Available online at www.sciencedirect.com

ScienceDirect

Artificial Intelligence 172 (2008) 1045–1063

Artificial Intelligence

www.elsevier.com/locate/artint

# Active logic semantics for a single agent in a static world

Michael L. Anderson [a,d,*], Walid Gomaa [b,e], John Grant [b,c], Don Perlis [a,b]

[a] Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, USA
[b] Department of Computer Science, University of Maryland, College Park, MD 20742, USA
[c] Department of Mathematics, Towson University, Towson, MD 21252, USA
[d] Department of Psychology, Franklin & Marshall College, Lancaster, PA 17604, USA
[e] Department of Computer and Systems Engineering, Alexandria University, Alexandria, Egypt

Received 7 December 2006; received in revised form 14 November 2007; accepted 16 November 2007

Available online 22 November 2007

**Abstract**

For some time we have been developing, and have had significant practical success with, a time-sensitive, contradiction-tolerant logical reasoning engine called the active logic machine (ALMA). The current paper details a semantics for a general version of the underlying logical formalism, active logic. Central to active logic are special rules controlling the inheritance of beliefs in general (and of beliefs about the current time in particular), very tight controls on what can be derived from direct contradictions ($P\&\neg P$), and mechanisms allowing an agent to represent and reason about its own beliefs and past reasoning. Furthermore, inspired by the notion that until an agent *notices* that a set of beliefs is contradictory, that set *seems* consistent (and the agent therefore reasons with it as if it *were* consistent), we introduce an "apperception function" that represents an agent's limited awareness of its own beliefs, and serves to modify inconsistent belief sets so as to yield consistent sets. Using these ideas, we introduce a new definition of logical consequence in the context of active logic, as well as a new definition of soundness such that, when reasoning with consistent premises, all classically sound rules remain sound in our new sense. However, not *everything* that is classically sound remains sound in our sense, for by classical definitions, all rules with contradictory premises are vacuously sound, whereas in active logic not everything follows from a contradiction.

*Keywords:* Logic; Active logic; Nonmonotonic logic; Paraconsistent logic; Semantics; Soundness; Brittleness; Autonomous agents; Time

## 1. Introduction

Real agents have some important characteristics that we need to take into account when thinking about how they might actually reason logically: (a) their reasoning takes time, meaning that agents always have only a limited, evolving awareness of the consequences of their own beliefs,[1] and (b) their knowledge is imperfect, meaning that some of their beliefs will need to be modified or retracted, and they will inevitably face direct contradictions and other in-

[*] Corresponding author at: Department of Psychology, Franklin & Marshall College, P.O. Box 3003, Lancaster, PA 17604-3003, USA.
*E-mail addresses:* michael.anderson@fandm.edu (M.L. Anderson), wgomaa@alex.edu.eg (W. Gomaa), jgrant@towson.edu (J. Grant), perlis@cs.umd.edu (D. Perlis).

[1] Levesque's distinction between *explicit* and *implicit* beliefs [29] points to this same issue; however, our approach is precisely to model the evolving awareness itself, rather than trying to model the full set of (implicit) consequences of a given belief set.

0004-3702/$ – see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.artint.2007.11.005

consistencies. Indeed, real agents will not only often find their beliefs contradicted by experience, but will sometimes find that their beliefs have been internally inconsistent for some time, although they are only now in a position to notice this inconsistency, having derived a certain set of consequences that makes it apparent. The challenge from the standpoint of classical logical formalisms is that, if an agent's knowledge base can be inconsistent, then according to classical logic, it is permissible to derive *any* formula from it.

This fact about classical logics is commonly known by the Latin phrase *ex contradictione quodlibet*: from a contradiction everything follows. However, Graham Priest has coined the somewhat more vivid term *explosive logics*: a logic is explosive iff for all formulas $A$ and $B$, $(A\&\neg A) \models B$. Priest defines a paraconsistent logic precisely as one which is not explosive [40–42]. Now, clearly real agents cannot tolerate the promiscuity of belief resulting from explosive logics, and must somehow maintain control over their reasoning, watching for and dealing with contradictions as they arise. The reasoning of real agents, that is, must be paraconsistent. But what *sort* of paraconsistent logic might agents usefully employ, what methods might agents use to control inference and deal with contradictions, and how can these logics (and methods) be modeled in terms of truth and consequence in structures?

In the current paper we are primarily interested in the last of these questions. For some time we have been developing, and have had significant practical success with a time-sensitive, contradiction-tolerant logical reasoning engine called the active logic machine (ALMA) [46]. Because ALMA was designed with the above challenges in mind, its underlying formalism, active logic [17,18,33,34], includes special rules controlling the inheritance of beliefs in general (and of beliefs about the current time in particular), very tight controls on what can be derived from direct contradictions ($P\&\neg P$), and mechanisms allowing an agent to represent and reason about its own beliefs and past reasoning.

Here we offer a semantics for a general version of active logic. We hope and expect it will be of interest as a specific model of formal reasoning for real-world agents that have to face both the relentlessness of time, and the inevitability of contradictions.

In Sections 2–6 we will introduce the formal semantics for active logic, discuss a new definition of the consequence relation, and give examples of sound and unsound active logic inferences. This will be followed by some more informal discussion of the various properties of active logic (Section 7), a comparison of active logic with related approaches (Section 8), and a discussion of the practical issues involved with the use of active logic in real-world agents (Sections 9 and 10).

## 2. A semantics for real-world reasoning

In this section we propose a semantics for a time-sensitive, contradiction-tolerant reasoning formalism, incorporating the basic features of active logic.

### 2.1. Starting assumptions

In order to make the problem tractable for our first specification of the semantics, we will work under the following assumptions concerning the agent, the world (i.e., everything apart from the agent), and their interactions:

- There is only one agent $a$.
- The agent starts its life at time $t = 0$ $(t \in N)$ and runs indefinitely.
- The world is stationary for $t \geqslant 0$. Thus, changes occur only in the beliefs of the agent $a$.

Given these assumptions, there is one and only one true complete theory of the world; however, given that the agent's beliefs evolve over time, there is a different true complete theory of the agent for each time $t$.

### 2.2. The language $\mathcal{L}$

In order to express theories about such an agent-and-world, we define a sorted first-order language $\mathcal{L}$. We define it in two parts: the language $\mathcal{L}_w$, a propositional language in which will be expressed facts about the world, and the language $\mathcal{L}_a$, a first-order language used to express facts about the agent, including the agent's beliefs, for instance that the agent's time is now $t$, that the agent believes $P$, or that the agent discovered a contradiction in its beliefs at a given time. We write $Sn_K$ to mean the set of sentences of any language $K$. We are using the complete set of connectives $\{\neg, \rightarrow\}$ from which other connectives, such as $\wedge$, and $\vee$, can be derived. We assume that double negations are removed from formulas. For the sentence symbols the subscripts are used to indicate different propositional sentences and, for a fixed subscript, the superscripts are used to indicate different apperceptions (see Section 4) of the agent of the same proposition. The superscript 0 is used for the original sentence symbol (without superscript).

**Definition 1.** Let $\mathcal{L}_w$ be a propositional language consisting of the following symbols:

- a set $S$ of sentence symbols (propositional or sentential variables) $S = \{S_i^j\colon\ i, j \in N\}$ ($N$ is the set of natural numbers)
- the propositional connectives $\neg$ and $\rightarrow$
- left and right parentheses ( and )

$Sn_{\mathcal{L}_w}$ is the set of sentences of $\mathcal{L}_w$ formed in the usual way. These represent the propositional beliefs of the agent about the world. For instance $S_1^0$ might mean "John is happy". For later use we assume there is a fixed lexicographic ordering for the sentences in $Sn_{\mathcal{L}_w}$.

**Definition 2.** Let $\sigma, \theta \in Sn_{\mathcal{L}_w}$. We say that $\{\sigma, \theta\}$ is a *direct contradiction* if one of the following holds: either $\theta$ is the formula $\sigma$ preceded by a negation, or $\sigma$ is the formula $\theta$ preceded by a negation, that is $\theta = \neg\sigma$ or $\sigma = \neg\theta$.

Before giving the definition of the language $\mathcal{L}_a$, we remark the following:

i. In its current version $\mathcal{L}_a$ is a restricted form of first-order logic that is essentially propositional. In future work we intend to extend it to the full power of first-order logic.
ii. $\mathcal{L}_a$ contains a belief predicate that captures the fact that the agent believed a certain proposition at some time $t$. We allow for sentences of the form: at time $s$ the agent believed that she believed that she .... To allow for this indefinite (however finite) nesting, the definition of $\mathcal{L}_a$ has to be inductive where at stage $n + 1$ all sentences from the previous levels are captured for the belief predicate.

**Definition 3.** The language $\mathcal{L}_a$ is a sorted restricted version of first-order logic having three sorts:

- $\mathcal{S}_1$ is used to represent $Sn_{\mathcal{L}_w}$.
- $\mathcal{S}_2$ is used to represent time.
- $\mathcal{S}_3$ is used to represent $Sn_{\mathcal{L}}$ at its various stages of construction as shown below.

$\mathcal{L}_a$ will be defined as the union of a sequence of languages $\{\mathcal{L}_n\}_{n\in N}$ which are defined as follows:

- $n = 0$: $\mathcal{L}_0$ is a restricted first-order sorted language that does not contain variables or quantifiers, and consists of the following symbols:
  - the propositional connective $\neg$
  - a set of constant symbols $C = \{i\colon\ i \in N\}$ of sort $\mathcal{S}_2$ (this represents the time indices)
  - a set of constant symbols $D = \{\sigma\colon\ \sigma \in Sn_{\mathcal{L}_w}\}$, each is of sort $\mathcal{S}_1$ (here for simplicity, the constant symbols used to represent $Sn_{\mathcal{L}_w}$ are the sentences themselves)
  - a set of constant symbols $E_0 = \{\theta\colon\ \theta \in Sn_{\mathcal{L}_w}\}$, each is of sort $\mathcal{S}_3$ (again for simplicity the sentences themselves are used as constant symbols)
  - the unary predicate symbol *Now* of sort $\mathcal{S}_2$
  - the binary predicate symbol *Contra* of sort $(\mathcal{S}_1 \times \mathcal{S}_2)$.
- $n \geqslant 1$: Assume that $\mathcal{L}_m$ has already been defined for all $0 \leqslant m < n$. $\mathcal{L}_n$ is a restricted first-order sorted language that does not contain variables or quantifiers. In addition to the symbols of $\mathcal{L}_{n-1}$, it contains a set of constant symbols $E_n = \{\theta\colon\ \theta \in Sn_{\mathcal{L}_{n-1}}\}$ of sort $\mathcal{S}_3$. Also, $\mathcal{L}_1$ (and hence all $\mathcal{L}_i$, $1 \leqslant i$) contains a binary predicate symbol *Bel* of sort $(\mathcal{S}_3 \times \mathcal{S}_2)$.

Let $E = \bigcup_{i \in N} E_i$. Let the language $\mathcal{L}_a = \bigcup_{n \in N} \mathcal{L}_n$ so $Sn_{\mathcal{L}_a} = \bigcup_{n \in N} Sn_{\mathcal{L}_n}$. $C$, $D$, $E$ are sets of constant symbols. All of these constant symbols are terms in the language $\mathcal{L}_a$.

The sort $\mathcal{S}_2$ stands for time. In $\mathcal{L}_a$, *Now* is used to express the agent's time, *Contra* is used to indicate the existence of a direct contradiction in its beliefs, and *Bel* expresses the fact that the agent had a particular belief at a given time. We use these predicate symbols because they are crucial for active logic. The semantics for these predicates will be defined formally in Definition 8. Note that $\mathcal{L}_a$ contains only the connective $\neg$; hence statements such as $Bel(\theta, t) \rightarrow Bel(\theta, t+1)$ are not in the language.

However, we do not specify one specific set of active logic rules. This means that the semantics we will specify is applicable to a class of active logics with different rules that share a few common features.

**Definition 4.** Let $\mathcal{L} = \mathcal{L}_a \cup \mathcal{L}_w$, in the sense that $Sn_{\mathcal{L}} = Sn_{\mathcal{L}_a} \cup Sn_{\mathcal{L}_w}$.

**Definition 5.** Let the agent's knowledge base at time $t$, $KB_t$, be a finite set of sentences from $\mathcal{L}$, that is, $KB_t \subseteq Sn_{\mathcal{L}}$. In the case of $KB_0$ the only formulas of $Sn_{\mathcal{L}_w}$ we allow are those whose superscripts are all 0.

We can imagine $KB_0$ containing any sentences from $\mathcal{L}$ with which a system designer would equip an agent. $KB_0$ may or may not be consistent (no system designer is perfect!). After time 0, each $KB_t$ can be different from $KB_0$ because of inference, observation, forgetting, and the like. Also, although all the sentences about the world initially have superscript 0 for the sentence symbols, as we will see later the agent may assign different superscripts to the sentence symbols thereby changing a possibly inconsistent set to one that is consistent.

### *2.3. The semantics of $\mathcal{L}_w$*

In the following several definitions, we define the semantics of the formalism given above, in the standard way.

**Definition 6.** An $\mathcal{L}_w$-truth assignment is a function $h : S \rightarrow \{T, F\}$ defined over the set $S$ of sentence symbols in $\mathcal{L}_w$.

**Definition 7.** An $\mathcal{L}_w$ interpretation $h$ (we keep the same notation for this induced interpretation) is a function $h : Sn_{\mathcal{L}_w} \rightarrow \{T, F\}$ over $Sn_{\mathcal{L}_w}$ that extends an $\mathcal{L}_w$-truth assignment $h$ as follows:

$$h(\neg\varphi) = T \iff h(\varphi) = F$$

$$h(\varphi \rightarrow \psi) = F \iff (h(\varphi) = T \text{ and } h(\psi) = F)$$

We also stipulate a standard definition of consistency for $\mathcal{L}_w$: a set of $\mathcal{L}_w$ sentences is *consistent* iff there is some interpretation $h$ in which all the sentences are true. Notationally we write the usual $h \models \Sigma$, to mean all the sentences of $\Sigma$ are assigned $T$ by $h$.

We call $W_t$ the set of all $\mathcal{L}$-expressible facts about the external world. Thus, $W_t$ is consistent at every time $t$. This means that for every $t$ there exists an $\mathcal{L}_w$-truth assignment function $h_t$, such that $h_t \models W_t$. We also call the induced interpretation $h_t$, keeping the same notation. This result does not depend upon the assumption that the world is stationary. In a stationary world a single $h_t$ will work for all $t$; in a changing world there may be a *different* $h_t$, but there will still be *some* $h_t$, for each $t$. For a brief discussion of how we intend to approach the issue of a changing world in future work, see Section 11.

This does not mean that the agent's *beliefs* about the world are all true, consistent and constant (indeed, we expect they will contain contradictions and change over time), only that there is *some* set of true and consistent sentences that describe the world for every $t$. We'll detail later how to interpret and model the agent's world-knowledge (this being the crux of the issue when dealing with inconsistency). First, however, we turn to a model of the agent's meta-knowledge.

## 3. A model of the agent's $\mathcal{L}_a$ beliefs

First of all it is important to note that, even in the case where the agent's beliefs are incomplete, incorrect, or inconsistent, there is always a complete and consistent theory *of* those beliefs at the meta level, and this theory can be expressed using the language $\mathcal{L}_a$. For instance, if the agent believes both that John is happy ($S_1^0$) and that John is not happy ($\neg S_1^0$), the two sentences "the agent believes that John is happy" ($Bel(S_1^0)$) and "the agent believes that John is not happy" ($Bel(\neg S_1^0)$) can both be true at the same time.

Now we define an interpretation that models the theory about the agent at the meta level. In what follows, $\Sigma$ is to be understood formally as any set of sentences from $\mathcal{L}$; typically we will assume it to be some subset of the agent's knowledge base, combining beliefs about the world and about the agent, at some time $t$. Our point of view is that at any time $t$ the agent may deduce new beliefs from its knowledge base at time $t-1$, may add new sentences, for example from observations, or delete some sentences.

The following definition consists of ten bullet points: the first identifies the domain, the next three provide the interpretations for the three sorts; the following three provide the interpretations for all the constant symbols; the last three provide the interpretations for the three predicate symbols. *Now* keeps track of the time, and indicates the current time of the agent's internal clock. *Contra* indicates the existence of a direct contradiction in $\Sigma$ at some time $s \leqslant t$. *Bel* has the rough meaning "believes that", and states that a given sentence from $\mathcal{L}$ was in the agent's *KB* at some time $s \leqslant t$. We define the interpretation $H_{t+1}^{\Sigma}$ (at time $t+1$ based on $\Sigma$) modeling facts about the agent as follows.

**Definition 8.** $H_{t+1}^{\Sigma}$ is defined as the following interpretation:

- $Domain(H_{t+1}^{\Sigma}) = N \cup Sn_{\mathcal{L}}$.
- $H_{t+1}^{\Sigma}(\mathcal{S}_1) = Sn_{\mathcal{L}_w}$ (the set of propositions about the world).
- $H_{t+1}^{\Sigma}(\mathcal{S}_2) = N$ (all non-negative integers).
- $H_{t+1}^{\Sigma}(\mathcal{S}_3) = Sn_{\mathcal{L}}$ (the set of sentences representing the agent's knowledge base).
- $\forall n \in C$: $H_{t+1}^{\Sigma}(n) = n$.
- $\forall \sigma \in D$: $H_{t+1}^{\Sigma}(\sigma) = \sigma$.
- $\forall \theta \in E$: $H_{t+1}^{\Sigma}(\theta) = \theta$.
- The predicate symbol *Now* has the following semantics: $H_{t+1}^{\Sigma} \models Now(s) \iff s = t+1$ and $Now(t) \in \Sigma$; otherwise $H_{t+1}^{\Sigma} \models \neg Now(s)$.
- The predicate symbol *Contra* has the following semantics: $H_{t+1}^{\Sigma} \models Contra(\sigma, s) \iff$ either $s < t$ and $Contra(\sigma, s) \in \Sigma$ or $s = t$ and $\exists \sigma, \neg\sigma \in \Sigma$; otherwise $H_{t+1}^{\Sigma} \models \neg Contra(\sigma, s)$.
- The predicate symbol *Bel* has the following semantics: $H_{t+1}^{\Sigma} \models Bel(\theta, s) \iff$ either $s < t$ and $Bel(\theta, s) \in \Sigma$ or $s = t$ and $\theta \in \Sigma$; otherwise $H_{t+1}^{\Sigma} \models \neg Bel(\theta, s)$.

## 4. A model of the agent's $\mathcal{L}_w$ beliefs

Now we turn to the challenging problem of how to model, at the object level, the agent's beliefs about the world, given that these beliefs are not just evolving from moment to moment, but that at any given time, they may be inconsistent. Our scenario is as follows. At time 0 the agent has a finite set of initial beliefs, $KB_0$, about the world. All the sentence symbols have superscript 0. Then the agent starts to reason about the world using rules of active logic. This is where the agent may assign, via its apperception function, non-zero superscripts to some sentence symbols to avoid inconsistency. The agent may also obtain additional information about the world over time through other means, such as observations.

We will tackle this problem initially in three steps. First, we define a weak notion of consistency allowing for inconsistency in the agent's knowledge about the world; second, we will define a class of "apperception functions" intended to capture the intuition that an inconsistent *KB* will not necessarily *seem* inconsistent to the agent; and finally, we will show that there is some apperception function that, when applied to a given set of sentences, always produces a consistent set. Having shown this, we will proceed in the following sections to define a viable notion of active consequence, discuss the relation of this notion of consequence to the classical notion of logical consequence, and prove the soundness of some of the central inference rules of active logic.

Recalling that $\Sigma$ need not be consistent concerning facts about the world we define a weak version of consistency.

**Definition 9.** A set of sentences $\Sigma \subseteq Sn_{\mathcal{L}}$ is said to be $\mathcal{L}_a$ consistent iff $\Sigma \cap Sn_{\mathcal{L}_a}$ is classically consistent.

**Remark 1.** From now on, we will assume that $\Sigma$ is $\mathcal{L}_a$ consistent. We also introduce the symbol $\Gamma$ to refer to the potentially *inconsistent* set of $\mathcal{L}_w$ sentences in $\Sigma$: $\Gamma = \Sigma \cap Sn_{\mathcal{L}_w}$.

We next define an *apperception* (*self-awareness*) *function* for the agent. The notion of an apperception function is intended to help capture, at least roughly, how the world might seem to an agent with a given belief set $\Sigma$. For a real agent, only some logical consequences are believed at any given time, since it cannot manage to infer all the potentially infinitely many consequences in a finite time, let alone in the present moment. Moreover, even if the agent has contradictory beliefs, the agent still has a view of the world, and there will be limits on what the agent will and won't infer. This is in sharp distinction to the classical notion of a model, where (i) inconsistent beliefs are ruled out of bounds, since then there are no models, and (ii) all logical consequences of the *KB* are true in all models.

The task we are addressing, then, is that of finding a notion of semantics that avoids both (i) and (ii) above. Our idea—via apperception functions—is to suppose that an agent's limited resources apply also to its ability to inspect its own knowledge. Thus, if $S_i^0$ and $\neg S_i^0$ are both in $\Sigma$, the agent might not realize, at first, that the two instances of $S_i$ are in fact instances of the same sentence symbol. Thus it might seem to the agent that the world is one in which, say, $S_i^1$ is true, and so is $\neg S_i^2$. This allows the agent to have inconsistent beliefs while still having a consistent world model. Moreover, it allows us to see how an agent with inconsistent beliefs could avoid vacuously concluding *any* proposition, and also reason in a directed way, by applying inference rules only to an appropriately apperceived subset of its beliefs. We hope that this approach can shed some light on focused, step-wise, resource-bounded reasoning more generally.

An example of issue (i) might be Fred, who believes that if John is from the midwest then John is unhappy ($S_2^0 \to \neg S_1^0$), believes that John is from the midwest ($S_2^0$), and believes that John is happy ($S_1^0$). We represent the world view of such an agent by supposing that at least one of these beliefs is taken to have a different apparent meaning, one that is not inconsistent with the others (e.g. $S_2^0 \to \neg S_1^1$). This might happen because Fred hasn't thought carefully about all his beliefs, nor realized all of their consequences, and so never noticed that his beliefs entail both $S_1^0$ and $\neg S_1^0$. Note, however, that in our model, should Fred ever conclude $\neg S_1^0$ (or $\neg S_1^1$, from the apperceived version of the implication) he *will* recognize the contradiction at that time, and remove it (see below).

An example of issue (ii), although one currently beyond what our formalism can represent, might be Andrew Wiles working on a proof of Fermat's Last Theorem (FLT). He did not know, until he had completed his proof, that FLT was true. Yet he did have among his beliefs sufficient axioms to guarantee FLT as a consequence. So how did the world seem to him? Along the lines we are suggesting, he viewed some sentences as having possible interpretations different from what he later discovered to be the case. In effect, apperception functions, collectively, allow for a blurring of the identities, and hence meanings, of symbols.

The apperception functions we define can make changes only to $\Gamma$. An apperception function does not change $\Sigma - \Gamma$. We use the same notation *ap* when the apperception function is applied to an occurrence of a sentence symbol, a sentence, or a set of sentences. We start by defining a function that changes the superscripts of sentence symbols to 0. This is used to recover the original direct contradictions that were modified by the assignment of superscripts.

**Definition 10.** For any sentence $\phi \in Sn_{\mathcal{L}_w}$, let $z(\phi)$ be the sentence $\phi$ with all superscripts reset to 0. If $\Sigma \subseteq Sn_{\mathcal{L}_w}$, then $z(\Sigma) = \{z(\phi) | \phi \in \Sigma\}$.

**Definition 11.** An apperception *ap* is a function $ap: \Sigma \to \Sigma'$ where $\Sigma$ and $\Sigma'$ are sets of $\mathcal{L}$-sentences. An *ap* is represented as a finite sequence of non-negative integers: $\langle n_1, \dots, n_p \rangle$. The effect of *ap* on $\Sigma$ is as follows:

1. Let $\Sigma$ be a set of $\mathcal{L}$-sentences and let $\Gamma = \Sigma \cap \mathcal{L}_w$. Using the lexicographic order given earlier, let the $k$th sentence symbol in $\Gamma$ be $S_i^j$. The effect of the $ap = \langle n_1, \dots, n_p \rangle$ is to change $S_i^j$ to $S_i^{n_k}$ if $1 \leqslant k \leqslant p$, otherwise $S_i^j$ is unchanged.
2. $ap(\Sigma) = (\Sigma - \Gamma) \cup ap(\Gamma)$ (*ap* does not change $\Sigma - \Gamma$).

**Example 2.** Let $\Sigma = \{Now(5), Bel(S_2^0, 4), \neg S_2^1, S_2^1, S_1^0 \to S_5^4\}$. In this case $\Gamma = \{\neg S_2^1, S_2^1, S_1^0 \to S_5^4\}$. Writing the elements lexicographically yields $ord(\Gamma) = \{S_2^1, \neg S_2^1, S_1^0 \to S_5^4\}$. Consider $ap = \langle 1, 3, 2, 16, 7 \rangle$. Then $ap(\Sigma) = \{Now(5), Bel(S_2^0, 4), S_2^1, \neg S_2^3, S_1^2 \to S_5^{16}\}$.

Infinitely many apperception functions are needed because a finite set of sentences in $\mathcal{L}_w$ may have an arbitrarily large (finite) number of sentence symbols. However, if $\Gamma$ is known to contain $p$ occurrences of sentence symbols, then it suffices to deal only with apperception functions that are sequences of up to $p$ integers as the integers in the later locations are not applied. There are only finitely many such apperception functions.

The purpose of the apperception functions is to get rid of inconsistencies in $\Sigma$. Hence we are interested only in apperception functions that output consistent sets. The set of apperception functions that do this depends on $\Sigma$.

**Definition 12.** Let *AP* denote the class of all apperception functions. $AP^{\Sigma} = \{ap \in AP | ap(\Sigma) \text{ is consistent}\}$.

Next we show that $AP^{\Sigma}$ is never empty.

**Theorem 1.** *For all $\Sigma$, $AP^{\Sigma} \neq \emptyset$.*

**Proof.** Let *ap* assign a unique superscript to each occurrence of every sentence symbol in $\Gamma$. Then no sentence symbol appearing in $ap(\Gamma)$ is duplicated, hence each can be assigned a truth value independently. So $ap(\Gamma)$ is consistent. Since the remaining sentences in $\Sigma$ are consistent by assumption, and are in $\mathcal{L}_a$, $ap \in AP^{\Sigma}$. □

## 5. Active consequence

### 5.1. The definition of active consequence

At this point we are ready to define the notion of *active consequence* at time *t*—the active logic equivalent of logical consequence. We start by defining the concept of 1-*step active consequence* as a relationship between sets of sentences $\Sigma$ and $\Theta$ of $\mathcal{L}$, where $\Sigma \subseteq KB_t$ and $\Theta$ is a potential subset of $KB_{t+1}$. When we define this notion we want to make sure that $\Theta$ contains only sentences required by $\Sigma$ and the definition of $H_{t+1}^{\Sigma}$. This is the reason for the next definition.

**Definition 13.** Given $\Sigma$ and $ap \in AP^{\Sigma}$, define

$$dcs(\Gamma) = \{\phi \in \Gamma | \exists \psi \in \Gamma \text{ such that } z(\phi) = \neg z(\psi) \text{ or } \neg z(\phi) = z(\psi)\},$$
$$ap^{z}(\Gamma) = ap(\Gamma) - dcs(\Gamma).$$

The meaning of Definition 13 is that we are removing direct contradictions from $ap(\Gamma)$ while ignoring the superscripts.

**Definition 14.** Let $\Sigma, \Theta \subseteq Sn_{\mathcal{L}}$. Then $\Theta$ is said to be a 1-*step active consequence* of $\Sigma$ at time *t*, written $\Sigma \models_1 \Theta$ if and only if $\exists ap \in AP^{\Sigma}$ such that

i. if $\sigma \in \Theta \cap Sn_{\mathcal{L}_w}$ then $ap^{z}(\Gamma) \models \sigma$ ($\sigma$ is a classical logical consequence of $ap^{z}(\Gamma)$), and
ii. if $\sigma \in \Theta \cap Sn_{\mathcal{L}_a}$ then $H_{t+1}^{(\Sigma-\Gamma)\cup z(\Gamma)} \models \sigma$.

In this definition, for the sentences of $\Theta$ in the agent's language (at the meta level) 1-step active consequence depends on the interpretation $H_{t+1}$. Instead of $\Gamma$, we include $z(\Gamma)$ to capture all direct contradictions even if the superscripts have been changed. This also means that the *Bel* and *Contra* statements will contain sentence symbols only with superscript 0. For all the sentences of $\Theta$ expressing facts about the world, there must be some apperception function such that the apperception of $\Sigma$ (the $\mathcal{L}_w$ part) minus the direct contradictions classically implies these sentences. In the following we define the more general case of $n$-active consequence for any positive integer $n$ (similarly, as a result of this definition $\Theta$ is a potential part of $KB_{t+n}$).

**Definition 15.**

i. Let $\Sigma, \Theta \subseteq Sn_{\mathcal{L}}$. Then $\Theta$ is said to be an *n-step active consequence* of $\Sigma$ at time $t$, written $\Sigma \models_n \Theta$, if and only if

$$\exists \Delta \subseteq Sn_{\mathcal{L}}\colon\ \Sigma \models_{n-1} \Delta \text{ and } \Delta \models_1 \Theta \tag{5.1}$$

ii. We say that $\Theta$ is an *active consequence* of $\Sigma$, written $\Sigma \models_a \Theta$, if and only if $\Sigma \models_n \Theta$ for some positive integer $n$.

Next we give some examples to illustrate the concept of active consequence.

**Example 3.**

i. Let $\Sigma = \{S_1^0, \neg S_1^0\}$ and $\Theta = \{Contra(S_1^0, t)\}$. Then $\Sigma \models_1 \Theta$.
ii. Let $\Sigma = \{Now(t), S_1^0, S_1^0 \to S_4^0, S_{12}^0\}$ and $\Theta = \{Now(t+1), S_4^0, S_{12}^0\}$. Let $ap \in AP^{\Sigma}$ be the identity function. It is easy to see that $\{S_4^0, S_{12}^0\}$ are logical consequences of $\{S_1^0, S_1^0 \to S_4^0, S_{12}^0\}$. Also by definition $H_{t+1}^{\Sigma} \models Now(t+1)$. Hence $\Sigma \models_1 \Theta$.
iii. Let $\Sigma, \Theta$ be as in the previous example with $Bel(S_5^0, t)$ added to $\Theta$. Since $S_5^i \notin \Sigma$ for any $i$, $H_{t+1} \not\models Bel(S_5^0, t)$, hence $\Sigma \not\models_1 \Theta$. Therefore, for any later time $t+k$ and $\Delta$ obtained by active consequence from $\Sigma$, $H_{t+k}^{\Delta} \not\models$ $Bel(S_5, t)$, so $\Sigma \not\models_a \Theta$.
iv. Let $\Sigma = \{Now(t)\}$ and $\Theta = \{Now(t+5)\}$. Then $H_{t+1}^{\Sigma} \not\models \Theta$. However, $H_{t+1}^{\Sigma} \models Now(t+1)$. So $\{Now(t)\} \models_1$ $\{Now(t+1)\}$, and we get $\{Now(t)\} \models_5 \{Now(t+5)\}$, so $\{Now(t)\} \models_a \{Now(t+5)\}$. Hence $\Sigma \models_a \Theta$.
v. Let $\Sigma = \{S_1^0, S_2^0, S_2^0 \to \neg S_1^0\}$ and $\Theta = \{Contra(S_1^0, t+1)\}$. We will see that $\Sigma \models_2 \Theta$. Let $\Delta = \{S_1^1, \neg S_1^2\}$. Then $\Sigma \models_1 \Delta$, through the apperception function $ap(\Sigma) = \{S_1^1, S_2^2, S_2^2 \to \neg S_1^2\}$. Then $\Delta \models_1 \Theta$ by the second part of the definition, regardless of the apperception function applied in this step.

Note that in Example 3.v, it is not the case that $\Sigma \models_1 \{Contra(S_1^0, t)\}$ even though the conditions for the later appearance of the relevant direct contradiction were already in place at time $t$. This underlines the fact that in active logic it can take time for consequences to appear in the *KB*. In the case of $\mathcal{L}_a$ sentences, this temporal aspect of the logic is regulated and enforced directly by the semantics. For $\mathcal{L}_w$ sentences, it is an artifact of the particular set of rules that a given active logic agent is equipped with (see Sections 9 and 10 for more discussion of this issue).

Thus, for instance, considering the types of rules in active logic, given a rule like: $\frac{t:\ \alpha,\alpha\to\varphi,\varphi\to\psi}{t+1:\quad \psi}$ an agent could infer $\psi$ in one step from the formulas given at time $t$; however an agent equipped only with a simple version of modus ponens, such as that given in Definition 22 (see Section 6.1) would take two time steps to conclude $\psi$ from the same formulas. Both rules would be sound in active logic (see Definition 16), but a given agent might not be equipped with both rules (see Section 10). Since our definition of 1-step active consequence for sentences in $\mathcal{L}_w$ is based on logical implication, it is at least as powerful any set of sound syntactical rules could be.

### *5.2. The relationship between active consequence and 1-step active consequence*

By our definition of active consequence, $\Sigma \models_1 \Theta$ implies $\Sigma \models_a \Theta$. We may wonder how much bigger $\Theta$ may be in the latter case. Consider first a very simple situation: $\Sigma = \{S_1^0\}$, $\Theta = \{Bel(S_1^0, t)\}$ and $\Theta' = \{Bel(Bel(S_1^0, t), t+1)\}$. Here we have $\Sigma \models_1 \Theta$ and $\Theta \models_1 \Theta'$, hence $\Sigma \models_2 \Theta'$. This illustrates that considering $\mathcal{L}_a$ there can be additional sentences for each n-step active consequence for each new value of n. We show that this not does not happen for sentences of $\mathcal{L}_w$.

**Theorem 2.** *Suppose $\Sigma$, $\Theta \subseteq Sn_{\mathcal{L}_w}$. Then $\Sigma \models_1 \Theta \Leftrightarrow \Sigma \models_a \Theta$.*

**Proof.** Since both $\Sigma$ and $\Theta$ are sentences in $\mathcal{L}_w$, it suffices to deal only with sentences in $\mathcal{L}_w$. The $\Rightarrow$ part follows from the definition of $\models_a$.

Going in the other direction assume that $\Sigma \models_a \Theta$. By Definition 15 there must be a positive integer $n$ such that $\Sigma \models_n \Theta$, and that means that there is a $\Delta \subset Sn_{\mathcal{L}_w}$ such that $\Sigma \models_{n-1} \Delta$ and $\Delta \models_1 \Theta$. We divide the proof into two cases depending on the consistency of $\Sigma$.

Suppose that $\Sigma$ is consistent. Consider what can happen in $n-1$ steps, that is, $\Sigma \models_{n-1} \Delta$ where $\phi \in \Delta$. Such a $\phi$ must have been obtained by $n-1$ applications of classical logical implication to $\Sigma$ except that we may also change sentence symbol superscripts through $n-1$ apperception functions, one at each step. The key observation here is that both the application of classical logical implication and the application of apperception functions are transitive operations. This means that whatever can be obtained by $n-1$ applications of logical implication can already be obtained by a single application of logical implication, and the same goes for apperception functions. Hence $\Sigma \models_1 \Delta$. Doing this process again, but using $\Delta \models_1 \Theta$, we obtain $\Sigma \models_1 \Theta$.

Suppose next that $\Sigma$ is not consistent. Then in the first step of the implication, that is, to get $\Sigma \models_1 \Delta$, an apperception function, *ap*, must have been applied to $\Sigma$ first, making $ap(\Sigma)$ consistent (and removing direct contradictions), only then is the 1-step active consequence determined. Thus $\Sigma \models_1 \Delta$ iff $ap(\Sigma) \models_1 \Delta$ for some $ap \in AP^{\Sigma}$, where $ap(\Sigma)$ is consistent. But then we are back at the previous case where $\Sigma$ was consistent (where now $ap(\Sigma)$ is consistent) and the result follows. □

Although we proved this result only for sentences in $\mathcal{L}_w$, the same proof works (restricted to sentences of $\mathcal{L}_w$) even if $\Sigma$ and $\Theta$ contain sentences in $\mathcal{L}_a$.

### 5.3. The relationship between classical logical consequence and active consequence

How does classical logical consequence compare to active logic consequence? For sentences in $Sn_{\mathcal{L}_a}$ the two are incomparable. For consider $\Sigma = \{Now(t)\}$. Clearly, $\Sigma \models \Sigma$, but $\Sigma \not\models_a \Sigma$ because $Now(t)$ will not be true at any time after $t$. Next consider $\Theta = \{Bel(Now(t), t)\}$. Then $\Sigma \not\models \Theta$ but $\Sigma \models_a \Theta$.

So for the comparison we restrict our attention to $Sn_{\mathcal{L}_w}$. In classical logic an inconsistent set of sentences logically implies every sentence, but that is not the case for active consequence. The interesting question is what happens if $\Sigma \subseteq Sn_{\mathcal{L}_w}$ is consistent. It seems reasonable to expect active consequence to behave just like logical consequence. Recalling our theorem from the previous subsection, it suffices to compare only $\models$ and $\models_1$ because in this case $\models_1$ and $\models_a$ give the same result.

Thus in the consistent case we might expect $\Sigma \models \Theta \Leftrightarrow \Sigma \models_a \Theta$. The first implication, $\Sigma \models \Theta \Rightarrow \Sigma \models_a \Theta$, holds because we can choose the apperception function to be the identity function. Intuitively the opposite implication should hold as well. For consider that every given set of consistent sentences has a certain definite set of conclusions (consequences)—call this the "*inferential power*" of the set. We would expect this same set in active logic to have no more inferential power than it has under classical logical consequence. For consider an apperception function that assigns a different number to every sentence symbol in $\Sigma = \{S_1^0, S_1^0 \to S_2^0\}$, e.g., turns it into $\Theta = \{S_1^1, S_1^2 \to S_2^3\}$. Now the sentence symbol $S_2$ can no longer be inferred for any superscript. But this also presents a problem for the reverse implication. For $\Sigma \models_a \Theta$ holds but $\Sigma \models \Theta$ does not. The equivalence holds, however, if we restrict all sentence symbols to have superscript 0.

**Theorem 3.** *Let $\Sigma, \Theta \subseteq Sn_{\mathcal{L}_w}$. If $\Sigma$ is consistent, $\Sigma = z(\Sigma)$, and $\Theta = z(\Theta)$, then $\Sigma \models \Theta \Leftrightarrow \Sigma \models_a \Theta$.*

**Proof.** By Theorem 2 it suffices to prove that $\Sigma \models \Theta \Leftrightarrow \Sigma \models_1 \Theta$. It follows from $\Sigma = z(\Sigma)$ and $\Theta = z(\Theta)$ that all superscripts of sentences must be 0. In the application of the definition of 1-step consequence, an apperception function must be used. Since the apperception function leaves all superscripts at 0, it must be the identity function, so 1-step active consequence is identical to logical consequence, that is, $\Sigma \models \Theta \Leftrightarrow \Sigma \models_1 \Theta$. □

In Section 2.2 we stated that our semantics does not presuppose any one specific set of active logic rules because it is applicable to many different active logic systems with different rules. This means that we cannot expect to obtain the kind of completeness theorem for this semantics that one might get for a single specific set of rules. However, it is clear that 1-step active consequence is very powerful for consistent sets of sentences. It encompasses any set of active logic rules for $Sn_{\mathcal{L}_w}$. In that sense it is the limiting case for all possible sets of such active logic rules and provides an approximation to a completeness result. In the following, we write $\vdash$ for derivability in active logic, instead of the vertical notation commonly used there. See the next section for the standard vertical notation.

**Theorem 4.** *Suppose that $\Sigma, \Theta \subseteq Sn_{\mathcal{L}_w}$, $\Sigma = z(\Sigma)$, $\Theta = z(\Theta)$, $\Sigma$ is consistent and $\Sigma$ and $\Theta$ are finite.*

(a) *Let $\vdash$ represent the derivability relation for any active logic. If $\Sigma \vdash \Theta$ then $\Sigma \models_a \Theta$.*
(b) *If $\Sigma \models_a \Theta$ then there is an active logic with derivability relation $\vdash$ such that $\Sigma \vdash \Theta$.*

**Proof.** (a) If $\Sigma \vdash \Theta$ then every $\phi \in \Theta$ must logically follow from $\Sigma$, hence $\Sigma \models_1 \Theta$, so $\Sigma \models_a \Theta$.

(b) If $\Sigma \models_a \Theta$ then for each $\phi \in \Theta$ introduce a (valid) active logic rule stating that $\Sigma \vdash \phi$. For the active logic defined by these rules (for $Sn_{\mathcal{L}_w}$), $\Sigma \vdash \Theta$. □

## 6. Sound and unsound inferences in active logic

At this point we consider possible inference rules for active logic. We start with some notes about the syntax of active logic rules. Because active logic is a step logic, we always precede both the antecedent and the consequent (which are divided by a horizontal line) with an indication of the time, thus:

$$\frac{t\colon \textit{antecedent}}{t+1\colon \textit{consequent}}$$

The antecedent can be any of the following:

- a single formula, e.g. $\theta$, or $Now(t)$
- any set of formulas separated by commas, e.g. $\theta, \theta \to \sigma$
- any set of formulas meeting some specified conditions, and represented by a single capital letter, with a semi-colon between the capital letter and the conditions e.g. $\Sigma; \theta \in \Sigma$
- any set of formulas representing the database of an agent at a specific time. This will be represented by $KB_t$, and may also specify conditions using the same convention as above.

The consequent can be any of the following:

- a single formula, e.g. $\theta$, or $Now(t)$
- any set of formulas separated by commas, e.g. $\theta, \theta \to \sigma$.

Now we define the notion of a-sound inference.

**Definition 16.** An active sound (*a-sound*) inference is one in which the consequent is a 1-step active consequence of the antecedent.

Recall that (1-step) active consequence is defined between sets of sentences. However, in accordance with the syntax defined above, we will omit the set notation symbols { and }.

### *6.1. Some active-sound inference rules*

For all six rules given here, a-soundness follows directly from the definitions. We prove the last as an illustration.

**Definition 17.** If $Now(t) \in KB_t$ then the *timing inference rule* is defined as follows:

$$\frac{t\colon Now(t)}{t+1\colon Now(t+1)}$$

**Definition 18.** If $\varphi, \neg\varphi \in KB_t$, where $\varphi \in Sn_{\mathcal{L}_w}$, then the *direct contradiction inference rule* is defined as follows:

$$\frac{t\colon \varphi, \neg\varphi}{t+1\colon Contra(\varphi, t)}$$

**Definition 19.** If $\varphi \in KB_t$, where $\varphi \in Sn_{\mathcal{L}}$, then the *positive introspection inference rule* is defined as follows:

$$\frac{t\colon \varphi}{t+1\colon Bel(\varphi, t)}$$

**Definition 20.** If $\varphi \notin KB_t$, where $\varphi \in Sn_{\mathcal{L}}$, then the *negative introspection inference rule* is defined as follows:

$$\frac{t\colon\ KB_t;\ \varphi \notin KB_t}{t+1\colon\ \neg Bel(\varphi, t)}$$

**Definition 21.** If $\varphi \in Sn_{\mathcal{L}}$ such that $\varphi \in KB_t$, $\neg\varphi \notin KB_t$, $\varphi \neq Now(t)$, and $\varphi$ is not a contradiction, then the *inheritance inference rule* is defined as follows:

$$\frac{t\colon\ \varphi}{t+1\colon\ \varphi}$$

**Definition 22.** Let $\Theta = \{\varphi, \varphi \to \psi\} \subseteq (KB_t \cap Sn_{\mathcal{L}_w})$ such that $\Theta$ is consistent. Assume $\neg\varphi \notin KB_t$ and $\neg(\varphi \to \psi) \notin KB_t$ (see Section 6.3 for more on this restriction), then the *active modus ponens inference rule* is defined as follows:

$$\frac{t\colon\ \varphi, \varphi \to \psi}{t+1\colon\ \psi}$$

**Theorem 5.** *The rules given in Definitions* 17–22 *are a-sound.*

For Definitions 17–20, their a-soundness follows from the definitions. By way of illustration, consider the following for active modus ponens (Definition 22):

**Proof.** Use an apperception function which is the identity on $\Theta$ and assigns a unique different superscript to any other symbol in $KB_t$. □

## 6.2. Active-unsound inference rules

We examine a number of instances of classically unsound inference rules, and get the expected intuitive results that these inferences are also active-unsound. In all cases $\varphi$ and $\psi$ are arbitrary sentences of $\mathcal{L}$.

**Definition 23.** We call this first rule the $\varphi$ *implies* $\psi$, or $\varphi \to \psi$ rule.

$$\frac{t\colon\ \varphi}{t+1\colon\ \psi}$$

**Theorem 6.** *The* $\varphi \to \psi$ *inference rule is not a-sound* (*is a-unsound*).

**Proof.** Let $\varphi = S_1^0$ and let $\psi = \neg(S_1^0 \to S_1^0)$. Then $\psi$ is not an active consequence of $\varphi$, because by Theorem 3, this would mean that $\psi$ classically follows from $\varphi$, and that is false. □

**Definition 24.** We call this next rule the $\varphi$ *implies not* $\varphi$, or $\varphi$-not-$\varphi$ rule: We assume that $\varphi$ is a consistent formula.

$$\frac{t\colon\ \varphi}{t+1\colon\ \neg\varphi}$$

**Theorem 7.** *The* $\varphi$*-not-*$\varphi$ *inference rule is a-unsound*

**Proof.** Let $\varphi = S_1^0$ and apply Theorem 3. □

Interestingly, although the $\varphi$-not-$\varphi$ inference rule is a-unsound in general (with respect to the big language $\mathcal{L}$), there is one special instance in which it *is* sound, namely:

$$\frac{t\colon\ Now(t)}{t+1\colon\ \neg Now(t)}$$

This further underlines the special status of time and the *Now*() predicate in active logic; this result would obviously not be classically sound.

However, one rule that is classically sound, but a-unsound, is the explosive rule. This shows that active logic is a *paraconsistent logic*, something we consider one of its advantages over classical formalisms.

**Definition 25.** Let $\Sigma \subseteq Sn_{\mathcal{L}_w}$ be inconsistent. Let $\psi \in Sn_{\mathcal{L}_w}$. We define the *explosive rule* with respect to the language $\mathcal{L}_w$ as follows.

$$\frac{t\colon\ \Sigma; \mathit{Inconsistent}(\Sigma)}{t+1\colon\ \psi}$$

**Theorem 8.** *The explosive inference rule is a-unsound.*

**Proof.** Let $\psi$ be $\neg(S_1^0 \to S_1^0)$. No apperception function *ap* that turns $\Sigma$ into a consistent set can logically derive $\psi$. Hence $ap(\Sigma) \not\models_1 \psi$. By Theorem 2 the result follows. □

*6.3. Inconsistent KBs, apperception functions and the application of a-sound rules*

We noted above that there can be no official catalog of rules for active logic; any a-sound rule can qualify, and a given active logic agent may be equipped with any number of these rules (see Section 9 for more on this). However, the fact that a-soundness is defined in terms active consequence, which is itself defined in terms of apperception functions, means that not every a-sound rule will be available for use in every situation. More specifically, whether or not there are direct contradictions in $\Sigma = KB_t$, the apperception function may change which rules can and cannot be applied for that $\Sigma$.[2] (We list only $\Gamma$ in the examples below.) Let $\Gamma = \{S_1^0, \neg S_1^0, S_1^0 \to S_2^0\}$. Because of the direct contradiction, the active modus ponens rule would not apply ($S_1^0$ and $\neg S_1^0$ would be removed from the *KB*).

Next, consider a case where active modus ponens does apply, namely, let $\Gamma = \{S_1^0, S_1^0 \to S_2^0, \neg S_2^0\}$. So we can derive $S_2^0$ by using an *ap* that only changes the superscript of $\neg S_2^0$. A different possible consequence is $\{S_2^0 \to S_3^0\}$, using an a-sound notational variant of the classically sound rule[3]

$$\frac{\neg\psi}{\psi \to \alpha}$$

and using an *ap* that only changes the superscript of $S_1^0$.

But note that the set $\{S_2^0, S_2^0 \to S_3^0\}$ is *not* an active consequence of $\Gamma$ because there is no *single* apperception function that would allow this set to be derived. Thus we cannot necessarily combine a-sound rules and guarantee that the result is an active consequence. (The problem exists only for rules involving $\mathcal{L}_w$.) This also underlines the fact once again that apperception functions can limit the inferential power of a given set of sentences. For a discussion of the practical effects of this limitation, see Sections 9 and 10.

This concludes the presentation of the active logic semantics. In the next two sections (7 and 8) we will discuss some of the general properties of active logic that follow from its semantics, and compare active logic to other related work. After that, in Sections 9 and 10, we will discuss some of the practical issues involved with using active logic in real-world reasoning agents.

## 7. General properties of active logic

One of the original motivations for active logic was the need to design formalisms for reasoning about an approaching deadline; for this use it is crucial that the reasoning take into account the ongoing passage of time as that reasoning proceeds. Thus, active logic reasons one step at a time, updating its belief about the current time at each step, using rules like the timing rule given in Definition 17.

[2] In fact, it is generally true of apperception functions that they will determine which rules are applicable in a given *KB* at a given time; however, in a consistent *KB*, there will always be an eligible apperception function that makes no alterations to the *KB*, thus not changing which rules apply. Thus, the remarks below are limited to the case of an inconsistent *KB*.

[3] While this rule *can* be written so as to be a-sound, it is rather a dangerous rule in a non-monotonic logic, and it would probably not be advisable to include it among the catalog of rules with which a practical active logic agent is equipped.

This step-wise, time-aware approach gives active logic fine control over what it does, and does not, derive and inherit at each step; for instance, $Now(t)$ is not inherited at time step $t + 1$. To "inherit" $P$ is, roughly speaking, to assert $P$ at time $t + 1$ just in case it was believed at time $t$. However, in a temporal, non-monotonic formalism, what is justified *now* may not be justified *later*. For a simple example, consider that a certain observation at time $t$ may justify the conclusion that it is raining at time $t$, and it may be reasonable to continue to believe this at time $t + 1$ (i.e. to inherit the belief). However, at some point in time, $t + n$, neither the original observation, nor the inherited belief can be considered justification for the continued belief that it is raining. Thus, although inheriting is a reasonable default behavior, there will be conditions and limits.[4] This is accomplished by special inheritance rules like Definition 21. Note in particular the conditions governing that rule, conditions that can be tailored for different agents and circumstances.

Such step-wise control over inference gives active logic the ability to explicitly track the individual steps of a deduction. Thus, for instance, an inference rule can refer to the results of all inferences *up until now*—i.e. through time $t$—as it computes the subsequent results (for time $t + 1$). This allows an active logic to reason, for example, about its own (past) reasoning; and in particular about what it has *not* yet concluded. Moreover, this can be performed quickly, since it involves little more than a lookup of the current knowledge base (see, e.g. Definition 20). Although the complexity of this operation is low—$O(n)$—it is nevertheless the case that if the *KB* is allowed to grow indefinitely, the operation will take increasing time. Currently beliefs older than some arbitrary threshold are removed from active memory and written to a searchable log file. However, we are investigating various more intelligent methods for selective "forgetting".

This last point is worth further elaboration and emphasis, for it is central to the active logic approach to modeling the reasoning of real-world agents. The reason that determining what one does not know—otherwise known as negative introspection—is simple in active logic is a direct result of the practical acknowledgment that any real agent is limited to reasoning only with whatever formulas (wffs) it has been able to come up with *so far*, rather than with implicit but not yet performed inferences. Thus, determining if a given formula $P$ is known is not a question of seeing if $P$ is a *consequence* of one's current beliefs, but only a question of seeing if $P$ is actually present in the *KB*. This approach is especially important to the issue of performing consistency checks before accepting new formulas into the *KB*. After all, before accepting $P$, one may well want to know whether $P$ is consistent with one's current beliefs. In general, $P$ is not consistent with the *KB* if $\neg P$ can be derived from *KB*. However, it is not in general possible to know, for any given formula if that formula is derivable from current beliefs, without actually going through the required deductions to *prove* it. That could take a great deal of time—more time than a typical agent will have before deciding to accept $P$. Cutting this process down to a simple *KB* look-up of $\neg P$, then, is an important practical simplification. So, instead of looking for arbitrary contradictions to P, we are only looking for *direct* contradictions (i.e. $\neg P$).[5]

But won't this practical simplification mean that active logic *KB*s are more likely to become inconsistent? That is certainly a possibility, and yet, insofar as (a) contradictions are an inevitable part of living in and reasoning about the real world, and (b) the consistency of complex *KB*s is practically impossible to determine or maintain, then it seems a better bet to focus less on maintaining consistency, and more on an ability to reason effectively in the presence of contradictions, taking action with respect to them only when they become revealed in the course of inference (which itself might be directed toward finding contradictions, to be sure).

This is where the other central features of active logic—its step-wise control over inference, and the built-in ability to refer to individual steps of reasoning—come into play, making active logic a natural formalism for detecting and reasoning about contradictions and their causes. For as soon as a contradiction reveals itself—that is, as soon as $P$ and $\neg P$ are both present in the KB—it is possible to "capture" it, preventing further reasoning using the contradictory formulas as premises (and thereby preventing any explosion of wffs), while at the same time marking their presence, to allow further consideration of the cause of the contradiction. Current implementations of active logic incorporate a "conflict-recognition" inference rule like Definition 18 for this purpose.

Through the use of such rules, *direct* contradictions can be recognized as soon as they occur, and further reasoning can be initiated to repair the contradiction, or at least to adopt a strategy with respect to it, such as simply avoiding the use of either of the contradictory formulas for the time being. Unlike in truth maintenance systems [15,16] where

[4] Inheritance and disinheritance are directly related to belief revision [23] and to the frame problem [11,31]; see [34] for further discussion.

[5] This discussion is not meant to imply that, if $\neg P$ is found in the KB, that the agent will necessarily, for that reason, reject $P$, for there may be good reason to reject $\neg P$, instead.

a separate process resolves contradictions using justification information, in active logic the contradiction detection and handling [32] occur in the same reasoning process. In fact, the *Contra* predicate is a meta-predicate: it is about the course of reasoning itself (and yet is also part of that same evolving history).

Thus, speaking somewhat more broadly, active logic is a paraconsistent logic that *achieves* its paraconsistency in virtue of possessing two simultaneously active (and interactive) modes of reasoning, which might be called *circumspective* and *literal*. In literal mode, the reasoning agent is simply working with, and deriving the consequences of, its current beliefs. In circumspective mode, the reasoning agent is reasoning *about* its beliefs, noting, for instance, that it has derived a contradiction, and deciding what to do about that. It is important to active logic that these are not separate, isolated modes, but interactive and part of the same overall reasoning process. Thus, for instance, the (circumspective) derivation of *Contra* is triggered by the (literal) derivation of $P$ and $\neg P$, and reasoning with *Contra* happens alongside reasoning about other matters. Likewise, reasoning about a contradiction may eventually result in the reinstatement of one of the conclusions, $P$ or $\neg P$, to be carried forward and reasoned with in literal mode. It is precisely this ongoing interaction between literal and circumspective modes, between reasoning and self-monitoring, that allows active logic to avoid the pitfalls of explosive logics, and makes it more appropriate to the needs of real-world agents.

## 8. Comparison with related work

Active logic is primarily related to two bodies of work—work on temporal logics, and work on paraconsistent logics. We will treat each of these subjects in turn.

### *8.1. Temporal logics*

Temporal logics—logical formalisms explicitly allowing for the representation of temporal information—were introduced by Prior (under the name of Tense Logic) in a series of writings between 1957 and 1969 [43–45]. Pnueli established the relevance of tense logic for understanding the runtime behavior of programs [38]. Such temporal logics are modal, with operators for notions such as the future truth of a predicate. A first-order approach to reasoning about time was employed by Allen [1], with expressions such as $\mathrm{Holds}(A, t)$ to mean $A$ is true at time $t$; Allen and others made major strides in the use of such formalisms (so-called action logics) in AI. Part of the effect of these latter efforts was to connect temporal logic to belief logics, i.e., logics for representing information about an agent that plans and acts in a dynamic world. Thus action logics typically have temporal aspects, since the passage of time is of central importance to the planning and carrying out of actions; see for instance [22].

Another central feature of most such logics is a treatment of the frame problem. Definition 21 (the inheritance rule) might be considered a kind of frame axiom. While it does not quite assert that $\phi$ remains true despite an action having occurred, it has a similar effect: it says that $\phi$ will remain believed unless there is a reason not to believe it, such as might happen if an action is known to have negated $\phi$.

Various logics of action and belief have been extensively studied for as long as AI has existed [28–31]. Typically, the formalism is designed to represent the formation of an agent's beliefs (including its beliefs about the results of actions) based on a starting set of information (initial beliefs, or axioms). However, since belief-formation in any real-world agent must occur as a process in time, it is natural to consider a logic in which not only is time represented (i.e., one is able to express things about time, as in a temporal logic) but also the *passage* of time is represented as an evolving process in which the "present" time moves forward during belief formation. Thus the agent has a certain set of beliefs "now", and another set at a later "now". But if the logic is to be used by the agent, then its own evolving notion of what time it is must be factored into the formalism as well. This is where active logics come in: an agent/temporal logic with a twist: an evolving now and corresponding time-sensitive inference rules.

Active logic is not the only formalism to consider time in this way. For instance, SNePS [50], especially as applied to the Embodied Cassie project [49], incorporates an indexical, evolving-time variable *NOW*. Cassie, a natural-language-using autonomous robot, uses this variable to track the passage of time, allowing it to do such things as appropriately alter verb tenses when discussing present or past actions. Cassie's temporal awareness also plays a role in time-sensitive planning projects like maintaining its battery and remediating unexploded land mines (in simulation).

The motivations for including such an evolving "now" in Cassie are quite similar to the motivations for including one in active logic. Ismail and Shapiro write: "[E]mbodied cognitive agents should ... act in and reason about a changing world, using reasoning in the service of acting and acting in the service of reasoning. Second, they should be able to communicate their beliefs, and report their past, ongoing, and future actions in natural language. This requires a representation of time ..." [27]. However, there are some significant differences in the nature of the "now" incorporated into each formalism, and how it can therefore be used.

Perhaps the biggest difference is that for the SNePS-based agent Cassie, *NOW* is a meta-logical variable, rather than a logical term fully integrated into the SNePS semantics. The variable *NOW* is implemented so that it does, indeed, change over time (and, in particular, changes whenever Cassie acts in any way, including by reasoning), but this change is the result of actions triggering an external time-variable update. In active logic, in contrast, reasoning itself *implies* the passage of time. Perhaps in part because of this difference, SNePS is a monotonic logic, whereas active logic is non-monotonic, leveraging the facts that beliefs are had at times, and beliefs can be had about beliefs, to easily represent such things as "I used to believe $P$, but now I believe $\neg P$" using the *Bel* operator. SNePS is also able to represent beliefs about beliefs, but there is no indication that this ability is leveraged by SNePS to guide belief updates. Rather, all Cassie's beliefs are about states holding over time, so that belief change is effected by allowing beliefs to expire, rather than by formally retracting them. This is a strategy similar to that employed by the situation calculus (which does not itself incorporate a changing *Now* term) [31]. Finally, although SNePS is a paraconsistent logic, it is so in virtue of the fact that contradictions imply nothing at all, whereas in active logic contradictions imply *Contra*, a meta-level operator that can trigger further reasoning.

### *8.2. Paraconsistent logics*

As mentioned in the introduction, the term *paraconsistent logic* is applied to logics that are not explosive. Another way to look at this concept is to consider that classical logic is so averse to inconsistency that it cannot distinguish between local inconsistency, where for some formula $A$ both $A$ and $\neg A$ hold, and global inconsistency, where for all formulas $A$ both $A$ and $\neg A$ hold. So in a paraconsistent logic, local inconsistency does not imply global inconsistency. For various reasons, including philosophical issues, the intrinsic interest of investigating paraconsistency, and particularly the increasing number of applications involving inconsistencies, there has been growing interest in this field, including several books, numerous papers, and three World Congresses on Paraconsistency: [5] and [12] are the Proceedings of the first two (see also [13] for a historical survey).

As noted in the survey paper [24], paraconsistency may be achieved in several different ways. Modifying the axioms or rules is one technique. Another method stays within the framework of classical logic by the use of maximal consistent subsets of formulas. Consider an inconsistent set of formulas $\Gamma$. There must always be some subsets of $\Gamma$ that are consistent (for example, $\emptyset$ is consistent) hence there must be maximal consistent subsets of $\Gamma$. In this method $A$ is deduced from $\Gamma$ if $A$ is deduced classically from all maximal consistent subsets of $\Gamma$ [48]. Some researchers use additional criteria to find preferred consistent subsets and work with those [8].

Another technique [7] extends the set of classical truth values from $\{\mathit{True}, \mathit{False}\}$ to a larger set. Usually, the set of truth values is given an algebraic structure, typically a lattice. Perhaps the best-known of these is the lattice $\mathit{FOUR} = \{\mathit{True}, \mathit{False}, \mathit{Both}, \mathit{Neither}\}$ where *Both* stands for an inconsistency. A fourth approach extends classical logic by the addition of modal or metalevel operators. Modal logic has an operator for a formula to be possible (true in some world) and necessary (true in all worlds) where worlds are selected in some way. Both a formula $A$ and its negation $\neg A$ may be possible because they are true in different worlds, but that does not mean that all formulas are possible.

Consider now how active logic fits into the classification given above. In active logic the rules of inference are limited, and are based on the passage of time. Also the language contains the meta-level operator *Contra* to indicate contradictory statements. Hence active logic combines two of the methods above to achieve paraconsistency.

### *8.3. Other related work*

Several other interesting frameworks exist that encompass many logical systems in a uniform manner. We briefly discuss two such frameworks here.

A Labelled Deductive System (LDS) [20] is a logical reasoning system employing both formulas and annotations for those formulas, called labels. The labels can have various contents with effects on the deductions. For instance, if the label indicates that one formula is better supported by evidence than another, then deductions using the better supported formula may be preferred over those using the other, especially in cases of conflict. Such an LDS would implement a non-monotonic logic with preferences or prioritized defaults. A group of LDSs called restricted access logics [21] deal specifically with inconsistent information. We mention LDS here because there is some evidence that the simple version of active logic $SL_7$, described in [17], can likewise be implemented by or described as an LDS [3]. This is an interesting finding, and although it is not clear that the version of active logic described here can likewise be expressed as an LDS, this may well turn out to be the case. Even so, active logic would be a special case of LDS with some interesting and valuable properties, such as non-monotonicity, paraconsistency, and temporal sensitivity.

Another interesting general framework is called adaptive logics. The adaptive logics that handle inconsistencies are called inconsistency-adaptive logics [6]. Adaptive logics are characterized by a lower limit logic, a set of abnormalities, and an adaptive strategy. The purpose of adaptive logics is to characterize inference relations $\vdash$ for which there is no positive test that for every $\Sigma$ and $\varphi$ will answer "yes" if $\Sigma \vdash \varphi$. Active logics are more well behaved and have such a positive test. In fact, as we will show in the next section, even the limiting use of $\models_a$ has such a test in our case where the logic is essentially propositional.

## 9. The ideal active logic agent

If we imagine an active logic agent working literally as we describe in the semantics, it should be clear that any application of an apperception function to turn an inconsistent $KB_t$ into a consistent one will itself reduce the inferential power of (number of things that can be inferred from) $KB_t$, with the obvious extreme case being the application of an apperception function that uses a different unique superscript for every occurrence of every symbol in $KB_t$. In such a case (ignoring for the moment the fact that $KB_t$ can also change through observation, and not just through inference), the only things that could be inferred would be tautologies and simple elaborations, such as concluding $\neg S_1^0 \to S_2^0$ from $S_1^0$. Thus, it might appear that any actual active logic agent will only be able to infer a set of active consequences from its $KB$ that is much more limited than what is permitted by the semantics, and many will very quickly run out of interesting things to conclude, as a result of the apperception functions they apply. This suggests a significant practical problem.

Nevertheless, we believe that this semantics could in fact be used as a guide to building active logic agents, through a concept called the *ideal active logic agent*. This is an agent that can infer in one time-step exactly the 1-step active consequences of any $\Sigma$. Consider how such an ideal active logic agent could be built.

Obtaining the 1-step active logic consequences in $Sn_{\mathcal{L}_a}$ is fairly easy using Definition 14. It is just a matter of getting the appropriate *Now*, *Contra*, and *Bel* statements. The difficult part is getting the 1-step active logic consequences in $Sn_{\mathcal{L}_w}$.

Since $\Sigma$ is finite, as remarked in Section 4 only finitely many apperception functions are needed, say $ap_1, ap_2, \ldots, ap_n$. We may imagine one subagent $A_i$ per $ap_i$, $1 \leqslant i \leqslant n$. Observe that not all of these $n$ apperception functions are necessarily in $AP^{\Sigma}$ because some of them may not yield a consistent set in $\mathcal{L}_w$. Hence each subagent first needs to check if $ap_i \in AP^{\Sigma}$. In our case, since the language is basically propositional, this can be done by truth tables in a finite amount of time.

Next, continuing only with those sub-agents that passed the first test, enumerate all sentences in $Sn_{\mathcal{L}_w}$: $\varphi_1, \varphi_2, \ldots$ and have each $A_i$ check if $ap_i^z(\Sigma) \models \varphi_j$ for $j = 1, 2, \ldots$. Again, each such check can be done by truth tables in a finite amount of time. Each $A_i$ should include the set of $\varphi_j$s that pass this test. This way, the $n$ agents will generate (up to equivalence of apperception functions) exactly those sentences of $\mathcal{L}_w$ that are 1-step active consequences. We could even dispense with the equivalence above by using infinitely many subagents.

Although it thus appears *possible* to build an ideal active logic agent, it is pretty clearly not a *practical* way to obtain active logic consequences. (In fact, if the logic were truly first-order, the consistency and implication checks could not be done.) So we introduce another concept: the *practical active logic agent*. The practical active logic agent is an implementable inferencing (and observing) agent that will infer *only* 1-step active consequences of its $KB$, but not *all* of them. The practical research question, then, is how such practical active logic agents might be built—with what rules and what apperception functions—so that they are limited to inferring only 1-step active consequences of their beliefs, but do not thereby have severely limited inferential powers. We will discuss this in the next section.

## 10. Practical active logic agents

In an earlier attempt at defining a semantics for active logic [2], we discussed a version of the apperception function quite similar to what we have laid out here, but with the key difference that after an inference had taken place (i.e., after each time step) all the superscripts were automatically returned to 0. Let's call this the "reset version" of the apperception function. The advantage of the reset version was that it was possible to imagine literally implementing it, such that the agent could choose a small, clearly consistent set of formulas on which to focus to do inference, and then apply an apperception function such that the symbols in this set of formulas retained the superscript 0, and all other symbols were given unique non-zero superscripts. This guaranteed the consistency of the *KB* in a practically implementable way, and allowed inferences similar to those described here.

However, while such an agent is practically implementable, and inferentially powerful, Johan Hovold showed that it would in fact be *too* powerful, capable of inferring things that are not active consequences of its *KB* [26]. In particular, he showed that a logic using such an apperception function would be explosive.

Consider the following (inconsistent) set of sentences at time $t$: $\{S_1^0, S_1^0 \rightarrow S_2^0, \neg S_2^0\}$. Applying an apperception function such that this becomes $\{S_1^1, S_1^0 \rightarrow S_2^0, \neg S_2^0\}$ yields a consistent set, and the following set is a possible 1-step active consequence $\{S_1^1, S_1^0 \rightarrow S_2^0, \neg S_2^0, S_2^0 \rightarrow S_3^0\}$.[6] Since in this version of the apperception function all superscripts are now returned to zero, we get, at time $t+1$: $\{S_1^0, S_1^0 \rightarrow S_2^0, \neg S_2^0, S_2^0 \rightarrow S_3^0\}$. Now imagine an apperception function with the following effect: $\{S_1^0, S_1^0 \rightarrow S_2^0, \neg S_2^1, S_2^0 \rightarrow S_3^0\}$. A 1-step active consequence of this set at time $t+2$ is $\{S_3^0\}$. Since this would be true for any arbitrary sentence $S_3$, a logic using this version of the apperception function is indeed explosive, in that any arbitrary sentence is derivable (in two time steps) from an inconsistent set.

It was in response to this discovery by Hovold that we changed the apperception function used in defining the *semantics* of active logic to the version detailed in Definition 11. But a practical active logic agent could still use the reset version and avoid the charge of explosivity, just as long as it lacked the necessary rule for inferring $S_1 \rightarrow S_2$ from $\neg S_1$. Thus, we claim that an agent equipped with the reset version of the apperception function, along with a carefully chosen set of inference rules, could be a practical active logic agent: it would be possible to implement, and would never conclude anything that was not an active consequence of its *KB* (according to Definitions 14 and 15 of active consequence given above).

The simplest example of such an agent is one equipped with only the timing rule (Definition 17), the direct contradiction rule (Definition 18), the positive introspection rule (Definition 19), the inheritance rule (Definition 21), and active modus ponens (Definition 22). Clearly, such an agent will have somewhat limited inferential abilities, yet would be perfectly adequate to many practical situations. The obvious question about such practical active logic agents is: exactly which rules, in which combinations, allow for maximum inferential power, while still limiting the agent to inferring only active consequences (according to Definitions 14 and 15) of its *KB*? This question is something we leave to future work.

## 11. Conclusion and future work

In this paper we have outlined a semantics for a time-sensitive, contradiction-tolerant logical reasoning formalism designed for on-board use by real-world agents. Central to the semantics is the notion of an apperception function, inspired by the idea that, until an agent *notices* that a set of beliefs is inconsistent, that set *seems* consistent—and that when an inconsistency *is* noticed, that fact can be explicitly registered by the agent, and further reasoning with the inconsistent beliefs can be curtailed.

To keep this initial presentation relatively simple, we made a number of assumptions that in future work we hope to discard. For example, we assumed that the world is stationary, and thus all facts about the world are timelessly true. It should be noted that there is no problem in principle with applying active logic to the case of reasoning about a changing world—after all, the facts that beliefs are held at times, that the *KB* changes over time, and that inference is itself a temporal phenomenon, are all already explicitly modeled by the formalism. To handle a changing world, we would also have to model the additional facts that beliefs can be held not just *at* times, but *about* facts-at-times, and even about the *durations* of facts—e.g. that it rained yesterday, or that it rained yesterday for 1 hour between

[6] The added formula is a consequence of $\neg S_2^0$.

noon and one. Such modification is straightforward, especially given that we already have defined the changing set of world-facts $W_t$, by reference to which the truths of temporally relative beliefs would be determined.

But whereas these changes are easily handled, there are some tricky aspects to modeling proper *reasoning* with temporally relative beliefs in a changing world. For instance, although in active logic we do *not* inherit $Now(t)$ at time $t + 1$, we probably *do* want to inherit a belief like "It is raining at $t$" to time $t + 1$, for even at time $t + 1$ it remains true that it was raining at $t$. Further, it will generally be reasonable to conclude from "It is raining at $t$" that "It is raining at $t + 1$", for if it is raining now, it will probably still be raining at the next moment. But unlike the case of $Now(t)$, from which it will *always* be correct to conclude $Now(t + 1)$, it will *not* always be correct to conclude "It is raining at $t + 1$" from "It is raining at $t$". Moreover, for different facts, the likely duration of their continued truth will also differ. From the fact that it is raining now, I might reasonably conclude that it is still raining five minutes later; but it would not be reasonable to conclude, just on this basis, that it is still raining twenty-four hours later. In contrast, from the fact that a mountain is in such-a-such a place it will be reasonable to continue to infer its truth for a very long time indeed. To handle such issues we can avail ourselves of the extensive literature on default reasoning and non-monotonic temporal logics, e.g., [4,9,10,14,19,25,35–37,39,47]. Because of these and similar complications, we thought that the issue of reasoning about a changing world deserved special, separate treatment.

Future work will also consider the extension of $\mathcal{L}_w$ to first-order logic; multiple agents, reasoning both about the world and about one another's beliefs; and extending the semantics to include other predicates.

## Acknowledgements

This work was supported in part by a grant from AFOSR. A preliminary version was presented at the 7th Annual Symposium on the Logical Formalization of Commonsense Reasoning, in May, 2005. We thank the members of the audience for suggestions. We particularly want to thank Johan Hovold and the referees for their careful reading and many helpful remarks on an earlier version.

## References

[1] J. Allen, Towards a general theory of action and time, Artificial Intelligence 23 (1984) 123–154.

[2] M.L. Anderson, W. Gomaa, J. Grant, D. Perlis, On the reasoning of real-word agents: Toward a semantics for active logic, in: Proceedings of the 7th Annual Symposium on the Logical Formalization of Commonsense Reasoning, Dresden University Technical Report (ISSN 1430-211X), 2005.

[3] M. Asker, J. Malec, Reasoning with limited resources: Active logics expressed as labeled deductive systems, Bulletin of the Polish Academy of Sciences (2005) 123–154.

[4] F. Bacchus, A. Grove, J. Halpern, D. Koller, Statistical foundations for default reasoning, in: IJCAI, 1993.

[5] D. Batens, C. Mortensen, G. Priest, J.-P. Van Bendegen, Frontiers of Paraconsistent Logic, Taylor & Francis Group, 2000.

[6] D. Batens, J. Meheus, Recent results by the inconsistency-adaptive labourers, Technical report, Universiteit Gent, 2005.

[7] N.D. Belnap, A useful four-valued logic, in: J.M. Dunn, G. Epstein (Eds.), Modern Uses of Multiple-Valued Logics, D. Reidel, 1977, pp. 8–37.

[8] S. Benferhat, D. Dubois, H. Prade, Some syntactic approaches to the handling of inconsistent knowledge bases: A comparative study, part 1: The flat case, Studia Logica 58 (1997) 17–45.

[9] P. Besnard, T. Schaub, An approach to context-based default reasoning, Fundamenta Informaticae (1995).

[10] R. Brachman, I lied about the trees or, defaults and definitions in knowledge representation, AI Magazine 6 (3) (1985) 80–93.

[11] F. Brown (Ed.), The Frame Problem in Artificial Intelligence, Morgan Kaufmann, 1987.

[12] A. Carnielli, M.E. Coniglio, I.M.L. D'Ottaviano, Paraconsistency: The Logical Way to the Inconsistent, Marcel Dekker, Inc., 2002.

[13] N.C.A. da Costa, J.-Y. Beziau, O. Bueno, Paraconsistent logic in a historical perspective, Logique & Analyse 150 (2) (1995) 111–125.

[14] J.P. Delgrande, An approach to default reasoning based on first-order conditional logic: Revised report, Artificial Intelligence 36 (1) (1988) 63–90.

[15] J. Doyle, A truth maintenance system, Artificial Intelligence 12 (1979) 231–272.

[16] J. Doyle, A model for deliberation action, and introspection, PhD thesis, Massachusetts Institute of Technology, 1980.

[17] J. Elgot-Drapkin, Step-logic: Reasoning situated in time, PhD thesis, Department of Computer Science, University of Maryland, College Park, Maryland, 1988.

[18] J. Elgot-Drapkin, D. Perlis, Reasoning situated in time I: Basic concepts, Journal of Experimental and Theoretical Artificial Intelligence 2 (1) (1990) 75–98.

[19] D. Etherington, A semantics for default logic, in: Proceedings of the 10th Int'l Joint Conference on Artificial Intelligence, Milan, Italy, 1987, pp. 495–498.

[20] D. Gabbay, Labelled Deductive Systems, Oxford University Press, 1996.

[21] D.M. Gabbay, A. Hunter, Restricted access logics for inconsistent information, in: M. Clarke, R. Kruse, S. Moral (Eds.), Symbolic and Quantitative Approaches to Reasoning and Uncertainty, Springer, 1993, pp. 137–144.

[22] E. Giunchiglia, J. Lee, V. Lifschitz, N. McCain, H. Turner, Nonmonotonic causal theories, Artificial Intelligence 153 (2004) 49–104.

[23] P. Gärdenfors, Knowledge in Flux: Modeling the Dynamics of Epistemic States, MIT Press, Cambridge, MA, 1988.

[24] J. Grant, V.S. Subrahmanian, Applications of paraconsistency in data and knowledge bases, Synthese 125 (2000) 121–132.

[25] S. Hanks, D. McDermott, Nonmonotonic logic and temporal projection, Artificial Intelligence 33 (1987) 379–412.

[26] J. Hovold, On a semantics for active logic, MA Thesis, Department of Computer Science, Lund University, 2005.

[27] H.O. Ismail, S.C. Shapiro, Two problems with reasoning and acting in time, in: Principles of Knowledge Representation and Reasoning: Proceedings of the Seventh International Conference, 2000.

[28] K. Konolige, A Deduction Model of Belief, Pitman, London, 1986.

[29] H. Levesque, A logic of implicit and explicit belief, in: Proceedings of the National Conference on Artificial Intelligence, Austin, TX, American Association for Artificial Intelligence, 1984, pp. 198–202.

[30] J. McCarthy, Programs with common sense, in: Proceedings of the Symposium on the Mechanization of Thought Processes, Teddington, England, National Physical Laboratory, 1958.

[31] J. McCarthy, P. Hayes, Some philosophical problems from the standpoint of artificial intelligence, in: B. Meltzer, D. Michie (Eds.), Machine Intelligence, Edinburgh University Press, 1969, pp. 463–502.

[32] M. Miller, A view of one's past and other aspects of reasoned change in belief, PhD thesis, Department of Computer Science, University of Maryland, College Park, Maryland, 1993.

[33] M. Miller, D. Perlis, Presentations and this and that: logic in action, in: Proceedings of the 15th Annual Conference of the Cognitive Science Society, Boulder, Colorado, 1993.

[34] M. Nirkhe, S. Kraus, M. Miller, D. Perlis, How to (plan to) meet a deadline between *now* and *then*, Journal of Logic and Computation 7 (1) (1997) 109–156.

[35] M. Nirkhe, D. Perlis, S. Kraus, Reasoning about change in a changing world, in: Proceedings of FLAIRS-93, 1993.

[36] D. Perlis, Intentionality and defaults, Internat. J. Expert Systems 3 (1990) 345–354, Special issue on the Frame Problem. Reprinted as a chapter in: K. Ford, P. Hayes (Eds.), Advances in Human and Machine Cognition, vol. 1: the Frame Problem in Artificial Intelligence, JAI Press, 1991.

[37] D. Perlis, J. Elgot-Drapkin, M. Miller, Stop the world!—I want to think!, Internat. J. Intelligent Systems 6 (1991) 443–456. Special issue on temporal reasoning.

[38] A. Pnueli, The temporal logic of programs, in: Proceedings of the 18th IEEE Symposium on Foundations of Computer Science, 1977, pp. 46–67.

[39] D. Poole, A logical framework for default reasoning, Artificial Intelligence 36 (1988) 27–47.

[40] G. Priest, Paraconsistent logic, in: D. Gabbay, F. Guenther (Eds.), Handbook of Philosophical Logic, second ed., Kluwer Academic Publishers, 2002, pp. 287–393.

[41] G. Priest, R. Routley, J. Norman, Paraconsistent Logic: Essays on the Inconsistent, Philosophia Verlag, München, 1989.

[42] G. Priest, K. Tanaka, Paraconsistent logic, in: E.N. Zalta (Ed.), The Stanford Encyclopedia of Philosophy, Summer 2004.

[43] A.N. Prior, Past, Present and Future, Clarendon Press, Oxford, 1967.

[44] A.N. Prior, Papers on Time and Tense, Clarendon Press, Oxford, 1968.

[45] A. Prior, Time and Modality, Oxford University Press, 1957.

[46] K. Purang, Systems that detect and repair their own mistakes, PhD thesis, Department of Computer Science, University of Maryland, College Park, Maryland, 2001.

[47] R. Reiter, A logic for default reasoning, Artificial Intelligence 13 (1, 2) (1980) 81–132.

[48] N. Rescher, R. Manor, On inference from inconsistent premises, Theory and Decision 1 (1970) 179–219.

[49] S.C. Shapiro, Embodied cassie, in: Cognitive Robotics: Papers from the 1998 AAAI Fall Symposium, AAAI Press, Menlo Park, CA, 1998, pp. 136–143.

[50] S.C. Shapiro, Sneps: A logic for natural language understanding and commonsense reasoning, in: L.M. Iwanska, S.C. Shapiro (Eds.), Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language, AAAI Press/The MIT Press, 2000, pp. 175–195.