioning the RSs of the parent wffs and removing from the resulting set some sets which would be redundant, namely, those that would violate one of the two conditions listed above. The function μ is defined as follows:

$$\mu(\{r_1, \ldots, r_m\}, \{o_1, \ldots, o_n\}) = \sigma(\Psi(r_1 \cup \cdots \cup r_m, o_1 \cup \cdots \cup o_n)) ,$$

where

$$\Psi(R, O) = \{\alpha \mid (\alpha \in R \wedge \alpha \cap O = \emptyset) \vee$$
$$\exists(\beta)[\beta \in R \wedge \beta \cap O \neq \emptyset \wedge \alpha = \beta - O]\} ,$$

[13] A supported wff with "ext" OT has to be treated specially in order to avoid the introduction of irrelevancies.

[14] It is important to distinguish between a set *being* inconsistent and a set *being known to be* inconsistent. An inconsistent set is one from which a contradiction *can be* derived; a set known to be inconsistent is an inconsistent set from which a contradiction *has been* derived. The goal of adding RSs is to avoid reconsidering *known* inconsistent sets of hypotheses.

[15] Otherwise, the set "r" would contain extra information, namely, all the wffs in $r \cap \alpha$.

[16] Otherwise, the set "s" could be discarded from the restriction set without any loss of information. Since "r" belongs to the RS, we know that $\alpha \cup r \vdash \rightarrow\leftarrow$. Also, since any set containing an inconsistent set is itself inconsistent (Theorem 4, in Appendix A), we could infer that $\alpha \cup s$ is inconsistent, since $(\alpha \cup r) \subset (\alpha \cup s)$.

i.e., $\Psi(R, O)$ produces a set like R but with all the members disjoint from$O$; and

$$\sigma(R) = \{\alpha \mid \alpha \in R \wedge \neg\exists(\beta)(\beta \neq \alpha \wedge \beta \in R \wedge \beta \subset \alpha)\}\ ,$$

i.e., $\sigma(R)$ produces a set like R, none of whose members are supersets of other members. In other words, the function μ takes as arguments a set of restriction sets and their associated origin sets and produces a restriction set containing all the information of the original restriction set but satisfying the conditions of minimality.

The function $\int$ is used whenever a rule of inference *generates a supported wff with a smaller OS than the parent wffs.* It takes the RS of the several hypotheses in the resulting OS and computes a minimal RS from those RSs. The function $\int$ is defined as follows:[17]

$$\begin{aligned}\int(O) = \mu(&\{r \mid \exists(\boldsymbol{H})\,\mathbf{wff}(\boldsymbol{H}) \in O \wedge \mathbf{ot}(\boldsymbol{H}) = \text{hyp} \wedge \mathbf{rs}(\boldsymbol{H}) = r\}\ ,\\ &\{o \mid \exists(\boldsymbol{H})\,\mathbf{wff}(\boldsymbol{H}) \in O \wedge \mathbf{ot}(\boldsymbol{H}) = \text{hyp} \wedge \mathbf{os}(\boldsymbol{H}) = o\})\ ,\end{aligned}$$

i.e., $\int(O)$ directly computes the set of sets known to be inconsistent with O from the hypotheses in O.

To compute the OT of a supported wff resulting from the application of the rules of inference, we define the function Λ as follows:

$$\Lambda(\alpha, \beta) = \begin{cases} \text{ext}, & \text{if } \alpha = \text{ext or } \beta = \text{ext}\ ,\\ \text{der}, & \text{otherwise}\ ;\end{cases}$$

$$\Lambda(\alpha, \beta, \ldots, \gamma) = \Lambda(\alpha, \Lambda(\beta, \ldots, \gamma))\ .$$

Two supported wffs are said to be *combinable* by some rule of inference if the supported wff resulting from the application of the rule of inference has an OS that is not known to be inconsistent. We define the predicate "Combine", which decides the combinability of the supported wffs $\boldsymbol{A}$ and $\boldsymbol{B}$:

$$\text{Combine}(\boldsymbol{A}, \boldsymbol{B}) = \begin{cases} \text{false}, & \text{if } \exists r \in \mathbf{rs}(\boldsymbol{A}) \colon r \subset \mathbf{os}(\boldsymbol{B})\ ,\\ \text{false}, & \text{if } \exists r \in \mathbf{rs}(\boldsymbol{B}) \colon r \subset \mathbf{os}(\boldsymbol{A})\ ,\\ \text{true}, & \text{otherwise}\ .\end{cases}$$

The rules of inference of the SWM system, guarantee the following:

(1) The OS of a supported wff contains *every* hypothesis that was used in its derivation.

(2) The OS of a supported wff contains *only* the hypotheses that were used in its derivation.

[17] The $\int$ (integral sign) is used because this function "integrates" the information contained in the RSs of a set of hypotheses.

(3) The RS of a supported wff records *every* set *known* to be inconsistent with the wff's OS.

(4) The application of a rule of inference is blocked if the resulting supported wff would have an OS *known* to be inconsistent.

The OT and OS of a supported wff are related with a *particular derivation* of its wff whereas the RS reflects our current knowledge about how the hypotheses underlying this derivation relate to the other hypotheses in the knowledge base. Once a supported wff is generated, its OT and OS remain constant; however, its RS changes as the knowledge about all the propositions in the knowledge base does. It should be stressed that this statement does not mean that we fail to address the problem of multiple derivations of the same proposition. This is discussed in Section 5.

2.4. The inference rules

The following are the rules of inference of the SWM system.[18]

Hypothesis (Hyp). For any wff A and sets of wffs $R_1, \ldots, R_n$ ($n \geq 0$), such that $\forall r \in \{R_1, \ldots, R_n\}$: $r \cap \{A\} = \emptyset$ and $\forall r,s \in \{R_1, \ldots, R_n\}$ and $r \neq s$: $r \not\subset s$, we may add the supported wff $\langle A, \text{hyp}, \{A\}, \{R_1, \ldots, R_n\}\rangle$ to the knowledge base, provided that A has not already been introduced as a hypothesis. The rule of hypothesis allows us to add new information to the knowledge base. In general, the hypotheses added will have empty restriction sets, which will be changed as new contradictions are discovered. However, it may be the case that in the domain that we are modeling there are some known incompatibilities between propositions which we want to tell the system about. This can be done by the specification of hypotheses with nonempty RS.

Implication Introduction ($\rightarrow$I). From $\langle B, \text{der}, o, r\rangle$ and any hypothesis $H \in o$, infer $\langle H \rightarrow B, \text{der}, o - \{H\}, \int(o - \{H\})\rangle$. Notice that any hypothesis in the OS of B *was used* in its derivation, and thus it implies B under the assumption of the remaining hypotheses.

Modus Ponens—Implication Elimination (MP). From $\langle A, t_1, o_1, r_1\rangle$, $\langle A \rightarrow B, t_2, o_2, r_2\rangle$, and Combine($\langle A, t_1, o_1, r_1\rangle$, $\langle A \rightarrow B, t_2, o_2, r_2\rangle$), infer $\langle B, \mathcal{A}(t_1, t_2), o_1 \cup o_2, \mu(\{r_1, r_2\}, \{o_1, o_2\})\rangle$.

Modus Tollens—Implication Elimination (MT). From $\langle A \rightarrow B, t_1, o_1, r_1\rangle$, $\langle \neg B, t_2, o_2, r_2\rangle$, and Combine($\langle A \rightarrow B, t_1, o_1, r_1\rangle$, $\langle \neg B, t_2, o_2, r_2\rangle$), infer $\langle \neg A, \mathcal{A}(t_1, t_2), o_1 \cup o_2, \mu(\{r_1, r_2\}, \{o_1, o_2\})\rangle$.

[18] There is an extra connective in the SWM system, the *truth-functional or*, which will not be discussed in this paper. For a description of this connective, refer to [25, 29].

Negation Introduction ($\neg$I). From $\langle A, t_1, o, r\rangle$ $\langle \neg A, t_2, o, r\rangle$, and any set $\{H_1, \ldots, H_n\} \subset o$, infer $\langle \neg(H_1 \wedge \cdots \wedge H_n)$, $\mathcal{A}(t_1, t_2)$, $o - \{H_1, \ldots, H_n\}$, $\int(o - \{H_1, \ldots, H_n\})\rangle$.

From $\langle A, t_1, o_1, r_1\rangle$, $\langle \neg A, t_2, o_2, r_2\rangle$, $o_1 \neq o_2$, Combine($\langle A, t_1, o_1, r_1\rangle$, $\langle \neg A, t_2, o_2, r_2\rangle$), and any set $\{H_1, \ldots, H_n\} \subset (o_1 \cup o_2)$, infer $\langle \neg(H_1 \wedge \cdots \wedge H_n)$, ext, $(o_1 \cup o_2) - \{H_1, \ldots, H_n\}$, $\int((o_1 \cup o_2) - \{H_1, \ldots, H_n\})\rangle$. This rule states that from the hypotheses underlying a contradiction we can conclude that the conjunction of any number of them must be false under the assumption of the others.

Negation Elimination ($\neg$E). From $\langle \neg\neg A, t, o, r\rangle$, infer $\langle A, \mathcal{A}(t, t), o, r\rangle$.

Updating of Restriction Sets (URS). From $\langle A, t_1, o_1, r_1\rangle$, and $\langle \neg A, t_2, o_2, r_2\rangle$, we *must* replace each hypothesis $\langle H, \text{hyp}, \{H\}, R\rangle$ such that $H \in (o_1 \cup o_2)$ by $\langle H, \text{hyp}, \{H\}, \sigma(R \cup \{(o_1 \cup o_2) - \{H\}\})\rangle$. Furthermore, we *must* also replace every supported wff $\langle F, t, o, r\rangle$ ($t =$ der or $t =$ ext) such that $o \cap (o_1 \cup o_2) \neq \emptyset$ by $\langle F, t, o, \sigma(r \cup \{(o_1 \cup o_2) - o\})\rangle$. This is a special rule of inference. It is obligatorily applied whenever a contradiction is detected. Its effect is to take every supported wff in the knowledge base whose OS is not disjoint from the set just discovered to be inconsistent and update its restriction set accordingly. Note that in the implemented system it is not really necessary to change the entire knowledge base when this rule is triggered.

And-Introduction ($\wedge$I). From $\langle A, t_1, o, r\rangle$ and $\langle B, t_2, o, r\rangle$, infer $\langle A \wedge B, \mathcal{A}(t_1, t_2), o, r\rangle$.

From $\langle A, t_1, o_1, r_1\rangle$, $\langle B, t_2, o_2, r_2\rangle$, $o_1 \neq o_2$, and Combine($\langle A, t_1, o_1, r_1\rangle$, $\langle B, t_2, o_2, r_2\rangle$), infer $\langle A \wedge B, \text{ext}, o_1 \cup o_2, \mu(\{r_1, r_2\}, \{o_1, o_2\})\rangle$.

And-Elimination ($\wedge$E). From $\langle A \wedge B, t, o, r\rangle$, and $t \neq$ ext, infer either $\langle A, \text{der}, o, r\rangle$ or $\langle B, \text{der}, o, r\rangle$ or both. The rule of and-elimination is only applicable if the origin tag is not extended. This avoids the "smuggling" of hypotheses into the OSs.

Or-Introduction ($\vee$I). From $\langle \neg A \rightarrow B, t_1, o, r\rangle$ and $\langle \neg B \rightarrow A, t_2, o, r\rangle$, infer $\langle A \vee B, \mathcal{A}(t_1, t_2), o, r\rangle$.

Or-Elimination ($\vee$E). From $\langle A \vee B, t_1, o_1, r_1\rangle$, $\langle \neg A, t_2, o_2, r_2\rangle$, and Combine ($\langle A \vee B, t_1, o_1, r_1\rangle$, $\langle \neg A, t_2, o_2, r_2\rangle$), infer $\langle B, \mathcal{A}(t_1, t_2), o_1 \cup o_2, \mu(\{r_1, r_2\}, \{o_1, o_2\})\rangle$.

From $\langle A \vee B, t_1, o_1, r_1\rangle$, $\langle \neg B, t_2, o_2, r_2\rangle$, and Combine($\langle A \vee B, t_1, o_1, r_1\rangle$, $\langle \neg B, t_2, o_2, r_2\rangle$), infer $\langle A, \mathcal{A}(t_1, t_2), o_1 \cup o_2, \mu(\{r_1, r_2\}, \{o_1, o_2\})\rangle$.

From $\langle A \vee B, t_1, o_1, r_1 \rangle$, $\langle A \rightarrow C, t_2, o_2, r_2 \rangle$, $\langle B \rightarrow C, t_3, o_2, r_2 \rangle$, and Combine($\langle A \vee B, t_1, o_1, r_1 \rangle$, $\langle A \rightarrow C, t_2, o_2, r_2 \rangle$),[19] infer $\langle C, A(t_1, t_2, t_3), o_1 \cup o_2, \mu(\{r_1, r_2\}, \{o_1, o_2\}) \rangle$.

Universal Introduction ($\forall$I). From $\langle B(t), \text{der}, o \cup \{A(t)\}, r \rangle$, in which $A(t)$ is a hypothesis which uses a term (t) never used in the system prior to A's introduction, infer $\langle \forall(x)\,[A(x) \rightarrow B(x)], \text{der}, o, \int(o) \rangle$.[20]

Universal Elimination ($\forall$E). From the supported wffs $\langle \forall(x)\,[A(x) \rightarrow B(x)], t_1, o_1, r_1 \rangle$, $\langle A(c), t_2, o_2, r_2 \rangle$ and Combine($\langle \forall(x)\,[A(x) \rightarrow B(x)], t_1, o_1, r_1 \rangle$, $\langle A(c), t_2, o_2, r_2 \rangle$), in which "$c$" is any individual symbol, infer $\langle A(c) \rightarrow B(c), A(t_1, t_2), o_1 \cup o_2, \mu(\{r_1, r_2\}, \{o_1, o_2\}) \rangle$.

Existential Introduction ($\exists$I). From $\langle A(c), t, o, r \rangle$ in which "c" is an individual constant, infer $\langle \exists(x)\,[A(x)], A(t, t), o, r \rangle$.

Existential Elimination ($\exists$E). From $\langle \exists(x)\,[A(x)], t, o, r \rangle$ and any individual constant "c" that was never used before, infer $\langle A(c), A(t, t), o, r \rangle$.

Among others, the following theorems hold for SWM (their proofs can be found in Appendix A).

Theorem. *All the supported wffs in the knowledge base resulting from the application of the rules of inference of SWM have minimal RS* (Theorem 4).

Theorem. *In the knowledge base resulting from the application of the rules of inference of SWM, if two supported wffs have the same OS, then they have the same RS as well* (Theorem 5).

Corollary. *Every OS has recorded with it* every *known inconsistent set* (Corollary 5.1).

The SWM system, described in this section, was developed to support belief revision systems. Some of its features are recording dependencies of propositions, capturing the notions of causality and relevance, and providing for dealing with contradictions.

[19] Notice that $A \rightarrow C$ and $B \rightarrow C$ have the same OS and thus only one of them is needed in the predicate Combine.

[20] According to this rule of inference, the universal quantifier can only be introduced in the context of an implication. This is not a drawback, as may seem at first, since the role of the antecedent of the implication ($A(x)$) is to define the type of objects that are being quantified. This is sometimes called relativized quantification.

The SWM system is based on relevance logic and associates two sets with each derivation of a proposition: the *origin set* contains all the hypotheses that were used in the derivation of the proposition; the *restriction set* contains those sets of hypotheses which are known to be incompatible with the proposition's origin set.

There are some relationships between the OS and RS of supported wffs. In fact, each supported wff in the knowledge base resulting from the application of the rules of inference of SWM has a *minimal* RS in the sense that RSs are free from some kinds of redundancies. We can also say that each supported wff in the knowledge base has a *maximal* RS in the sense that its RS records *all* inconsistent sets *known* so far. We show in Appendix A that given any two supported wffs, if they have the same OS, then they have the same RS as well, reflecting the fact that RSs are both minimal and maximal.

As a final remark, we should stress that the benefits to an AI system of founding in a formal logic are a concise statement of the model and a set of assertions for validating the model and these benefits represent the reason why we developed SWM before beginning to work on the implementation.

3. Nonstandard Connectives

3.1. Introduction

In the previous section, we were concerned with the definition of a logic that would keep track of dependencies of propositions, would not allow the introduction of irrelevancies, and would be able to deal with contradictions. Here, we are concerned with using such a logic in practical applications. To do so, we extend SWM by adding nonstandard connectives (${}_n\mathbb{X}_i^j$, ${}_n\Theta_i$, $\vee\!\!\rightarrow$, $\wedge\!\!\rightarrow$). These nonstandard connectives [46, 47, 49] were motivated by knowledge representation issues and interest in carrying out deductions *within* the representation formalism, rather than *about* it. They were originally implemented in SNePS using a standard logic [46]; we shall refer to them as the *SNePS connectives*. We expand our set of formation rules with the addition of the following rules for creating wffs: if $A_1, \ldots, A_n$, $C_1, \ldots, C_m$ are wffs, then $(A_1, \ldots, A_n) \wedge\!\!\rightarrow (C_1, \ldots, C_m)$ and $(A_1, \ldots, A_n) \vee\!\!\rightarrow (C_1, \ldots, C_m)$ are wffs; if $A_1, \ldots, A_n$ are wffs, $i \geq 0$, $i \leq j$ and $j \leq n$, then ${}_n\mathbb{X}_i^j(A_1, \ldots, A_n)$ is a wff; if $A_1, \ldots, A_n$ are wffs and $0 \leq i \leq n$, then ${}_n\Theta_i(A_1, \ldots, A_n)$ is a wff.

The discussion in this section does not mean that we chose a "wrong" set of connectives when defining SWM. The connectives that were defined in Section 2 are simple and make the task of talking about the properties of the logic easier. However, they lack some expressive power, and this is the reason for the introduction of the SNePS connectives. A second point in favor of having defined standard connectives in SWM is that the SNePS connectives are defined in this section in terms of the standard ones, and therefore a wff using the SNePS connectives should be considered as a *syntactic abbreviation* of a complex wff using the standard connectives.

To make the rules of inference for the SNePS connectives easier to state, we introduce the following notation:

(1) Let $C_i^n(a_1, \ldots, a_n)$ represent an ordered set (using some ordering criteria that we don't care about) of all the i-combinations of the elements of the set $\{a_1, \ldots, a_n\}$. We represent the kth element of $C_i^n(a_1, \ldots, a_n)$ by ${}_kc_i^n(a_1, \ldots, a_n)$, and the set $\{a_1, \ldots, a_n\} - {}_kc_i^n(a_1, \ldots, a_n)$ by ${}_k\bar{c}_i^n(a_1, \ldots, a_n)$. For notational convenience, when the arguments of C_i^n, ${}_kc_i^n$, and ${}_k\bar{c}_i^n$ are left unspecified, they will default to $P_1, \ldots, P_n$. For example, $C_2^3(A, B, C) = \{\{A, B\}, \{A, C\}, \{B, C\}\}$, ${}_1c_2^3(A, B, C) = \{A, B\}$, ${}_2c_2^3(A, B, C) = \{A, C\}$, and ${}_2\bar{c}_2^3(A, B, C) = \{B\}$. Also, $C_2^3 = \{\{P_1, P_2\}, \{P_1, P_3\}, \{P_2, P_3\}\}$, and ${}_2\bar{c}_2^3 = \{P_2\}$.

(2) We write $\varphi\{P_1, \ldots, P_n\}$ to denote the function application $\varphi(P_1, \ldots, P_n)$, and we write $\varphi\langle\{P_1, \ldots, P_n\}\rangle$ to denote the set $\{\varphi(P_1), \ldots, \varphi(P_n)\}$. For example, $\wedge\{A_1, \ldots, A_n\} = A_1 \wedge \cdots \wedge A_n$; and $\neg\langle\{A_1, \ldots, A_n\}\rangle = \{\neg A_1, \ldots, \neg A_n\}$.

The function Combine is extended in the following way to decide the combinability of more than two supported wffs:

$$\text{Combine}(A_1, \ldots, A_n) = \begin{cases} \text{false}, & \text{if } \exists r \in \mu(\mathbf{rs}\langle\{A_1 \ldots A_n\}\rangle, \mathbf{os}\langle\{A_1 \ldots A_n\}\rangle): \\ & r \subset \cup[\mathbf{os}\langle\{A_1 \ldots A_n\}\rangle], \\ \text{true}, & \text{otherwise}. \end{cases}$$

3.2. And-entailment

And-entailment is a generalization of SWM's entailment to take two sets of arguments, a set of antecedents and a set of consequents. And-entailment, written $\wedge\!\!\rightarrow$, takes as arguments two sets of propositions. The proposition represented by the wff $(A_1, \ldots, A_n) \wedge\!\!\rightarrow (C_1, \ldots, C_m)$ asserts that the conjunction of the antecedents $(A_1, \ldots, A_n)$ implies the conjunction of the consequents $(C_1, \ldots, C_m)$.

Before presenting the introduction and elimination rules for and-entailment, we should say that, as opposed to SWM's entailment, there is only one rule for eliminating $\wedge\!\!\rightarrow$, corresponding to a generalization of MP. From our experience, and-entailment is most useful in the derivation of consequents from antecedents rather than otherwise, and we have decided not to generalize MT. However, such a generalization is trivial.

The introduction and elimination rules for $\wedge\!\!\rightarrow$ are formalized as follows:

And-Entailment Introduction ($\wedge\!\!\rightarrow$I). From $\langle(C_1 \wedge \cdots \wedge C_m), \text{der}, o \cup \{A_1, \ldots, A_n\}, r\rangle$, infer $\langle(A_1, \ldots, A_n) \wedge\!\!\rightarrow (C_1, \ldots, C_m), \text{der}, o, \int(o)\rangle$.

And-Entailment Elimination ($\wedge\!\!\rightarrow$E). From $\langle (A_1, \ldots, A_n) \wedge\!\!\rightarrow (C_1, \ldots, C_m), t, o, r\rangle$, $\langle A_1, t_1, o_1, r_1\rangle, \ldots, \langle A_n, t_n, o_n, r_n\rangle$, and Combine($\langle (A_1, \ldots, A_n) \wedge\!\!\rightarrow (C_1, \ldots, C_m), t, o, r\rangle, \langle A_1, t_1, o_1, r_1\rangle, \ldots, \langle A_n, t_n, o_n, r_n\rangle$), infer $\langle C_1 \wedge \cdots \wedge C_m, \mathcal{A}(t, t_1, \ldots, t_n), \cup\{O\}, \mu(R, O)\rangle$, where O and R are the sets $O = \{o, o_1, \ldots, o_n\}$ and $R = \{r, r_1, \ldots, r_n\}$.

3.3. Or-entailment

Or-entailment is another generalization of SWM's entailment to take a set of antecedents and a set of consequents. Or-entailment, written $\vee\!\!\rightarrow$, takes as arguments two sets of wffs. The proposition represented by the wff $(A_1, \ldots, A_n) \vee\!\!\rightarrow (C_1, \ldots, C_m)$ asserts that any antecedent implies the conjunction of the consequents. In other words, $(A_1, \ldots, A_n) \vee\!\!\rightarrow (C_1, \ldots, C_m)$ can be thought of as a shorthand for $A_1 \rightarrow (C_1 \wedge \cdots \wedge C_m) \wedge \cdots \wedge A_n \rightarrow (C_1 \wedge \cdots \wedge C_m)$.

The rules for or-entailment introduction and elimination are stated as follows:

Or-Entailment Introduction ($\vee\!\!\rightarrow$I). If for each k such that $1 \leq k \leq n$ we have $\langle (C_1 \wedge \cdots \wedge C_m)$, der, $o \cup \{A_k\}, r_k\rangle$,[21] infer $\langle (A_1, \ldots, A_n) \vee\!\!\rightarrow (C_1, \ldots, C_m)$, der, $o, \int(o)\rangle$.

Or-Entailment Elimination ($\vee\!\!\rightarrow$E). From $\langle (A_1, \ldots, A_n) \vee\!\!\rightarrow (C_1, \ldots, C_m), t_1, o_1, r_1\rangle$, $\langle A, t_2, o_2, r_2\rangle$, $A \in \{A_1, \ldots, A_n\}$ and Combine($\langle (A_1, \ldots, A_n) \vee\!\!\rightarrow (C_1, \ldots, C_m), t_1, o_1, r_1\rangle, \langle A, t_2, o_2, r_2\rangle$), infer $\langle C_1 \wedge \cdots \wedge C_m, \mathcal{A}(t_1, t_2), o_1 \cup o_2, \mu(\{o_1, o_2\}, \{r_1, r_2\})\rangle$.

3.4. And-or

And-or is a connective that generalizes $\neg$, $\wedge$, $\vee$, $\oplus$ (exclusive or), $|$ (nand), and $\downarrow$ (nor). And-or, written ${}_n\bowtie_i^j$, takes as arguments a set of n propositions. The proposition represented by the wff ${}_n\bowtie_i^j(P_1, \ldots, P_n)$ asserts that there is a connection between the propositions represented by the wffs $P_1, \ldots, P_n$ such that at least i and at most j of them must simultaneously be true. In other words, if $n - i$ arguments of ${}_n\bowtie_i^j$ are false, then the remaining i have to be true, and if j arguments of ${}_n\bowtie_i^j$ are true, then the remaining $n - j$ have to be false. In and-or, any argument can either be in consequent or antecedent position—i.e., used to supply information to deduce the other arguments, or being deduced from the information gathered about the other arguments.

[21] This means that for each such k, the entailment $\langle A_k \rightarrow (C_1 \wedge \cdots \wedge C_m)$, der, $o, \int(o)\rangle$ holds, and therefore from the n previous supported wffs we can assert $\langle (A_1 \rightarrow (C_1 \wedge \cdots \wedge C_m) \wedge \cdots \wedge A_n \rightarrow (C_1, \ldots, C_m))$, der, $o, \int(o)\rangle$.

We now define, without considering the OT, OS, and RS, what is needed to introduce and-or and what inferences can be drawn from it. Afterwards, we will express the introduction and elimination rules within the SWM formalism.

Suppose that we want to introduce ${}_n\bowtie_i^j(P_1, \ldots, P_n)$: the rule of $\bowtie$I requires the verification of the following two sets of conditions:

(1) If any $n-i$ arguments whatsoever are false, then all the others have to be true. This can be stated as: $\forall(k)$: $1 \leqslant k \leqslant \binom{n}{n-i}$: $\wedge[\neg\langle {}_k c^n_{n-i}\rangle] \rightarrow \wedge\,[{}_k \bar{c}^n_{n-i}]$.

(2) If any j arguments whatsoever are true, then all the others have to be false. This can be stated as: $\forall(k)$: $1 \leqslant k \leqslant \binom{n}{j}$: $\wedge[{}_k c^n_j] \rightarrow \wedge\,[\neg\langle {}_k \bar{c}^n_j\rangle]$.

Therefore, given n, i, and j, the $\bowtie$I rule requires the verification of the $\binom{n}{n-i}$ entailments listed under (1) and of the $\binom{n}{j}$ entailments listed under (2) above.

On the other hand, the following are the inferences allowed by $\bowtie$E:

(1) From ${}_n\bowtie_i^j(P_1, \ldots, P_n)$, $\neg A_1, \ldots,$ and $\neg A_{n-i}$, in which $\{A_1, \ldots, A_{n-i}\} \subset \{P_1, \ldots, P_n\}$, infer $\wedge[\{P_1, \ldots, P_n\} - \{A_1, \ldots, A_{n-i}\}]$.

(2) From ${}_n\bowtie_i^j(P_1, \ldots, P_n)$, $A_1, \ldots,$ and A_j, in which $\{A_1, \ldots, A_j\} \subset \{P_1, \ldots, P_n\}$, infer $\wedge[\neg\langle\{P_1, \ldots, P_n\} - \{A_1, \ldots, A_j\}\rangle]$.

Within SWM, the rules for introducing and eliminating and-or are stated as follows:

And-Or Introduction ($\bowtie$I). If for each k such that $1 \leqslant k \leqslant \binom{n}{n-i}$, we have $\langle A_1 \wedge \cdots \wedge A_i$, der, $o_k \cup \{\neg B_1, \ldots, \neg B_{n-i,}\}, r_k\rangle$, where $\{A_1, \ldots, A_i\} = {}_k\bar{c}^n_{n-i}$, and $\{B_1, \ldots, B_{n-i},\} = {}_k c^n_{n-i}$, meaning that $\langle(\neg B_1, \ldots, \neg B_{n-i}) \wedge\!\!\rightarrow (A_1, \ldots, A_i)$, der, $o_k, \int(o_k)\rangle$, and thereby that the $\binom{n}{n-i}$ entailments listed under (1) above are verified (we will refer to each of these supported wffs as $\langle F_k$, der, $O_k, R_k\rangle$); and also if for each q such that $1 \leqslant q \leqslant \binom{n}{j}$, we have $\langle\neg A_1 \wedge \cdots \wedge \neg A_{n-j}$, der, $o'_q \cup \{B_1, \ldots, B_j\}, r'_q\rangle$ in which $\{A_1, \ldots, A_{n-j}\} = {}_q\bar{c}^n_j$ and $\{B_1, \ldots, B_j\} = {}_q c^n_j$, meaning that $\langle(B_1, \ldots, B_j) \wedge\!\!\rightarrow (\neg A_1, \ldots, \neg A_{n-j})$, der, $o'_q, \int(o'_q)\rangle$ and thereby that the $\binom{n}{j}$ entailments listed under (2) above are verified (we will refer to each of these supported wffs as $\langle G_q$, der, $O'_q, R'_q\rangle$), then, letting $O = \{O_1, \ldots, O_{\binom{n}{n-i}}, O'_1, \ldots, O'_{\binom{n}{j}}\}$ and $R = \{R_1, \ldots, R_{\binom{n}{n-i}}, R'_1, \ldots, R'_{\binom{n}{j}}\}$:

(1) if $\forall\alpha,\beta \in O$, $\alpha = \beta$, infer $\langle {}_n\bowtie_i^j(P_1, \ldots, P_n)$, der, $O_1, R_1\rangle$;[22]

(2) if $\exists\alpha,\beta \in O$ such that $\alpha \neq \beta$ and also if Combine($\langle F_1$, der, $O_1, R_1\rangle, \ldots, \langle F_{\binom{n}{n-i}}$, der, $O_{\binom{n}{n-i}}, R_{\binom{n}{n-i}}\rangle$, $\langle G_1$, der, $O'_1, R'_1\rangle, \ldots, \langle G_{\binom{n}{j}}$, der, $O'_{\binom{n}{j}}, R'_{\binom{n}{j}}\rangle$), infer $\langle {}_n\bowtie_i^j(P_1, \ldots, P_n)$, ext, $\cup\{O\}, \mu(R, O)\rangle$.

And-Or Elimination ($\bowtie$E). From $\langle {}_n\bowtie_i^j(P_1, \ldots, P_n),\ t, o, r\rangle$, $\langle\neg A_1, t_1, o_1, r_1\rangle, \ldots, \langle\neg A_{n-i}, t_{n-i}, o_{n-i}, r_{n-i}\rangle$, $\{A_1, \ldots, A_{n-i}\} \subset \{P_1, \ldots, P_n\}$, and Combine($\langle {}_n\bowtie_i^j(P_1, \ldots, P_n), t, o, r\rangle$, $\langle\neg A_1, t_1, o_1, r_1\rangle, \ldots, \langle\neg A_{n-i},$

[22] Notice that since all the O are equal, then all R are equal as well (Theorem 5 in Appendix A) and for that reason, it doesn't matter which O or R we write down in the final wff.

$t_{n-i}, o_{n-i}, r_{n-i}\rangle)$, then, denoting by $\{B_1, \ldots, B_i\}$ the set $\{P_1, \ldots, P_n\} - \{A_1, \ldots, A_{n-i}\}$ and by O and R the sets $O = \{o, o_1, \ldots, o_{n-i}\}$ and $R = \{r, r_1, \ldots, r_{n-i}\}$, infer $\langle B_1 \wedge \cdots \wedge B_i,\ A(t, t_1, \ldots, t_{n-i}),\ \cup\{O\},\ \mu(R, O)\rangle$.

From $\langle {}_n\!\bowtie_i^j(P_1, \ldots, P_n), t, o, r\rangle$, $\langle A_1, t_1, o_1, r_1\rangle, \ldots, \langle A_j, t_j, o_j, r_j\rangle$, $\{A_1, \ldots, A_j\} \subset \{P_1, \ldots, P_n\}$ and Combine$(\langle {}_n\!\bowtie_i^j(P_1, \ldots, P_n), t, o, r\rangle, \langle A_1, t_1, o_1, r_1\rangle, \ldots, \langle A_j, t_j, o_j, r_j\rangle)$, then, denoting by $\{B_1, \ldots, B_{n-j}\}$ the set $\{P_1, \ldots, P_n\} - \{A_1, \ldots, A_j\}$ and by O and R the sets $O = \{o, o_1, \ldots, o_j\}$ and $R = \{r, r_1, \ldots, r_j\}$, infer $\langle \neg B_1 \wedge \cdots \wedge \neg B_{n-j},\ A(t, t_1, \ldots, t_j),\ \cup\{O\},\ \mu(R, O)\rangle$.

As a last point in the discussion of $\bowtie$I, let us consider the cases in which $i = 0$, $j = n$, $i = j = 0$, and $i = j = n$. When trying to introduce $\bowtie$ in any of these cases, we are faced with entailments that have either empty antecedent or empty consequent. When this happens, the proof of the entailments with empty antecedent or consequent should be disregarded and only the other set of entailments should be considered. Notice that we are not saying anything about how to introduce entailments with empty consequent or empty antecedent; in fact, such formulas are not wffs, and thus we don't have to worry about them. What we are saying here is that for some values of i and j, the $\bowtie$ connective becomes somewhat simplified, and thus not so much work is required in its introduction.

3.5. Thresh

Thresh generalizes equivalence to a set of arguments. Thresh, written ${}_n\Theta_i$, takes as arguments a set of n propositions. The proposition represented by the wff ${}_n\Theta_i(P_1, \ldots, P_n)$ asserts that there is a connection between the propositions represented by the wffs $P_1, \ldots, P_n$ such that either fewer than i of them are true or they all are true. If at least i of the arguments of ${}_n\Theta_i$ are true, then all the remaining arguments have to be true and if $i - 1$ arguments of ${}_n\Theta_i$ are true and at least one is false, then the remaining arguments have to be false. As in the last section, these inferences will be the guidelines for the ΘI rule and will be explicitly stated by the ΘE rule.

The introduction of ${}_n\Theta_i(P_1, \ldots, P_n)$ requires the verification of the following two sets of conditions:

(1) $\forall(k)\colon 1 \leq k \leq \binom{n}{i}\colon \wedge[{}_kc_i^n] \rightarrow \wedge\,[{}_k\bar{c}_i^n]$.

(2) $\forall(k)\colon 1 \leq k \leq \binom{n}{i}\colon \forall P \in {}_kc_i^n\colon \wedge[({}_kc_i^n - \{P\}) \cup \{\neg P\}] \rightarrow \wedge\,[\neg\langle {}_k\bar{c}_i^n\rangle]$.

Given n and i, the ΘI rule requires the verification of $\binom{n}{i}$ conditions of type (1) and $i \times \binom{n}{i}$ conditions of type (2) (there are $\binom{n}{i}$ different ways of choosing the antecedents of the entailment, and for each of those ways there are i ways of choosing the argument that is negated).

The conditions for the elimination rule are obtained directly from the conditions of the ΘI rule:

(1) From ${}_n\Theta_i(P_1, \ldots, P_n)$, $A_1, \ldots, A_i$, and $\{A_1, \ldots, A_i\} \subset \{P_1, \ldots, P_n\}$, infer $B_1 \wedge \cdots \wedge B_{n-i}$, where $\{B_1, \ldots, B_{n-i}\} = \{P_1, \ldots, P_n\} - \{A_1, \ldots, A_i\}$.

(2) From ${}_n\Theta_i(P_1, \ldots, P_n)$, $A_1, \ldots, A_{i-1}$, and $\neg A_i$, where $\{A_i\} \cap \{A_1, \ldots, A_{i-1}\} = \emptyset$, and $\{A_1, \ldots, A_{i-1}, A_i\} \subset \{P_1, \ldots, P_n\}$, infer $\neg B_1 \wedge \cdots \wedge \neg B_{n-i}$, where $\{B_1, \ldots, B_{n-i}\} = \{P_1, \ldots, P_n\} - \{A_1, \ldots, A_i\}$.

Within SWM, the rules for thresh introduction and elimination can be stated as follows:

Thresh Introduction (ΘI). If for each k such that $1 \leq k \leq \binom{n}{i}$ we have $\langle A_1 \wedge \cdots \wedge A_{n-i}$, der, $o_k \cup \{B_1, \ldots, B_i\}, r_k\rangle$, where $\{A_1, \ldots, A_{n-i}\} = {}_k\bar{c}_i^n$, and $\{B_1, \ldots, B_i\} = {}_kc_i^n$, meaning that $\langle (B_1, \ldots, B_i) \wedge\!\!\rightarrow (A_1, \ldots, A_{n-i})$, der, o_k, $\int(o_k)\rangle$ (which we will refer to as $\langle F_k$, der, O_k, $R_k\rangle$),[23] and also if for each q such that $1 \leq q \leq \binom{n}{i}$ and for each E such that $E \in {}_qc_i^n$ we have $\langle \neg A_1 \wedge \cdots \wedge \neg A_{n-i}$, der, $o'_q \cup ((\{B_1, \ldots, B_i\} - \{E\}) \cup \{\neg E\}), r'_q\rangle$, where $\{A_1, \ldots, A_{n-i}\} = {}_q\bar{c}_i^n$, and $\{B_1, \ldots, B_i\} = {}_qc_i^n$, meaning that $\langle((\{B_1, \ldots, B_i\} - \{E\}) \cup \{\neg E\}) \wedge\!\!\rightarrow (\neg A_1, \ldots, \neg A_{n-i})$, der, o'_q, $\int(o'_q)\rangle$ we will refer to these supported wffs as $\langle G_s$, der, O'_s, $R'_s\rangle$;[24] then, letting $O = \{O_1, \ldots, O_{\binom{n}{i}}, O'_1, \ldots, O'_{i\times\binom{n}{i}}\}$ and $R = \{R_1, \ldots, R_{\binom{n}{i}}, R'_1, \ldots, R'_{i\times\binom{n}{i}}\}$:

(1) if $\forall \alpha, \beta \in O$, $\alpha = \beta$, infer $\langle {}_n\Theta_i(P_1, \ldots, P_n)$, der, O_1, $R_1\rangle$;

(2) if $\exists \alpha, \beta \in O$ such that $\alpha \neq \beta$, and Combine($\langle F_1$, der, O_1, $R_k\rangle, \ldots, \langle F_{\binom{n}{i}}$, der, $O_{\binom{n}{i}}$, $R_{\binom{n}{i}}\rangle$, $\langle G_1$, der, O'_1, $R'_1\rangle, \ldots, \langle G_{i\times\binom{n}{i}}$, der, $O'_{i\times\binom{n}{i}}$, $R'_{i\times\binom{n}{i}}\rangle$), infer $\langle {}_n\Theta_i(P_1, \ldots, P_n)$, ext, $\cup\{O\}$, $\mu(R, O)\rangle$.

Thresh Elimination (ΘE).

(1) If $\langle {}_n\Theta_i(P_1, \ldots, P_n), t, o, r\rangle$, $\langle A_1, t_1, o_1, r_1\rangle, \ldots, \langle A_i, t_i, o_i, r_i\rangle$ are supported wffs, $\{A_1, \ldots, A_i\} \subset \{P_1, \ldots, P_n\}$, and Combine($\langle {}_n\Theta_i(P_1, \ldots, P_n), t, o, r\rangle$, $\langle A_1, t_1, o_1, r_1\rangle, \ldots, \langle A_i, t_i, o_i, r_i\rangle$), then, denoting by O and R the sets $O = \{o, o_1, \ldots, o_i\}$ and $R = \{r, r_1, \ldots, r_i\}$, infer $\langle B_1 \wedge \cdots \wedge B_{n-i}$, $\Lambda(t, t_1, \ldots, t_i)$, $\cup\{O\}$, $\mu(R, O)\rangle$ in which $\{B_1, \ldots, B_{n-i}\} = \{P_1, \ldots, P_n\} - \{A_1, \ldots, A_i\}$.

(2) If $\langle {}_n\Theta_i(P_1, \ldots, P_n), t, o, r\rangle$, $\langle A_1, t_1, o_1, r_1\rangle, \ldots, \langle A_{i-1}, t_{i-1}, o_{i-1}, r_{i-1}\rangle$ and $\langle \neg A_i, t_i, o_i, r_i\rangle$ are supported wffs, $\{A_i\} \cap \{A_1, \ldots, A_{i-1}\} = \emptyset$, $\{A_1, \ldots, A_{i-1}, A_i\} \subset \{P_1, \ldots, P_n\}$ and Combine($\langle {}_n\Theta_i(P_1, \ldots, P_n), t, o, r\rangle$, $\langle A_1, t_1, o_1, r_1\rangle, \ldots, \langle A_{i-1}, t_{i-1}, o_{i-1}, r_{i-1}\rangle$, $\langle \neg A_i, t_i, o_i, r_i\rangle$), then, denoting by O and R the sets $O = \{o, o_1, \ldots, o_{i-1}, o_i\}$ and $R = \{r, r_1, \ldots, r_{i-1}, r_i\}$, infer $\langle \neg B_1 \wedge \cdots \wedge \neg B_{n-i}, \Lambda(t, t_1, \ldots, t_{i-1}, t_i), \cup\{O\}$, $\mu(R, O)\rangle$ in which $\{B_1, \ldots, B_{n-i}\} = \{P_1, \ldots, P_n\} - \{A_1, \ldots, A_i\}$.

[23] This means that the $\binom{n}{i}$ entailments listed under (1) are verified.

[24] Notice that for each q there are i of these supported wffs, i.e., s ranges from 1 to $i \times \binom{n}{i}$, and thereby that the $i \times \binom{n}{i}$ entailments listed under (2) are verified.

4. MBR. The Abstract Level

4.1. Contexts and belief spaces

Having presented SWM, we now discuss how a belief revision system using SWM should interpret the extended wffs and how SWM's features can be used in applications of belief revision. Here, we provide what we call a *contextual interpretation* for SWM. We use the word "contextual interpretation" instead of just "interpretation" for the following two reasons: On the one hand, we want to stress that we are not providing an interpretation for SWM in the logician's sense of the word; on the other hand, we want to emphasize that our definition of truth depends on the notion of context. This contextual interpretation defines the behavior of an abstract belief revision system (i.e., not tied to any particular implementation), which we call MBR (multiple belief reasoner). MBR works with a knowledge base containing propositions that are associated with an OT, OS, and RS (in SWM's sense). The propositions are added to the knowledge base according to the rules of inference of SWM.

We define a *context* to be a set of hypotheses. A context determines a *belief space* (BS), which is the set of all the hypotheses defining the context and all the propositions that were derived exclusively from them. Within the SWM formalism, the supported wffs in a given BS are characterized by having an OS that is contained in the context. The set of contexts represented in the knowledge base is the power set of the set of hypotheses introduced.

Any operation performed within the knowledge base (query, addition, etc.) will be associated with a context. We refer to the context under consideration, i.e., the context associated with the operation currently being performed, as the *current context*. While the operation is being carried out, the only propositions that will be considered are the propositions in the BS defined by the current context. This BS will be called the *current belief space*. A proposition is said to be *believed* if it belongs to the current BS.

A common goal of belief revision systems is to stay away from contradictions, i.e., to avoid the simultaneous belief of a proposition and its negation. Taking this into account, it would seem natural to constrain contexts to be consistent sets of hypotheses, not just any sets of hypotheses. Let us note, however, that determining whether a contradiction is derivable from a set of hypotheses is a difficult problem in logic, and thus the condition that contexts are not inconsistent may be very difficult to enforce. For that reason, we may settle for the weaker condition that contexts are not *known* to be inconsistent. Within MBR, we can detect whether a context is not known to be inconsistent by considering the RSs of the hypotheses defining the context. Given the context $\{H_1, \ldots, H_n\}$, the condition $\forall(\mathbf{H})\, \mathbf{ot}(\mathbf{H}) = \text{hyp} \wedge \mathbf{wff}(\mathbf{H}) = H \wedge H \in \{H_1, \ldots, H_n\}\ \forall r \in \mathbf{rs}(\mathbf{H})\colon\ r \not\subset (\{H_1, \ldots, H_n\} - \{H\})$ guarantees that the context $\{H_1, \ldots, H_n\}$ is not known to be inconsistent.[25]

[25] The condition $\exists(\mathbf{H})\quad \mathbf{ot}(\mathbf{H}) = \text{hyp} \wedge \mathbf{wff}(\mathbf{H}) = H \wedge H \in \{H_1, \ldots, H_n\}\quad (\exists r \in \mathbf{rs}(\mathbf{H})\colon (\{H\} \cup r) \subset \{H_1, \ldots, H_n\})$ guarantees that the context $\{H_1, \ldots, H_n\}$ is *known* to be inconsistent.

However, it may be the case that in MBR one desires to perform reasoning within the BS defined by an inconsistent context (a kind of counterfactual reasoning). In SWM, the existence of contradictions is not as damaging as in classical logic, in which anything can be derived from a contradiction. Thus, in MBR one may not want to bother discarding hypotheses after a contradiction is detected, since the contradiction will not affect the entire system. For these reasons, the condition that a context is not known to be inconsistent will not be compulsory but rather advisable if one doesn't explicitly want to perform reasoning in a BS that is known to be inconsistent. The reason why it is advisable is that within a BS defined by a context not known to be inconsistent some simplification can be considered during the application of the rules of inference, as stated by the following theorems:

Theorem. *If C is a context that is not known to be inconsistent, then, for any two supported wffs, A and B, in the BS defined by the context C, we have* Combine(A, B) = true (Theorem 6 in Appendix A).

Corollary. *If one uses a context which is not known to be inconsistent, then MBR does not need to check for combinability between the supported wffs before the application of rules of inference* (Corollary 6.1 in the Appendix A).

4.2. The revision of beliefs

Let us now consider how MBR acts when a contradiction is detected. We discuss two levels of belief revision: belief revision within the current context and belief revision within a context strictly containing the current context. The main difference between them is that the former may require changes in the current context and allows the deduction of new wffs, while the latter leaves this context unchanged and does not allow the deduction of new wffs to the knowledge base. They are associated with the two rules of inference SWM has to handle contradictions: $\neg$I and URS.

The rule of $\neg$I states that from the combinable supported wffs $\langle A, t_1, o_1, r_1 \rangle$ and $\langle \neg A, t_2, o_2, r_2 \rangle$, we can deduce the negation of the conjunction of any number of hypotheses in $o_1 \cup o_2$ under an OS containing the remaining hypotheses. This rule may be applied whenever two contradictory wffs are found within the current BS. Its effect is twofold: (1) It may cause the current context to be changed. The fact that both A and $\neg A$ were derived within the current BS means that the current context is *now known* to be inconsistent. If one wants to maintain contexts that are not known to be inconsistent, then the current context has to be changed. (2) It allows the addition of new wffs to the knowledge base. Such wffs are negations of conjunctions, whose conjuncts are some hypotheses in the current context (the hypotheses in $o_1 \cup o_2$).

The rule of URS has the effect of recording the occurrence of contradictions in the RSs of all the hypotheses underlying a contradiction (and the supported wffs derived from them). This rule, however, does not allow the addition of new wffs to the knowledge base. This rule is obligatorily applied whenever two contradictory wffs are found, whether or not they belong to the current BS. Upon application of this rule, there will be an explicit record in the knowledge base about the possibility of the derivation of the contradictory wffs.

In summary, when a contradiction is detected, one of two things will happen:

(1) *Only one of the contradictory wffs belongs to the current BS*:[26] the contradiction is recorded (through the application of URS), but nothing more happens. The effect of doing so is to record that some set of hypotheses, strictly containing the current context, is now known to be inconsistent. This results in what we call *belief revision within a context strictly containing the current context.*

(2) *Both contradictory wffs belong to the current BS*: URS is applied, resulting in the updating of the RSs of the hypotheses in the current context (and the wffs derived from them), and, in addition, the rule of $\neg$I may also be applied. This results in what we call *belief revision within the current context*, normally producing disbelief in some of the hypotheses in the current context.

In this section we described how to "interpret" the supported wffs and defined *context* and *belief space*. This description was done at the abstract level, i.e., not tied up to any knowledge representation formalism.

5. SNeBR: An Implementation of MBR

5.1. Introduction

In this section, we describe a particular implementation of MBR using the SNePS semantic network processing system [46, 49]. The system we describe is called SNeBR.[27] The aspects of SNeBR discussed here are: the representation of supported wffs, in particular of those whose wff is obtained by multiple derivations; the representation of contexts; and some details of the process of revision of beliefs.

When using SNeBR, we can perform the following operations:

(1) *Add new hypotheses to the network.* There is a function that takes as arguments a proposition and the name of a context, adds the proposition to the network, and justifies it as a hypothesis that is added to the named context.[28]

[26] Note that at least one of the contradictory wffs belongs to the current BS, since a contradiction is detected whenever some newly derived wff contradicts some existing one, and newly derived wffs always belong to the current BS.

[27] SNePS with Belief Revision.

[28] In this function, and in all the other user functions available in SNeBR, if no context name is specified, it defaults to the name "current context".

(2) *Name a context.* We can assign a name to a given context and use that name whenever the context is being referenced.

(3) *Ask for all the nodes in a given BS that match a given pattern.* We can specify a node (which may contain free variables) and a context, and the network matching function will retrieve all the nodes that match the specified pattern and are part of the BS defined by the context.

(4) *Perform backward inference in a BS.* We can ask SNeBR to deduce a given proposition (possibly containing free variables) in the BS defined by a context. SNeBR will retrieve relevant rules in the specified BS and will try to derive the desired instances. All the instances derived will have an OT, OS, and RS computed by SNeBR according to the rules of inference of SWM.

(5) *Perform forward inference in a BS.* We can also ask SNeBR to perform forward reasoning with a given hypothesis in a given context. In this case, the hypothesis is built into the network, added to the context under consideration, and all its consequences are derived. All the instances derived will have an OT, OS, and RS computed by SNeBR according to the rules of inference of SWM.

5.2. Representation of propositions

A SNePS semantic network [46, 49], is a labeled directed graph in which nodes represent intensional concepts and arcs represent nonconceptual binary relations between concepts. One of the assumptions underlying the SNePS network is the so-called *uniqueness principle* [24]: each concept is represented in the network by a *unique* node. In SNePS, arcs are labeled with symbols intended to be mnemonically suggestive of the relation they represent. The relations represented solely by arc labels are not conceptual: they are used to form the basic structure of the semantic network.

In Fig. 6, we show two propositions in SNePS. The proposition represented

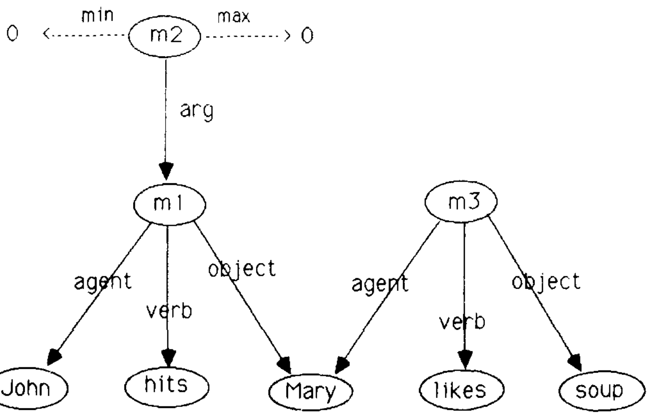

FIG. 6. SNePS nodes.

by node m3 asserts that "Mary likes soup". The proposition represented by node m2 asserts that the proposition represented by node m1 is false (m2 corresponds to the implementation of ${}_1\bowtie_0^0(\text{m1})$: the proposition ${}_n\bowtie_i^j(P_1, \ldots, P_n)$ is represented in the network by a node with arcs labeled "arg" to the nodes representing $P_1, \ldots, P_n$, an arc labeled "min" to i, and an arc labeled "max" to j). Thus, the proposition represented by node m2 asserts that "it is not the case that John hits Mary". SNePS "believes" every node that has no arcs pointing to it. Given the information of Fig. 6, SNePS would believe that Mary likes soup and that it is not the case that John hits Mary.

In SNeBR, propositions are represented by SNePS nodes. Associated with each node representing a proposition, there is another node, representing its support (the *supporting node*). The supporting node has arcs labeled "os" that point to the nodes representing the hypotheses in the OS and arcs labeled "rs" that point to the sets in the restriction set. Each of these sets is, in turn, represented by a node that has arcs labeled "ers" (element of the restriction set) to each hypothesis that it contains. The OT is represented by an arc (labeled either hyp, der, or ext) that connects the node representing the proposition with the supporting node. In Fig. 7, we show the network representation of hypotheses, and in Fig. 8, the representation of normally derived propositions (propositions with "ext" OTs have a similar representa-

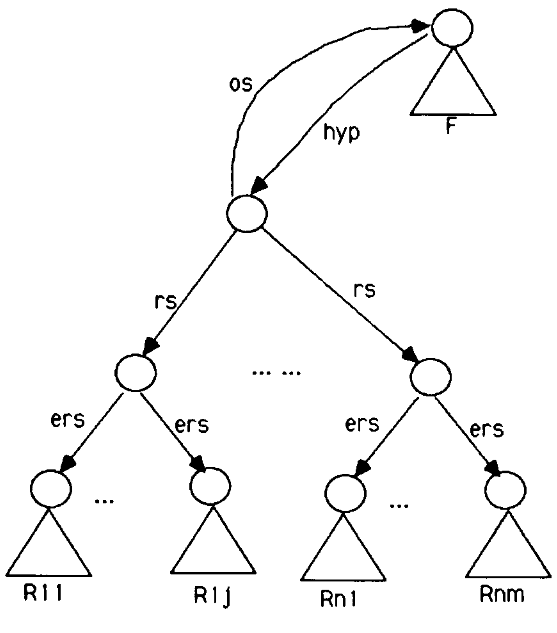

FIG. 7. Representation of ⟨F, hyp, {F}, {{R11, ..., R1j}, ..., {Rn1, ..., Rnm}}⟩.

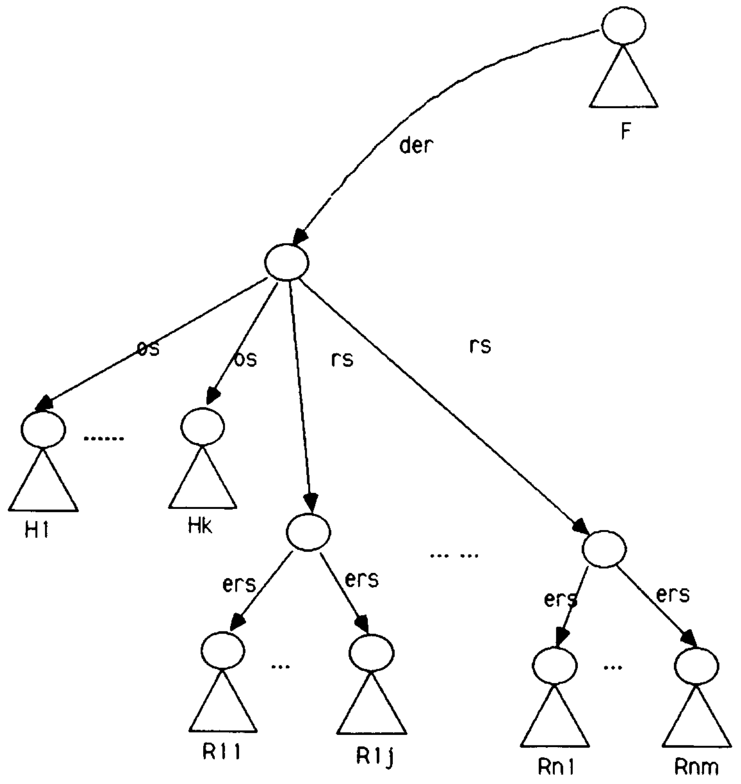

FIG. 8. Representation of ⟨F, der, {H1, …, Hk}, {{R11, …, R1j}, …, {Rn1, …, Rnm}}⟩.

tion). In these and in later figures, we use a triangle to denote the network structure corresponding to the proposition that is written below the triangle. Also, in our figures, some nodes may not have labels; this simply means that we do not care what the label is.

As opposed to SNePS, which believes every node that has no arcs pointing to it, SNeBR "believes" every node whose OS is contained in the current context. *The beliefs of SNeBR change as the current context changes.*

There are two aspects worth mentioning regarding the representation of propositions in SNeBR:

(1) *Representation of contradictory propositions.* The uniqueness principle (that is, each concept is represented by a unique node) guarantees that there is as much sharing among network structures as possible. According to this principle, two nodes representing contradictory propositions share a common network structure. To illustrate this, let us consider the supported wffs ⟨P, T1, {H1, H2}, { }⟩ and ⟨¬P, T2, {H3, H4}, { }⟩. These supported wffs share the proposition P. Their network representation is shown in Fig. 9. In this figure,

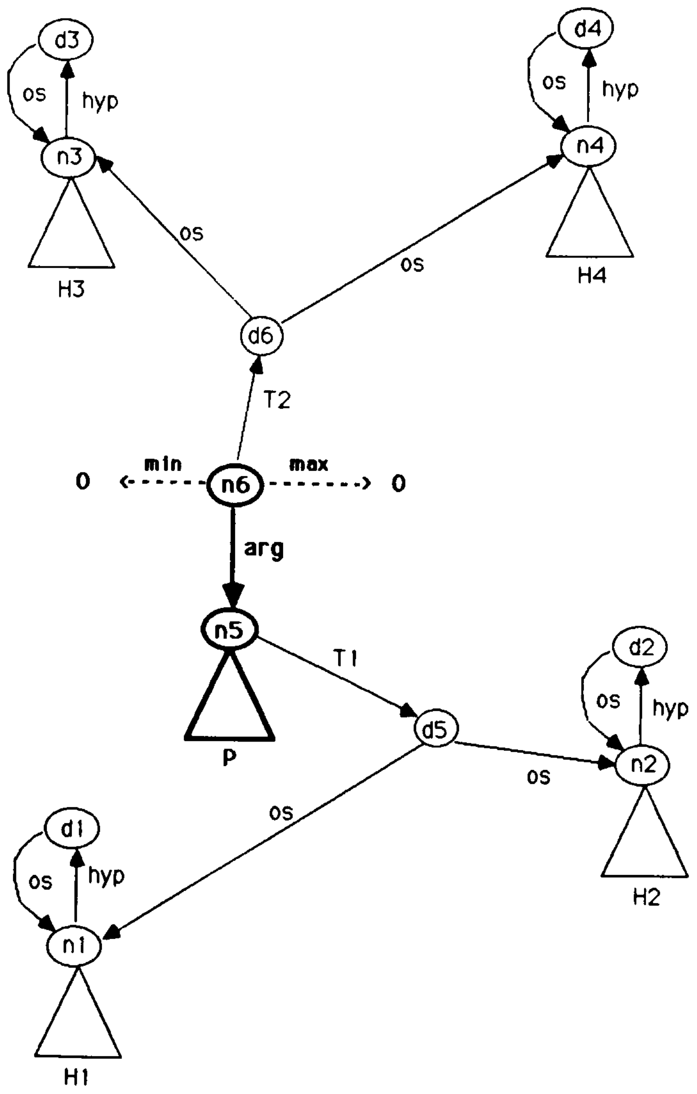

FIG. 9. Representation of contradictory propositions.

node n5 represents the proposition P and node n6 represents the proposition ¬P.

When a new node is about to be built, SNeBR first checks whether that node (or its negation) already exists in the network; in this case, the entire network is considered, not just that portion of it that belongs to the BS under consideration. The network matching function [44, 45] *guarantees* that this node is found without having to search every node in the network.

(2) *Representation of multiple derivations of the same proposition.* In SWM, if the same wff is derived from different sets of hypotheses,[29] then different supported wffs are added to the knowledge base—recall that the support of a supported wff pertains to a particular derivation of its wff. Suppose that a given proposition is derived in different ways (with different OTs or OSs); how should these multiple occurrences be represented? The uniqueness principle requires that the node representing this proposition be shared by the different occurrences of the supported wff. Our representation links the node that represents the proposition to multiple supporting nodes, each one of which represents one of the possible derivations. As an example, Fig. 10 shows the representation of the supported wffs $\langle$C, hyp, {C}, { }$\rangle$; $\langle$C, der, {A, A→C}, { }$\rangle$; and $\langle$C, der, {B, B→C}, {{D}}$\rangle$. In this figure, node d1 represents the support of $\langle$C, hyp, {C}, { }$\rangle$; node d2 represents the support of $\langle$C, der, {A, A→C}, { }$\rangle$; and node d3 represents the support of $\langle$C, der, {B, B→C}, {{D}}$\rangle$.

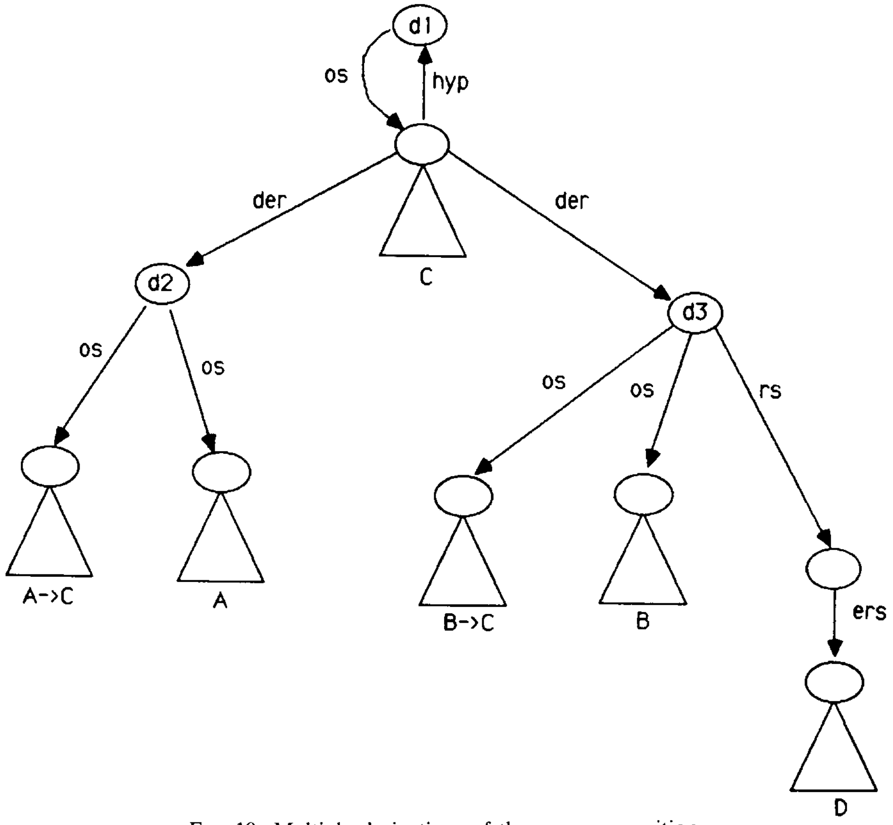

FIG. 10. Multiple derivations of the same proposition.

[29] Or with different OTs.

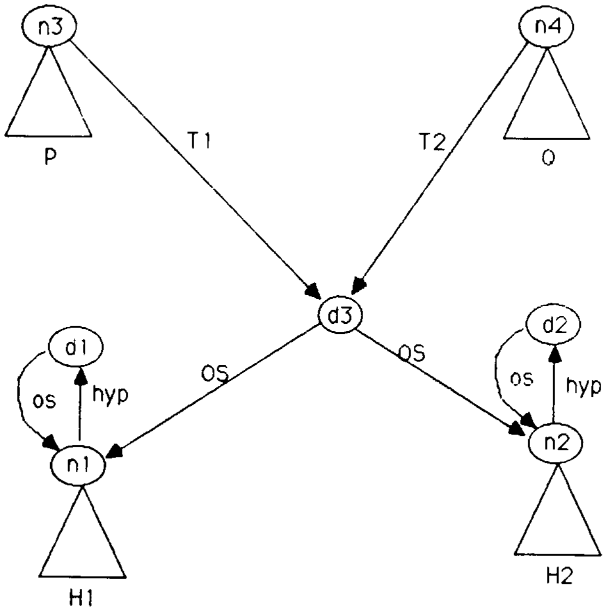

FIG. 11. Propositions sharing a supporting node.

Notice furthermore that since supported wffs with the same OS have the same RS (Theorem 5 in Appendix A), we allow network sharing between supporting nodes. This means that if there are two nodes corresponding to propositions with the same OS, then those nodes will share a common supporting node. This decision, besides yielding considerable savings in memory, allows some savings in processing as well when a network update (URS) due to the detection of a contradiction is being performed.

In Fig. 11, we show two supported wffs sharing the same supporting node (node d3); they correspond to the supported wffs ⟨P, T1, {H1, H2}, { }⟩ and ⟨Q, T2, {H1, H2}, { }⟩.

5.3. Representation of contexts

A context is represented by a set of nodes, each one of which represents a hypothesis. In order to allow a SNeBR user to refer to and talk about contexts, contexts can be named. The name of a context is represented in the network by a node that has arcs labeled ":val" to each node defining the context. Figure 12 shows the network representation of a context named "ct1" containing hypotheses H1 and H2. The reason for not having a circle around "ct1" and for the dashed arcs is beyond the scope of this paper and can be found in [46].

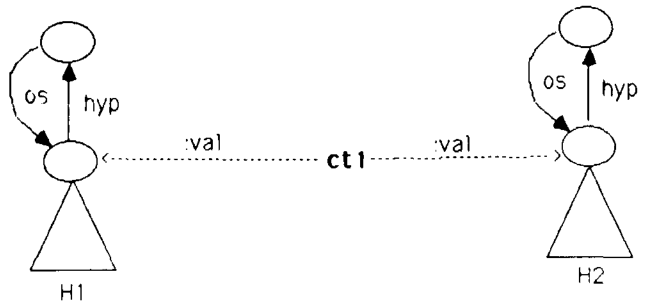

FIG. 12. Representation of the context "ct1".

5.4. Inference in SNeBR

SNeBR allows both backward and forward inference to be performed. Every rule in the BS under consideration may be used, in either backward or forward inference or both. When a rule is used, it is activated and remains that way until explicitly de-activated by the user. The activated rules are assembled into a set of processes, called an *active connection graph* (acg) [37, 38], which carry out the inferences. The acg also stores all the results generated by the activated rules. If, during some deduction, the inference system needs some of the rules activated during a previous deduction, it uses their results directly instead of re-deriving them. Forward and backward inference interface smoothly, giving rise to a behavior that we call *bi-directional inference* [48].

There are two main concepts involved in the implementation of the inference system: pattern matching and the use of procedural (or active) versions of rules. The *pattern matching* process is given a piece of the network (either to be deduced in backward inference or added in forward inference) and a context, and locates relevant rules in the BS defined by the context. The network matching function does a pattern match as if there were no contexts and then "filters" the nodes in the BS under consideration. The question of whether this "filtering" can be done before the match without extensive computation is still an open problem (see [25, pp. 7.5–7.16]). The rules retrieved by the match operation are then "compiled" into a *set of processes*, which are given to a multi-processing system for execution. The multi-processing system used by the inference system is called MULTI [37];[30] it is a LISP-based system mainly consisting of a simple evaluator, a scheduler, and system primitives. The evaluator continuously executes processes from a process queue until the queue becomes empty; the scheduler inserts processes into the process queue;

[30] Our approach was influenced both by Kaplan's producer-consumer model [21] and by Wand's frame model of computation [56].

system primitives include functions for creating processes, scheduling processes, and manipulating local variables or registers.

The inference is carried out by some of the processes that embody the rules of inference of SWM. There is one such process for each rule of inference. These processes are responsible for setting up the dependencies among propositions by computing the OT, OR, and RS for the results they generate. Details of the inference system and the processes it uses can be found in [25].

5.5. Handling contradictions

We now describe how contradictions are handled by SNeBR. A contradiction will be detected by SNeBR when one of the following conditions occurs:

(1) *Nodes representing contradictory wffs are built into the network.* Suppose that the inference system builds a node (say n2) representing proposition P, and discovers that there is a node (say n1) representing the proposition ¬P (the discovery of n1 is guaranteed by the uniqueness principle). Suppose furthermore that n1 has a supporting node, meaning that the proposition represented by n1 (¬P) belongs to some BS.[31] In this case, the rule of URS is immediately applied, having the effect of recording the new set known to be inconsistent. Afterwards, if both n1 and n2 belong to the current BS,[32] the rule of ¬I is applied and a decision has to be made about which hypothesis is the culprit for the contradiction. This decision is not made by the rules of inference but rather at another level. In the current implementation, this is done through an interaction with the user (see Section 6).

(2) *Information gathered by a process shows that a rule is invalidated by the data in the BS.* The other way to detect a contradiction in SNeBR is when a process, trying to derive instances of the consequents of a rule, gathers information that invalidates the rule. For example, the rule ${}_3\bowtie_1^1(A, B, C)$ states that *exactly* one of A, B, and C is true. If this rule exists in the current BS and SNeBR is asked whether C is true, it will try to find whether A or B are true. If it finds that *both* A and B are true, then it reports a contradiction, since the rule is invalidated by the data gathered. An example of this is shown in Section 6. Notice that this is just a shortcut to detect contradictions. In the case we just described, a contradiction would have been generated anyway (for example, the fact that A is true will cause the conclusion that B and C aren't). The detection of contradictions inside processes avoids reasoning that is known to lead to a contradiction.

[31] If n1 does not have a supporting node, then it means that n1 is a component of another proposition and thus does not contradict n2. For example, $\neg P$ and $P \vee Q$ are not contradictory propositions.

[32] The node n2 has just been derived; therefore it belongs to the current BS. The node n1 would belong to the current BS if one of its supporting nodes has an OS that is contained in the current context.

5.6. Implementation of URS

Let us now take a closer look at SNeBR's implementation of URS. This rule requires two traversals of the network: the first to reach the hypotheses underlying the contradiction, updating their RSs; the second, to update the RSs of all the wffs derived from them.

When a contradiction is found, the computation of all hypotheses underlying it is done by following the OT and os arcs *directly* linking the contradictory wffs with the hypotheses that they assume. The updating of the RSs of the hypotheses consists of introducing a new set in the RS of each one of them (or in modifying some existing RS), which is done by creating nodes representing those sets (or by deleting some of the arcs in the existing sets). To update the wffs derived from those hypotheses, only one arc has to be traversed for each hypothesis (the os^{-1} arc—the os converse arc—connecting the hypothesis with the supporting nodes of the wffs derived from it). Such wffs are updated by creating RSs or updating existing ones.

As an example, let us consider the network represented in Fig. 9, in which nodes n1, n2, n3, n4, and n5 represent, respectively, the hypotheses H1, H2, H3, H4, and the derived proposition P. Suppose that ¬P has just been derived. When SNeBR builds node n6 (representing ¬P), it detects a contradiction involving nodes n5 and n6, causing the rule of URS to be applied. The first step in the application of URS consists in computing the set of hypotheses underlying the contradiction. This set is computed by following the OT (in our example, these are T1 and T2) and os arcs leaving the contradictory nodes: from n5 we obtain {n1, n2}, and from n6 we obtain {n3, n4}; therefore, the set {n1, n2, n3, n4} has just been discovered to be inconsistent. The next step consists in updating the RS of each of the hypotheses underlying the contradiction, which is done by creating a new RS for each hypothesis. For example, when updating the RS of n1, we create a new RS (r1 in Fig. 13) with hypotheses represented by the nodes in the set {n2, n3, n4}. Finally, as the last step, the RSs of all the supported wffs (with "der" or "ext" OTs) depending on the inconsistent set are updated. This is done by following the os^{-1} arc leaving each of the contradictory hypotheses to find all the supporting nodes depending on that hypothesis (in this case, finding the supporting nodes d5 and d6) and updating their RS (the nodes r5 and r6 are created). Figure 13 represents the network of Fig. 9 after the application of URS.

There are two cases about the application of URS that are worth discussing:

(1) *The case when one of the contradictory nodes was obtained by more than one derivation* (it has more than one supporting node). An example of this situation is shown in Fig. 14. In this case, since n5 has two supporting nodes (d5 and d8), when n6 is derived the inference system uncovers two inconsistent sets: {H1, H2, H3, H4} and {H3, H4, H5} (represented by the nodes {n1, n2, n3, n4} and {n3, n4, n7}).

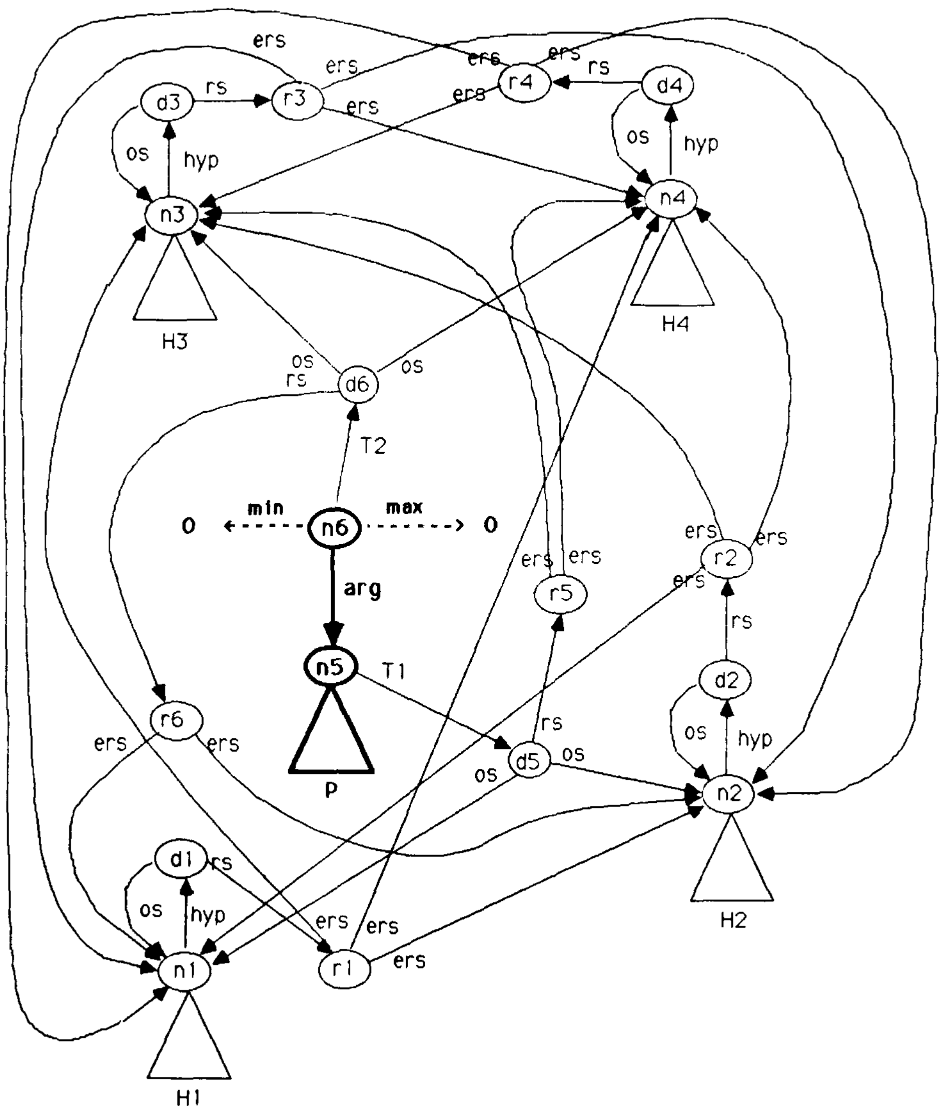

FIG. 13. Network after URS.

(2) *The case when RSs are simplified.* The rule of URS states that upon discovery of the inconsistent set $\mathscr{X}$, all the wffs that have an OS that is not disjoint from $\mathscr{X}$ should have their RSs updated. If $\langle F, t, o, r\rangle$ is a wff such that $o \cap \mathscr{X} \neq \emptyset$, then after application of URS this wff becomes $\langle F, t, o, \sigma(r \cup (\mathscr{X} - o))\rangle$. If "$r$" is the empty set, as was the case in the previous examples, then a new set is created in the wff's RS. However, if "r" is not the empty set,

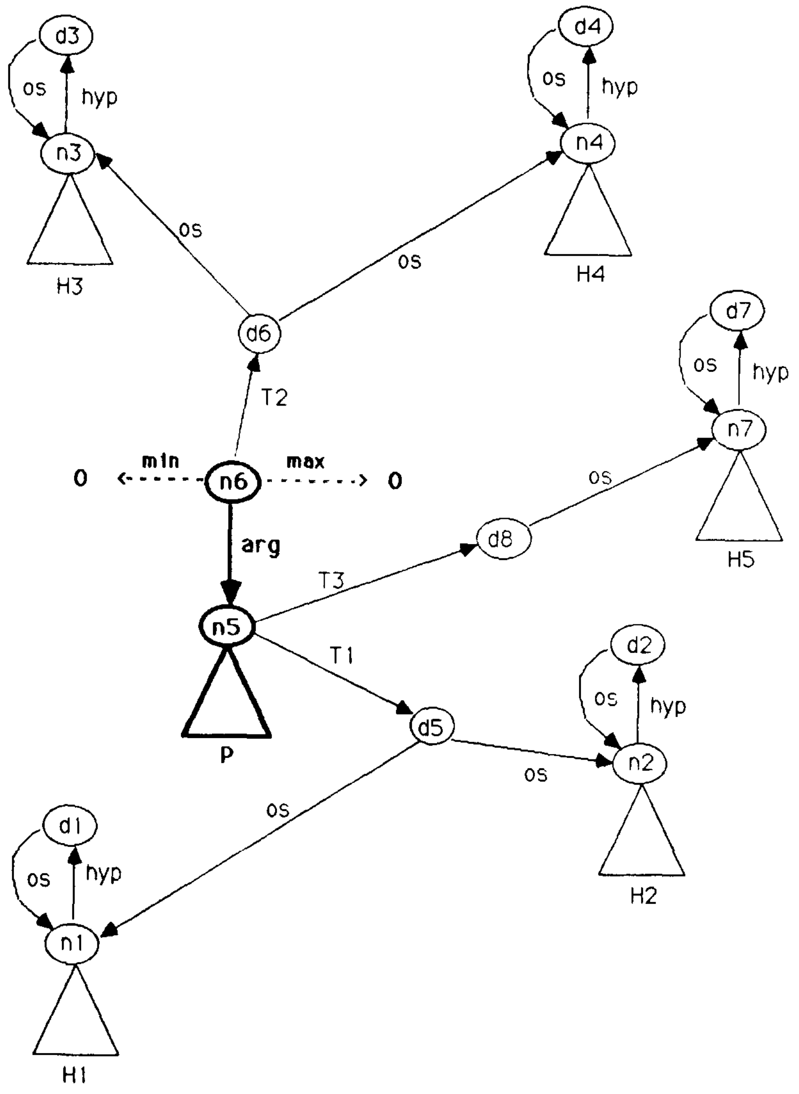

FIG. 14. Network before URS (node n5 has two supporting nodes).

the resulting RS may turn out to be smaller than the RS before the application of URS.

6. Selecting among Alternatives

In this section, we present an example of person-machine interaction by showing a run in which SNeBR solves the puzzle, "The Woman Freeman Will

Marry". In solving puzzles of this sort, one usually raises hypotheses, reasons from them, and, if a contradiction is detected, identifies the faulty hypotheses, replacing some of them and resuming the reasoning. The statement of the puzzle is as follows:

Freeman knows five women: Ada, Bea, Cyd, Deb and Eve.
(1) The women are in two age brackets: three women are under 30 and two women are over 30.
(2) Two women are teachers and the other three women are secretaries.
(3) Ada and Cyd are in the same age bracket.
(4) Deb and Eve are in different age brackets.
(5) Bea and Eve have the same occupation.
(6) Cyd and Deb have different occupations.
(7) Of the five women, Freeman will marry the teacher over 30.
Who will Freeman marry? (Summers [54, p. 6])

To use SNeBR to solve this puzzle, the propositions in the puzzle statement have to be represented in the network. What follows is a description of these propositions. The numbers associated with the wffs relate to the number of the node that represents it in the network. The wffs are described in a language called SNePSLOG [36], which is a logic programming interface to SNeBR. Assertions and rules written in SNePSLOG are stored as network structures in SNeBR; SNePSLOG queries are translated into deduction requests to SNeBR; output from SNeBR is translated back into SNePSLOG formulas.

– There are five women, Ada, Bea, Cyd, Deb, and Eve.

wff1: Woman(Ada);
wff2: Woman(Bea);
wff3: Woman(Cyd);
wff4: Woman(Deb);
wff5: Woman(Eve).

– The women are in two age brackets: three women are under 30 (wff12)[33] and

[33] With this proposition we can see the advantage of the SNePS connectives. With the standard connectives this proposition would have to be expressed in the following way: (¬age(Ada, u30) ∧ ¬age(Bea, u30) ∧ age(Cyd, u30) ∧ age(Deb, u30) ∧ age(Eve, u30)) ∨ (¬age(Ada, u30) ∧ age(Bea, u30) ∧ ¬age(Cyd, u30) ∧ age(Deb, u30) ∧ age(Eve, u30)) ∨ (¬age(Ada, u30) ∧ age(Bea, u30) ∧ age(Cyd, u30) ∧ ¬age(Deb, u30) ∧ age(Eve, u30)) ∨ (¬age(Ada, u30) ∧ age(Bea, u30) ∧ age(Cyd, u30) ∧ age(Deb, u30) ∧ ¬age(Eve, u30)) ∨ (age(Ada, u30) ∧ ¬age(Bea, u30) ∧ ¬age(Cyd, u30) ∧ age(Deb, u30) ∧ age(Eve, u30)) ∨ (age(Ada, u30) ∧ ¬age(Bea, u30) ∧ age(Cyd, u30) ∧ ¬age(Deb, u30) ∧ age(Eve, u30)) ∨ (age(Ada, u30) ∧ ¬age(Bea, u30) ∧ age(Cyd, u30) ∧ age(Deb, u30) ∧ ¬age(Eve, u30)) ∨ (age(Ada, u30) ∧ age(Bea, u30) ∧ ¬age(Cyd, u30) ∧ ¬age(Deb, u30) ∧ age(Eve, u30)) ∨ (age(Ada, u30) ∧ age(Bea, u30) ∧ ¬age(Cyd, u30) ∧ age(Deb, u30) ∧ ¬age(Eve, u30)) ∨ (age(Ada, u30) ∧ age(Bea, u30) ∧ age(Cyd, u30) ∧ ¬age(Deb, u30) ∧ ¬age(Eve, u30)).

two women are over 30 (wff18). It is implicit in this sentence that every woman is either under 30 or over 30 (wff27).

wff12: ${}_{5}\bowtie_{3}^{3}$(age(Ada, u30), age(Bea, u30), age(Cyd, u30), age(Deb, u30), age(Eve, u30));
wff18: ${}_{5}\bowtie_{2}^{2}$(age(Ada, o30), age(Bea, o30), age(Cyd, o30), age(Deb, o30), age(Eve, o30));
wff27: $\forall$(x) Woman(x) $\rightarrow$ ${}_{2}\bowtie_{1}^{1}$(age(x, u30), age(x, o30)).

– Two women are teachers (wff33) and the other three women are secretaries (wff39). The *the* in the previous sentence conveys the information that no woman is both a teacher and a secretary (wff48).

wff33: ${}_{5}\bowtie_{2}^{2}$(worker (Ada, teacher), worker(Bea, teacher), worker(Cyd, teacher), worker(Deb, teacher), worker(Eve, teacher));
wff39: ${}_{5}\bowtie_{3}^{3}$(worker(Ada, secretary), worker(Bea, secretary), worker(Cyd, secretary), worker(Deb, secretary), worker(Eve, secretary));
wff48: $\forall$(x) Woman(x) $\rightarrow$ ${}_{2}\bowtie_{1}^{1}$(worker(x, secretary), worker(x, teacher)).

– Ada and Cyd are in the same age bracket (wff53).

wff53: $\forall$(x) ${}_{2}\Theta_{1}$(age(Ada, x), age(Cyd, x)).

– Deb and Eve are in different age brackets.

wff58: $\forall$(x) ${}_{2}\bowtie_{1}^{1}$(age(Deb, x), age(Eve, x)).

– Bea and Eve have the same occupation.

wff63: $\forall$(x) ${}_{2}\Theta_{1}$(worker(Bea, x), worker(Eve, x)).

– Cyd and Deb have different occupations.

wff68: $\forall$(x) ${}_{2}\bowtie_{1}^{1}$(worker(Cyd, x), worker(Deb, x)).

– Exactly one woman over 30 is a teacher.

wff79: ${}_{5}{\times\!\!\times}_{1}^{1}({}_{2}{\times\!\!\times}_{2}^{2}$(age(Ada, o30), worker(Ada, teacher)),
${}_{2}{\times\!\!\times}_{2}^{2}$(age(Bea, o30), worker(Bea, teacher)),
${}_{2}{\times\!\!\times}_{2}^{2}$(age(Cyd, o30), worker(Cyd, teacher)),
${}_{2}{\times\!\!\times}_{2}^{2}$(age(Deb, o30), worker(Deb, teacher)),
${}_{2}{\times\!\!\times}_{2}^{2}$(age(Eve, o30), worker(Eve, teacher))).

– Freeman will marry the teacher over 30.

wff88: ∀(x) ${}_{2}\Theta_{1}$(marry(Freeman, x),
${}_{2}{\times\!\!\times}_{2}^{2}$(age(x, o30), worker(x, teacher))).

Using the propositions described above, we build into the network the hypotheses shown in Fig. 15—where wff6 represents the proposition ${}_{5}{\times\!\!\times}_{5}^{5}$(wff5, wff4, wff3, wff2, wff1), and wff89 represents the proposition ${}_{12}{\times\!\!\times}_{12}^{12}$(wff88, wff79, wff68, wff63, wff58, wff53, wff48, wff39, wff33, wff27, wff18, wff12). The hypothesis represented by wff6 states that there are five women and names those women, and the hypothesis represented by wff89 asserts all the specific information pertaining to these women and their relationship with Freeman. To solve the puzzle, we raise hypotheses about the ages and professions of the women and ask SNeBR to deduce whom Freeman will marry under those assumptions.[34] If the hypotheses raised are consistent with the puzzle's statement, the desired answer will be returned; otherwise, a contradiction will be detected.

Suppose that we raise the hypotheses shown in Fig. 16—where wff13 represents the proposition age(Ada, o30); wff15 represents the proposition age(Cyd, o30); wff28 represents the proposition worker(Ada, teacher); and wff31 represents the proposition worker(Deb, teacher). These hypotheses specify which women are supposed to be over 30 and which women are supposed to be teachers. Again, the numbers associated with the wffs correspond to the number of the node that represents it in the network. Suppose, furthermore,

⟨${}_{5}{\times\!\!\times}_{5}^{5}$(wff5, wff4, wff3, wff2, wff1), hyp, {wff6}, { }⟩
⟨${}_{12}{\times\!\!\times}_{12}^{12}$(wff88, wff79, wff68, wff63, wff58, wff53, wff48, wff39, wff33, wff27, wff18, wff12), hyp, {wff89}, { }⟩

FIG. 15. Hypotheses representing the puzzle's statement.

[34] Notice that specifying the ages of the two women over 30 completely determines the ages of the five women and that specifying the names of the two women who are teachers completely determines the profession of the five women.

⟨age(Ada, o30), hyp, {wff13}, { }⟩

⟨age(Cyd, o30), hyp, {wff15}, { }⟩

⟨worker(Ada, teacher), hyp, {wff28}, { }⟩

⟨worker(Deb, teacher), hyp, {wff31}, { }⟩

FIG. 16. Hypotheses raised.

that we ask who Freeman will marry in the BS defined by the context {wff6, wff13, wff15, wff28, wff31, wff89}. In this BS, there is no assertion about who Freeman will marry, but wff88 may enable its deduction. SNeBR sets up two subgoals, finding who is over 30 and finding who is a teacher (Fig. 17—in this and later figures, a "?" before a name means that the name is a variable).

Figure 18 shows SNeBR's deduction that Freeman will marry Ada. The inference does not stop here, however, since there is still information being propagated through the set of active processes and SNeBR reports inferences as shown in Fig. 19. Taking into account this information, a contradiction is detected (Fig. 20). In fact, Fig. 19 shows the inference that Deb is over 30. This means that there are now three women who are over 30 (Ada, Cyd and, now, Deb), which contradicts the wff $_5\bowtie_2^2$(age(Ada, o30), age(Bea, o30), age(Cyd, o30), age(Deb, o30), age(Eve, o30)).

Upon detecting the contradiction, SNeBR gives the options of continuing the reasoning within the inconsistent BS, of modifying the current context in order to obtain a consistent BS, or of giving up the request. Figures 20, 21,[35] and 22 show the interaction between the user and SNeBR during the process of belief revision. (The user's answers appear in bold.) Here, one of the advantages of

```
I wonder if marry(Freeman, ?who) holds within the BS defined by the context (wff31 wff28 wff15 wff13 wff89
wff6).

Let me try to use the rule ∀(?x) ₂Θ₁(marry(Freeman, ?x), ₂⋈²₂(age(?x, o30), worker(?x, teacher))).

I wonder if age(?x, o30) holds within the BS defined by the context (wff31, wff28, wff15, wff13, wff89, wff6).

I know age(Cyd, o30).

I know age(Ada, o30).

I wonder if worker(?x, teacher) holds within the BS defined by the context (wff31, wff28, wff15, wff13, wff89,
wff6).

I know worker(Deb, teacher).

I know worker(Ada, teacher).
```

FIG. 17. Who will Freeman marry?

[35] Notice that URS has already been applied.

Since worker(Ada, teacher) and age(Ada, o30)

I infer marry(Freeman, Ada).

FIG. 18. Freeman will marry Ada.

Since age(Ada, o30) and age(Cyd, o30)

I infer $_{2}{\bowtie}_{0}^{0}$(age(Eve, o30))

Since not age(Eve, o30)

I infer age(Deb, o30)

FIG. 19. Deb is over 30.

using SNeBR becomes apparent. During the deduction, all dependencies among propositions were computed by SNeBR itself, and upon detection of the contradiction we will be informed about *exactly* which hypotheses underlie the contradiction.

The *actual* revision of beliefs is carried out by the user through an interaction with the system. Work is being developed towards passing the responsibility of revising the beliefs to SNeBR itself [30]. At the present writing, SNeBR is a system that automatically computes dependencies, discovers contradictions, records them, and switches between belief spaces.

Figure 21 shows the inspection of the hypotheses that are responsible for the contradiction. Although the current context is {wff6, wff13, wff15, wff28, wff31, wff89}, only wff13, wff15, and wff89 were used in the derivation of the contradiction, and thus they are the only ones that *must* be changed in order to restore consistency, i.e., the only ones whose change may restore consistency. It should be further noticed that the SWM formalism guarantees that removing *exactly* one of them will generate a context that is not known to be inconsistent. In our example, we keep the hypothesis concerning the statement of the puzzle (wff89) and discard the hypotheses concerning the women's ages (wff13

WARNING!
Contradiction detected in the following and-or:
$_{5}{\bowtie}_{2}^{2}$(age(Ada, o30), age(Bea, o30), age(Cyd, o30), age(Deb, o30), age(Eve, o30)).
More true arguments than max.
Arguments in wrong number age(Ada, o30), age(Cyd, o30), age(Deb, o30)
You have the following options:

(1) Continue anyway, knowing that a contradiction is derivable.
(2) Re-start the exact same request in a different context which is not inconsistent.
(3) Drop the request altogether.

Do you want to continue anyway?
$\Rightarrow\Leftarrow$ **n**
Do you want to re-start the request in a new context?
$\Rightarrow\Leftarrow$ **yes**

FIG. 20. A contradiction is detected.

```
In order to make the context consistent you must delete one hypothesis from the set (wff13, wff15, wff89).

Do you want to take a look at wff13?
⇛⇚ n
There are 5 propositions depending on wff13:
(wff97, wff16, wff93, wff91, wff90).
Do you want to look at [a]ll of them, [s]ome of them, or [n]one?
⇛⇚ a
```

$\langle {}_{1}\bowtie^{0}_{0}$(marry(Freeman, Eve)), ext, {wff13, wff28, wff89}, {{wff15}}⟩

```
...
What do you want to do with wff13?
[d]iscard from the context, [k]eep in the context, [u]ndecided
⇛⇚ d

Do you want to take a look at wff15?
⇛⇚ y

⟨age(Cyd, o30), hyp, {wff15}, {{wff13, wff89}}⟩

There are 2 propositions depending on wff15:
(wff16, wff91).
Do you want to look at [a]ll of them, [s]ome of them, or [n]one?
⇛⇚ n
What do you want to do with wff15?
[d]iscard from the context, [k]eep in the context, [u]ndecided
⇛⇚ d

Do you want to take a look at wff89?
⇛⇚ n
There are 8 propositions depending on wff89:
(wff97, wff95, wff16, wff94, wff93, wff92, wff91, wff90).
Do you want to look at [a]ll of them, [s]ome of them, or [n]one?
⇛⇚ n
What do you want to do with wff89?
[d]iscard from the context, [k]eep in the context, [u]ndecided
⇛⇚ k
```

FIG. 21. Inspecting the inconsistent set of hypotheses.

```
The following (not known to be inconsistent) set of hypotheses was also part of the context where the
contradiction was derived:
(wff31, wff28, wff6).
Do you want to inspect or discard some of them?
⇛⇚ n

Do you want to add some new hypotheses?
⇛⇚ y
Enter an hypothesis using SNePSLOG
⇛⇚ age(Bea, o30)
Do you want to enter another hypothesis?
⇛⇚ y
Enter an hypothesis using SNePSLOG
⇛⇚ age(Deb, o30)
Do you want to enter another hypothesis?
⇛⇚ n
```

FIG. 22. Adding new hypotheses.

I wonder if marry(Freeman, who)
holds within the BS defined by the context
(wff14, wff16, wff6, wff28, wff31, wff89).

I know age(Deb, o30).

I know age(Bea, o30).

I know worker(Deb, teacher).

I know worker(Ada, teacher).

FIG. 23. Resuming the deduction.

and wff15). We also enter new hypotheses concerning the women's ages (Fig. 22).

After revising the beliefs, the inference resumes (Figs. 23 and 24). In this case there are no further contradictions detected and SNeBR reports that Freeman will marry Deb and will not marry Ada, Bea, Cyd or Eve. We should point out that some of the results derived in the first BS (and not shown in the figures), for example, that Bea is not a teacher, will be used, without further deduction, in this new BS since they are part of it. Two of the propositions returned are shown in Fig. 25. Notice that there are two ways of deducing that Freeman will not marry Eve and thus the two supports, one for each of the possible derivations.

The main point of this section was to illustrate how SNeBR can be used, the automatic computation of the dependencies of the propositions, and the package that allows the revision of the beliefs together with the inspection of the propositions that will become disbelieved by the removal of a given hypothesis. Also referred to in this section was the sharing of information derived in different BSs, without the need for re-deriving it.

Since ${}_{2}⨳_{2}^{2}$(age(Deb, o30), worker(Deb, teacher))
I infer marry(Freeman, Deb).

FIG. 24. Freeman will marry Deb.

⟨marry(Freeman, Deb), ext, {wff16, wff31, wff89}, {{wff13, wff15}}⟩

⟨${}_{1}⨳_{0}^{0}$(marry(Freeman, Eve)), ext, {wff16, wff31, wff89}, {{wff13, wff15}}⟩
⟨${}_{1}⨳_{0}^{0}$(marry(Freeman, Eve)), ext, {wff13, wff28, wff89}, {{wff15}}⟩

FIG. 25. Some of the propositions returned.

7. Concluding Remarks

Belief revision is important whenever reasoning is performed within a knowledge base that may contain contradictory information. Belief revision systems are capable of considering only part of the knowledge base (the set of *believed propositions*), performing inferences from this set, and, if a contradiction is detected, replacing this set by another one (*changing their beliefs*), disregarding every proposition that does not belong to the new set.

We presented SWM, a logic that captures the notion of propositional dependency and is able to deal with contradictions. SWM associates two sets with each derivation of a proposition: the *origin set* contains *every* hypothesis used in the derivation of the proposition; the *restriction set* contains those sets of hypotheses that are incompatible with the proposition's origin set.

We defined the behavior of MBR, an abstract program based on SWM. In MBR, a *context* is any set of hypotheses. A context determines a *belief space* (BS), which is the set of all propositions whose origin set is contained in the context: a BS contains all the propositions that depend exclusively on the hypotheses defining the context. At any moment, the only propositions that are believed (and thus retrievable from the knowledge base) are the ones that belong to the BS under consideration.

We presented SNeBR, an implementation of MBR using SNePS, and followed an example of its use. Our example shows that, after detection of a contradiction, SNeBR permits the identification of *exactly* which hypotheses contributed to the contradiction and the inspection of the consequences of removing each one of them. The SWM formalism that underlies SNeBR guarantees that the removal of exactly one of those hypotheses (no matter which) produces a context that is not known to be inconsistent.

In Section 1 we identified five problem areas that researchers in belief revision have to address: inference, nonmonotonicity, dependency recording, disbelief propagation, and the revision of beliefs. Our work contributed to the area of *inference* by the development of a logic that dictates the inferences of the system, computes the dependencies of propositions, and guarantees that the results produced make sense. We did not address the nonmonotonicity problem. Concerning *dependency recording*, we defined an assumption-based system; in fact SNeBR was the first assumption-based system implemented. *Disbelief propagation* is done by the use of contexts and belief spaces: the knowledge base retrieval function takes a context as one of its arguments and only considers the propositions in the belief space defined by this context. Finally, the *revision of beliefs* is done through an interaction with the user. The recording of the occurrence of contradictions is done in the RSs of the propositions that depend on the hypotheses involved in the contradiction.

Appendix A. Theorems about the SWM System

In this appendix we prove some of the properties of the SWM system.

Theorem 1. *Given the supported wffs $\langle A, t_a, o_a, r_a\rangle$ and $\langle B, t_b, o_b, r_b\rangle$, then the supported wff $\langle C, t_c, o_a \cup o_b, \mu(\{r_a, r_b\}, \{o_a, o_b\})\rangle$ has a minimal RS, regardless of whether or not $\langle A, t_a, o_a, r_a\rangle$ and $\langle B, t_b, o_b, r_b\rangle$ have minimal RSs.*

Proof. Suppose by way of contradiction that $\langle C, t_c, o_a \cup o_b, \mu(\{r_a, r_b\}, \{o_a, o_b\})\rangle$ does not have a minimal RS. This entails that one of the following two cases holds:

(1) $\exists(r)\, r \in \mu(\{r_a, r_b\}, \{o_a, o_b\})$: $r \cap (o_a \cup o_b) \neq \emptyset$.

(2) $\exists(r, s)\, r,s \in \mu(\{r_a, r_b\}, \{o_a, o_b\}) \wedge r \neq s$: $r \subset s$.

Suppose that condition (1) holds. Since $r \in \mu(\{r_a, r_b\}, \{o_a, o_b\})$ we can infer that $r \in \Psi(r_a \cup r_b, o_a \cup o_b)$, which, due to the way Ψ was defined, contradicts the fact that $r \cap (o_a \cup o_b) \neq \emptyset$.

Suppose that condition (2) holds. Since $r,s \in \mu(\{r_a, r_b\}, \{o_a, o_b\})$ we can infer that $r,s \in \sigma(\Psi(r_a \cup r_b, o_a \cup o_b))$, which, due to the way σ was defined, contradicts the fact that $r \subset s$.

Therefore $\langle C, t_c, o_a \cup o_b, \mu(\{r_a, r_b\}, \{o_a, o_b\})\rangle$ has minimal RS. $\square$

Theorem 2. *Given the supported wffs $\langle A, t_a, o_a, r_a\rangle$ and $\langle B, t_b, o_b, r_b\rangle$, then $\langle C, t_c, o_a - o_b, \int (o_a - o_b)\rangle$ has minimal RS.*

Proof. Suppose that $o_a - o_b = \{H_1, \ldots, H_n\}$. Taking into account that $\langle C, t_c, o_a - o_b, \int (o_a - o_b)\rangle$ is the same as $\langle C, t_c, o_a - o_b, \mu(\{\mathbf{rs}(\langle H_1, \text{hyp}, \{H_1\}, R_{H_1}\rangle), \ldots, \mathbf{rs}(\langle H_n, \text{hyp}, \{H_n\}, R_{H_n}\rangle)\}, \{\{H_1\}, \ldots, \{H_n\}\})\rangle$ the statement of this theorem follows from Theorem 1. $\square$

Theorem 3. *If A is an inconsistent set, then so is any set containing A.*

Proof. The proof presented is based on the fact that a proof of B from $\{A_1, \ldots, A_n\}$ is a sequence of lines, the first n of which are $A_1, \ldots, A_n$ and the last of which is B. Each line between A_n and B is obtained from the previous line(s) by the application of some rule of inference and is justified by $R(L_1, L_2)$ in which R represents some rule of inference and (L_1, L_2) are the line numbers of the parent wffs.

Suppose that $A = \{P_1, \ldots, P_n\}$ and let $r_1(L_{11}, L_{12})$, $r_2(L_{21}, L_{22}), \ldots, r_k(L_{k1}, L_{k2})$ represent the sequence of applications of rules of inference to the (ordered) set A (and to the wffs derived from them) to generate a contradiction. Suppose that $A \subset B$ and $B - A = \{S_1, \ldots, S_i\}$. B can, therefore, be written as the following ordered set (in which the order of $S_1, \ldots, S_i$ is irrelevant) $B = \{S_1, \ldots, S_i, P_1, \ldots, P_n\}$. Then, letting $L'_{fg} = L_{fg} + i$, the following sequence of rules of inference $r_1(L'_{11}, L'_{12})$, $r_2(L'_{21}, L'_{22}), \ldots, r_k(L'_{k1}, L'_{k2})$ describes a derivation of a contradiction from B. $\square$

Corollary 3.1. *If A is not known to be inconsistent, then neither is any set contained in A.*

Proof. Suppose by way of contradiction that A is not known to be inconsistent and that $B \subset A$ is known to be inconsistent. By the Theorem 3, A is known to be inconsistent, which is a contradiction. $\square$

Theorem 4. *All the supported wffs in the knowledge base resulting from the application of the rules of inference of the SWM system have minimal RSs.*

Proof. The proof will be done by complete induction on the number of applications of rules of inference.

The supported wffs which can be obtained by applying only one rule of inference are hypotheses and the rule of Hyp guarantees that they have minimal RSs.

Suppose now, by induction hypothesis, that all the supported wffs obtained by the application of less than n rules of inference have minimal RSs. We will have to prove that a supported wff obtained by the application of n rules of inference has minimal RSs as well:

Case 1. If the supported wff is obtained by the application of either $\neg$E, $\wedge$E, $\vee$I, $\exists$I, or $\exists$E, then it has the same OS and RS as the parent supported wff and consequently has minimal RS.

Case 2. If the supported wff is obtained by the application of either MP, MT, $\wedge$I, $\vee$E, or $\forall$E, then by Theorem 1 we may conclude that it has minimal RS.

Case 3. If the supported wff is obtained by the application of $\rightarrow$I, or $\forall$I, then by Theorem 2 we can conclude that it has minimal RS.

Case 4. If the supported wff is obtained by $\neg$I, we have to show that a supported wff obtained using the second part of the rule of $\neg$I has minimal RS (to show that a supported wff obtained using the first part of the rule of $\neg$I has minimal RS as well it suffices to take $o_1 = o_2 = o$ in the proof that follows). Consider $\langle A, t_1, o_1, r_1 \rangle$ and $\langle \neg A, t_2, o_2, r_2 \rangle$. For $\{H_1, \ldots, H_n\} \subset (o_1 \cup o_2)$, the second part of the rule of $\neg$I allows us to deduce $\langle \neg(H_1 \wedge \cdots \wedge H_n)$, ext, $(o_1 \cup o_2) - \{H_1, \ldots, H_n\}$, $\int ((o_1 \cup o_2) - \{H_1, \ldots, H_n\})\rangle$. To show that this supported wff has minimal RS let us suppose that it is obtained in two steps: first, deduce $\langle A \wedge \neg A$, ext, $o_1 \cup o_2$, $\mu(\{r_1, r_2\}, \{o_1, o_2\})\rangle$; second, from the above supported wff deduce $\langle \neg(H_1 \wedge \cdots \wedge H_n)$, ext, $(o_1 \cup o_2) - \{H_1, \ldots, H_n\}$, $\int ((o_1 \cup o_2) - \{H_1, \ldots, H_n\})\rangle$, by Theorem 2 this supported wff has minimal RS.

Case 5. If the supported wff is obtained by URS (upon discovery that the set $\mathcal{X}$ is inconsistent), we have to consider two subcases:

(a) The supported wff results from updating the RS of a hypothesis. Suppose that the supported wff is obtained by updating $\langle H, \text{hyp}, \{H\}, R\rangle$. We know that, prior to the updating, H has minimal RS. We want to show that $\langle H, \text{hyp}, \{H\}, \sigma(R \cup \{\mathscr{X} - \{H\}\})\rangle$ verifies the following conditions:

(i) $\forall(r)\; r \in \sigma(R \cup \{\mathscr{X} - \{H\}\})$: $r \cap \{H\} = \emptyset$.

(ii) $\forall(r, s)\; r,s \in \sigma(R \cup \{\mathscr{X} - \{H\}\}) \wedge r \neq s$: $r \not\subset s$.

Condition (i) holds since $\forall r \in R$, $r \cap \{H\} = \emptyset$ (before the update H had minimal RS) and $[\mathscr{X} - \{H\}] \cap \{H\} = \emptyset$ (by definition). Condition (ii) is verified by definition of σ.

(b) If the supported wff results from updating the RS of a supported wff whose OT is either "der" or "ext". Suppose that the supported wff is obtained by updating $\langle F, t, o, R\rangle$ ($t \neq \text{hyp}$). We want to show that $\langle F, t, o, \sigma(R \cup \{\mathscr{X} - o\})\rangle$ verifies the following two conditions:

(i) $\forall(r)\; r \in \sigma(R \cup \{\mathscr{X} - o\})$: $r \cap o = \emptyset$.

(ii) $\forall(r, s)\; r,s \in \sigma(R \cup \{\mathscr{X} - o\}) \wedge r \neq s$: $r \not\subset s$.

Suppose by way of contradiction that condition (i) does not hold. Since $\langle F, t, o, R\rangle$ has minimal RS we know that $\forall r \in R$, $r \cap o = \emptyset$, thereby $[\mathscr{X} - o] \cap o \neq \emptyset$, which is a contradiction, therefore condition (i) holds. Condition (ii) follows from the definition of σ.

Therefore all the supported wffs in the knowledge base resulting from the application of the rules of inference of the SWM system have minimal RS. □

Lemma 1. *Given n supported wffs* $\langle F_1, t_1, o_1, r_1\rangle, \ldots, \langle F_n, t_n, o_n, r_n\rangle$, *then the sets* $R_1 = \mu(\{r_1, \ldots, r_n\}, \{o_1, \ldots, o_n\})$ *and* $R_2 = \mu(\{\mu(\{r_1, \ldots, r_i\}, \{o_1, \ldots, o_i\}), \mu(\{r_{i+1}, \ldots, r_n\}, \{o_{i+1}, \ldots, o_n\})\}, \{o_1, \ldots, o_n\})$ $(1 \leq i \leq n-1)$ *are equal.*

Proof. Suppose by way of contradiction that $R_1 \neq R_2$. This means that either $\exists r \in R_1$ such that $r \notin R_2$ or that $\exists r \in R_2$ such that $r \notin R_1$. We will consider each of these cases in turn:

Case 1. Suppose that $r \in R_1$ and that $r \notin R_2$. Suppose furthermore that r was originated from a set s, i.e., $r = s - (o_1 \cup \cdots \cup o_n)$ and suppose that $s \in r_j$. Letting $O = o_1 \cup \cdots \cup o_n$ we have that $r = s - O$. The fact that r belongs to R_1 means that $\forall r_k \in \{r_1, \ldots, r_n\}$ and $r_k \neq r_j$ $\neg(\exists u \in r_k$: $u - O \subset s - O)$. Since $r \notin R_2$, the set s was deleted either by one of the following applications of the operation μ, $\mu(\{r_1, \ldots, r_i\}, \{o_1, \ldots, o_i\}) = R'$ or $\mu(\{r_{i+1}, \ldots, r_n\}, \{o_{i+1}, \ldots, o_n\}) = R''$, or else s was deleted by $\mu(\{R', R''\}, \{o_1, \ldots, o_n\})$.

(a) Suppose that s was deleted while creating the set R' (if s was deleted while creating the set R'' the reasoning would be similar) this means that $r_j \in \{r_1, \ldots, r_i\}$. It also means that $\exists r_q \in \{r_1, \ldots, r_i\}$ such that $\exists p \in r_q$: $p - (o_1 \cup \cdots \cup o_i) \subset s - (o_1 \cup \cdots \cup o_i)$. Therefore $p - O \subset s - O$ which is a contradiction.

(b) Suppose that s was deleted while creating R_2, i.e., either $s - (o_1 \cup \cdots \cup o_i) \in R'$ or $s - (o_{i+1} \cup \cdots \cup o_n) \in R''$. In either case, it means that $\exists r_p \in \{r_1, \ldots, r_n\}$ $r_p \neq r_j$ and $\exists u \in r_p$: $(u - O \subset s - O)$ which is a contradiction.

Case 2. Suppose that $r \in R_2$ and that $r \not\in R_1$. Suppose furthermore that r was originated by a set $s \in r_j$, i.e., $r = s - O$. Since $r \not\in R_1$, this means that $\exists r_k \in \{r_1, \ldots, r_n\}$, $r_k \neq r_j$ and $\exists u \in r_k$: $(u - O \subset s - O)$. Since $r \in R_2$, it means that s was not deleted while creating the sets R' and R'', nor was it deleted while creating R_2. We will examine the consequences of each of these two cases:

(a) Suppose that both r_j and r_k belong to one of $\{r_1, \ldots, r_i\}$ or $\{r_{i+1}, \ldots, r_n\}$, say that they both belong to $\{r_1, \ldots, r_i\}$. Then, since s was not deleted while creating R' it means that $[u - (o_1 \cup \cdots \cup o_i)] \not\subset [s - (o_1 \cup \cdots \cup o_i)]$. Now, if u was not deleted by the application of μ which creates R', then both $u - O$ and $s - O$ will be considered while creating R_2 and $s - O$ will be deleted by the application of μ, which is a contradiction; if u is deleted while creating R', then $\exists r_q \in \{r_1, \ldots, r_i\}$ such that $\exists p \in r_q$: $p - (o_1 \cup \cdots \cup o_i) \subset u - (o_1 \cup \cdots \cup o_i)$, meaning that both $p - (o_1 \cup \cdots \cup o_i)$ and $s - (o_1 \cup \cdots \cup o_i)$ belong to R' and therefore s will be deleted by the application of μ which creates R_2, which is a contradiction.

(b) Suppose that r_j and r_k do not both belong to one of $\{r_1, \ldots, r_i\}$ and $\{r_{i+1}, \ldots, r_n\}$. Then s would be deleted by the application of μ which creates R_2, which is a contradiction.

Therefore $R_1 = R_2$. □

Lemma 2. *Given two supported wffs with the same OS, if their RS was obtained exclusively by successive applications of the μ operation, then the supported wffs have the same RS as well.*

Proof. It follows directly from Lemma 1. □

Theorem 5. *In the knowledge base resulting from the application of the rules of inference of the SWM system, if two supported wffs have the same OS, then they have the same RS as well.*

Proof. The proof will be done by complete induction on the number of applications of rules of inference.

The only supported wffs which can be obtained by applying one rule of inference only are hypotheses and since the rule of Hyp guarantees that their OS is unique the theorem is verified.

Suppose, by induction hypothesis, that all the supported wffs obtained by the application of less than n rules of inference verify the conditions of the theorem. We will have to show that the supported wff obtained by the application of the nth rule of inference also satisfies the statement of the theorem. We will group the rules of inference of SWM according to the type of OS and RS they produce and will discuss the OS and RS of the supported wff produced by the application of such type of rule.

(a) Creation of new OSs (Hyp). The rule of Hyp creates a supported wff with a new OS. The assumption behind the application of this rule is that there is no supported wff in the knowledge base with such OS and therefore this supported wff satisfies the conditions of the theorem.

(b) Change in RS only (URS). Upon detection of an inconsistent set, $\mathscr{X}$, this rule changes the RS of every supported wff whose OS is not disjoint from $\mathscr{X}$. Every supported wff $\langle F, t, o, R\rangle$ such that $\mathscr{X} \cap o \neq \emptyset$ is replaced by $\langle F, t, o, \sigma(R \cup \{\mathscr{X} - o\})\rangle$. If prior to the application of this rule all the supported wffs with the same OS had the same RS the same condition will be verified after the application of the rule since the RSs of supported wffs with the same OS are affected in the same way. An important point to note is that $\sigma(R \cup \{\mathscr{X} - o\}) = \mu(R \cup \{\mathscr{X} - o\}, o)$ and thus the RS of the resulting supported wffs are the same as if the μ operation had been applied.

(c) No change in OS nor RS ($\neg$E, $\wedge$I (part 1), $\wedge$E, $\vee$I, $\exists$I, $\exists$E). The application of one of these rules creates a new supported wff whose OS and RS are the same as the OS and RS of a previous supported wff, therefore, by the induction hypotheses the statement of the theorem is verified.

(d) The OS is the union of the OSs of the parent wffs (MP, MT, $\wedge$I (part 2), $\vee$E, $\forall$E). Using one of these rules, if we combine $\langle F_1, t_1, o_1, r_1\rangle$ and $\langle F_2, t_2, o_2, r_2\rangle$ we obtain $\langle F_3, t_3, o_1 \cup o_2, \mu(\{r_1, r_2\}, \{o_1, o_2\})\rangle$. Lemma 2 guarantees that, if only μ operations had been applied to form the RSs of the supported wffs in the knowledge base, the RS of $\langle F_3, t_3, o_1 \cup o_2, \mu(\{r_1, r_2\}, \{o_1, o_2\})\rangle$ is the same as the RS of any wff whose OS is $o_1 \cup o_2$.

(e) The resulting OS is one (or several) hypothesis short ($\rightarrow$I, $\neg$I, $\forall$I). Using one of these rules, we take one supported wff $\langle F_1, t_1, o, r\rangle$ and create a new supported wff $\langle F_2, t_2, o - \{H_1, \ldots, H_n\}, \int(o - \{H_1, \ldots, H_n\})\rangle$, recall that $\int(O) = \mu(\{r \mid \exists(H)\colon\ \mathbf{wff}(H) \in O \wedge \mathbf{ot}(H) = \text{hyp} \wedge r = \mathbf{rs}(H)\}, \ \{o \mid \exists(H)\colon\ \mathbf{wff}(H) \in O \wedge \mathbf{ot}(H) = \text{hyp} \wedge o = \mathbf{os}(H)\})$ and therefore Lemma 2 guarantees that, if only μ operations had been applied to form the RSs of the supported wffs in the knowledge base, the statement of the theorem is verified.

As a final remark it should be noticed that since the application of σ in (b) can be reduced to an application of μ, the RS of *every* supported wff in the knowledge base is created by successive applications of the μ operation and therefore if two supported wffs in the knowledge base have the same OS they also have the same RS. □

Corollary 5.1. *Every OS has recorded with it* every *known inconsistent set.*

Theorem 6. *Suppose that $C=\{H_1,\ldots,H_n\}$ is a context which is not known to be inconsistent. Then, for any two supported wffs, say $\langle A,t_a,o_a,r_a\rangle$ and $\langle B,t_b,o_b,r_b\rangle$, in the BS defined by the context C, we have* Combine($\langle A,t_a,o_a,r_a\rangle$, $\langle B,t_b,o_b,r_b\rangle$) = true.

Proof. Suppose by way of contradiction that the supported wffs $\langle A,t_a,o_a,r_a\rangle$ and $\langle B,t_b,o_b,r_b\rangle$ belong to the BS defined by the context C and that Combine($\langle A,t_a,o_a,r_a\rangle$, $\langle B,t_b,o_b,r_b\rangle$) = false. Since $\langle A,t_a,o_a,r_a\rangle$ and $\langle B,t_b,o_b,r_b\rangle$ belong to the BS defined by C we have $o_a\subset C$ and $o_b\subset C$. Since Combine($\langle A,t_a,o_a,r_a\rangle$, $\langle B,t_b,o_b,r_b\rangle$) = false, one of the following conditions holds:

(a) $\exists r\in$ **rs**($\langle A,t_a,o_a,r_a\rangle$): $r\subset$ **os**($\langle B,t_b,o_b,r_b\rangle$). Since **os**($\langle B,t_b,o_b,r_b\rangle$) $\subset\{H_1,\ldots,H_n\}$ we have that $r\subset\{H_1,\ldots,H_n\}$. By definition of restriction set we know that $(o_a\cup r)\vdash\rightarrow\leftarrow$. Since $o_a\subset\{H_1,\ldots,H_n\}$ and $r\subset\{H_1,\ldots,H_n\}$ we have that $(o_a\cup r)\subset\{H_1,\ldots,H_n\}$. Therefore, there exists a set $S\subset\{H_1,\ldots,H_n\}$ such that $S\vdash\rightarrow\leftarrow$. By Theorem 3 $\{H_1,\ldots,H_n\}\vdash\rightarrow\leftarrow$ which contradicts the assumption that C is not known to be inconsistent.

(b) $\exists r\in$ **rs**($\langle B,t_b,o_b,r_b\rangle$): $r\subset$ **os**($\langle A,t_a,o_a,r_a\rangle$). The same line of reasoning used in (a) will derive a contradiction.

Therefore Combine($\langle A,t_a,o_a,r_a\rangle$, $\langle B,t_b,o_b,r_b\rangle$) = true. □

Corollary 6.1. *If one uses a context which is not known to be inconsistent, then the system using SWM does not need to check for combinability between the wffs before the application of rules of inference.*

ACKNOWLEDGMENT

Many thanks to John Corcoran, Jon Doyle, Donald McKay, Ernesto Morgado, J. Terry Nutter, William J. Rapaport, and members of the SNePS Research Group for their criticisms and suggestions. We are grateful to an anonymous reviewer of this paper who shed light into the problems involved in belief revision, thus contributing to the presentation of this paper.

This work was partially supported by the National Science Foundation under Grant MCS80-06314 and by the Instituto Nacional de Investigação Científica (Portugal), under Grant No. 20536. Preparation of this paper was supported in part by the Air Force Systems Command, Rome Air Development Center, Griffiss Air Force Base, New York 13441-5700, and the Air Force Office of Scientific Research, Bolling AFB DC 20332 under contract No. F30602-85-C-0008, which supports the Northeast Artificial Intelligence Consortium.

REFERENCES

1. Anderson, A. and Belnap, N., *Entailment: The Logic of Relevance and Necessity* **1** (Princeton University Press, Princeton, NJ, 1975).
2. Campbell, S.S. and Shapiro, S.C., Using belief revision to detect faults in circuits, SNeRG Tech. Note No. 15, Department of Computer Science, State University of New York at Buffalo, Buffalo, NY, 1986.

3. Charniak, E., Riesbeck, C. and McDermott, D., *Artificial Intelligence Programming* (Erlbaum, Hillsdale, NJ, 1980).
4. Clayton, B.D., ART programming primer, Inference Corporation, Los Angeles, CA, 1984.
5. de Kleer, J., Choices without backtracking, in: *Proceedings AAAI-84*, Austin, TX (1984) 79–85.
6. de Kleer, J., An assumption-based TMS, *Artificial Intelligence* **28** (2) (1986) 127–162.
7. de Kleer, J., Problem solving with the ATMS, *Artificial Intelligence* **28** (2) (1986) 197–224.
8. de Kleer, J. and Doyle, J., Dependencies and assumptions, in: A. Barr and E.A. Feigenbaum (Eds.), *The Handbook of Artificial Intelligence* (Kaufmann, Los Altos, CA, 1982) 72–76.
9. de Kleer, J. and Williams, B., Back to backtracking: Controlling the ATMS, in: *Proceedings AAAI-86*, Philadelphia, PA (1986) 910–917.
10. de Kleer, J. and Williams, B., Reasoning about multiple faults, in: *Proceedings AAAI-86*, Philadelphia, PA (1986) 132–139.
11. Doyle, J., Truth maintenance systems for problem solving, Tech. Rept. AI-TR-419, AI Lab, MIT, Cambridge, MA, 1978.
12. Doyle, J., A truth maintenance system, *Artificial Intelligence* **12** (3) (1979) 231–272.
13. Doyle, J., The ins and outs of reason maintenance, in: *Proceedings IJCAI-83*, Karlsruhe, F.R.G. (1983) 349–351.
14. Doyle, J. and London, P., A selected descriptor-indexed bibliography to the literature of belief revision, *SIGART Newslett.* **71** (1980) 7–23.
15. Fitch, F., *Symbolic Logic: An Introduction* (Ronald Press, New York, 1952).
16. Goodwin, J.W., A improved algorithm for non-monotonic dependency net update, Research Rept. LiTH-MAT-R-82-23, Software Systems Research Center, Linkoping Institute of Technology, Linkoping, Sweden, 1982.
17. Goodwin, J.W., WATSON: A dependency directed inference system, in: *Proceedings Non-Monotonic Reasoning Workshop* (1984) 103–114.
18. Goodwin, J.W., *A Theory and System for Non-Monotonic Reasoning*, Linkoping Studies in Science and Technology, Dissertations, No. 165. Department of Computer and Information Science, Linkoping University, Linkoping, Sweden, 1987.
19. Haack, S., *Philosophy of Logics* (Cambridge University Press, Cambridge, U.K., 1978).
20. Hollan, J.D., Hutchins, E.L. and Weitzman, L., STEAMER: An interactive inspectable simulation-based training system, *AI Mag.* **5** (2) (1984) 15–27.
21. Kaplan, R., A multi-processing approach to natural language, in: *Proceedings National Computer Conference* (1973) 435–440.
22. Lemmon, E.J., *Beginning Logic* (Hackett, Indianapolis, IN, 1978).
23. London, P., Dependency networks as representation for modelling in general problem solvers, Ph.D. Dissertation, Tech. Rept. 698, Department of Computer Science, University of Maryland, College Park, MD, 1978.
24. Maida, A. and Shapiro, S., Intensional concepts in propositional semantic networks, *Cognitive Sci.* **6** (4) (1982) 291–330.
25. Martins, J., Reasoning in multiple belief spaces, Ph.D. Dissertation, Tech. Rept. 203, Department of Computer Science, State University of New York at Buffalo, Buffalo, NY, 1983.
26. Martins, J., Belief revision, in: S. Shapiro (Ed.), *Encyclopedia of Artificial Intelligence* (Wiley, and Sons, New York, 1987) 58–62.
27. Martins, J. and Cravo, M.R., Revising beliefs through communicating critics, to appear.
28. Martins, J. and Shapiro, S., Reasoning in multiple belief spaces, in: *Proceedings IJCAI-83*, Karlsruhe, F.R.G. (1983) 370–373.
29. Martins, J. and Shapiro, S., A model for belief revision, in: *Proceedings Non-Monotonic Reasoning Workshop* (1984) 241–294.
30. Martins, J. and Shapiro, S., Theoretical foundations for belief revision, in: *Proceedings of the*

1986 Conference on Theoretical Aspects of Reasoning about Knowledge (Morgan Kaufmann, Los Altos, CA, 1986) 383-398.
31. McAllester, D., A three valued truth maintenance system, AI Memo 473, AI Lab, MIT, Cambridge, MA, 1978.
32. McAllester, D., An outlook on truth maintenance, AI Memo 551, AI Lab, MIT, Cambridge, MA, 1980.
33. McAllester, D., A widely used truth maintenance system, Unpublished, 1985.
34. McDermott, D., Contexts and data dependencies: A synthesis, *IEEE Trans. Pattern Anal. Mach. Intell.* **5** (3) (1983) 237–246.
35. McDermott, D., Data dependencies on inequalities, in: *Proceedings AAAI-83*, Washington, DC (1983) 266–269.
36. McKay, D. and Martins, J., Provisional SNePSLOG user's manual, SNeRG Tech. Note 4, Department of Computer Science, State University of New York at Buffalo, Buffalo, NY, 1981.
37. McKay, D. and Shapiro, S., MULTI—a LISP based multiprocessing system, in: *Proceedings 1980 LISP Conference* (1980) 29–37.
38. McKay, D. and Shapiro, S., Using active connection graphs for reasoning with recursive rules, in: *Proceedings IJCAI-81*, Vancouver, BC (1981) 368–374.
39. Morris, P.H. and Nado, R.A., Representing actions with an assumption based truth maintenance system, in: *Proceedings AAAI-86*, Philadelphia, PA (1986) 13–17.
40. Nardi, B.A. and Paulson, E.A., Multiple worlds with truth maintenance in AI applications, in: *Proceedings ECAI-86*, Brighton, U.K. (1986) 437–444.
41. NEURON, Nexpert—some advanced features, Neuron Data, Palo Alto, CA, 1986.
42. Petrie, C.J., New algorithms for dependency-directed backtracking, Tech. Rept. AI TR86-33, Artificial Intelligence Laboratory, The University of Texas at Austin, Austin, TX, 1986.
43. Rich, C., Knowledge representation languages and predicate calculus: How to have your cake and eat it too, in: *Proceedings AAAI-82*, Pittsburgh, PA (1982) 193–196.
44. Saks, V., A matcher for intensional semantic networks, SNeRG Tech. Note No. 12, Department of Computer Science, State University of New York at Buffalo, Buffalo, NY, 1985.
45. Shapiro, S., Representing and locating deduction rules in a semantic network, in: *Proceedings Workshop on Pattern-Directed Inference Systems, SIGART Newslett.* **63** (1977) 14–18.
46. Shapiro, S., The SNePS semantic network processing system, in: Findler (Ed.), *Associative Networks: Representation and Use of Knowledge by Computers* (Academic Press, New York, 1979) 179–203.
47. Shapiro, S., Using non-standard connectives and quantifiers for representing deduction rules in a semantic network, Presented at "Current Aspects of AI Research", Seminar held at the electrotechnical Laboratory, Tokyo, 1979.
48. Shapiro, S., Martins, J. and McKay, D., Bi-directional inference, in: *Proceedings Fourth Annual Conference of the Cognitive Science Society* (1982) 90–93.
49. Shapiro, S. and Rapaport, W., SNePS considered as a fully intensional propositional semantic network, in: McCalla and Cercone (Eds.), *The Knowledge Frontier* (Springer, New York, 1987) 262–315.
50. Shapiro, S. and Wand, M., The relevance of relevance, Tech. Rept. No. 46, Computer Science Department, Indiana University, Bloomington, IN, 1976.
51. Shrobe, E., Dependency-directed reasoning in the analysis of programs which modify complex data structures, in: *Proceedings IJCAI-79*, Tokyo (1979) 829–835.
52. SST, DUCK: A versatile logic programming language for building intelligent systems, Smart Systems Technology, McLean, VA, 1984.
53. Stallman, R.M. and Sussman, G.J., Forward reasoning and dependency-directed backtracking in a system for computer-aided circuit analysis, *Artificial Intelligence* **9** (2) (1977) 135–196.
54. Summers, G., *Test Your Logic* (Dover, New York, 1972).

55. Thompson, A., Network truth-maintenance for deduction and modelling, in: *Proceedings IJCAI-79*, Tokyo (1979) 877–879.
56. Wand, M., The frame model of computation, Tech. Rept. No. 20, Computer Science Department, Indiana University, Bloomington, IN, 1974.
57. Winograd, T., Extended inference modes in reasoning by computer systems, *Artificial Intelligence* **13** (1–2) (1980) 5–26.
58. Winslett, M., Is belief revision harder than you thought?, in: *Proceedings AAAI-86*, Philadelphia, PA (1986) 421–427.

Received August 1985; revised version received July 1987