JAN VAN EIJCK AND FER-JAN DE VRIES

REASONING ABOUT UPDATE LOGIC

ABSTRACT. Logical frameworks for analysing the dynamics of information processing abound [4, 5, 8, 10, 12, 14, 20, 22]. Some of these frameworks focus on the dynamics of the interpretation process, some on the dynamics of the process of drawing inferences, and some do both of these. Formalisms galore, so it is felt that some conceptual streamlining would pay off.

This paper is part of a larger scale enterprise to pursue the obvious parallel between information processing and imperative programming. We demonstrate that logical tools from theoretical computer science are relevant for the logic of information flow. More specifically, we show that the perspective of Hoare logic [13, 18] can fruitfully be applied to the conceptual simplification of information flow logics.

Part one of this program consisted of the analysis of 'dynamic interpretation' in this way, using the example of dynamic predicate logic [10]; the results were published in [7]. The present paper constitutes the second part of the program, the analysis of 'dynamic inference'. Here we focus on Veltman's update logic [22].

Update logic is an example of a logical framework which takes the dynamics of drawing inferences into account by modelling information growth as discarding of possibilities. This paper shows how information logics like update logic can fruitfully be studied by linking their dynamic principles to static 'correctness descriptions'.

Our theme is exemplified by providing a sound and complete Hoare/Pratt style deduction system for update logic. The Hoare/Pratt correctness statements use modal propositional dynamic logic as assertion language and connect update logic to the modal propositional logic S5.

The connection with S5 provides a clear link between the dynamic and the static semantics of update logic. The fact that update logic is decidable was noted already in [2]; the connection with S5 provides an alternative proof. The S5 connection can also be used for rephrasing the validity notions of update logic and for performing consistency checks.

In conclusion, it is argued that interpreting the dynamic statements of information logics as dynamic modal operators has much wider applicability. In fact, the method can be used to axiomatize quite a wide range of information logics.

Keywords and Phrases: dynamic interpretation, Hoare logic, dynamic logic, knowledge representation languages.

1. INTRODUCTION

In the logical analysis of information and information processing two approaches can be distinguished. One approach takes the notion of truth as central. In this static approach growth of information by means of an utterance is viewed as adding the truth conditional content of the utterance to a given information state. The notion of inference is also defined in terms of truth conditions, as meaning inclusion in all models. The other approach takes information and information change as its central notions. A state of information is given by the set of possibilities which it leaves open. The nature of the possibilities varies of course with the theory of information at issue. In Lewis [17] the possibilities are possible answers to questions like 'Who is the speaker?', 'What is the current topic of conversation?', etcetera. In Heim [12] and Kamp [14] the possibilities are the discoure markers that are salient for anaphoric reference at the current stage of the discourse. In Groenendijk and Stokhof [10] they are possible assignments of values to variables. In Gärdenfors [8] the possibilities are (in the simplest version of the theory) the possible models of a given set of sentences. In Veltman [22] the possibilities are possible valuations for a set of proposition letters.

The first approach to information processing may be called static: evaluation at a given state is the basic notion. The second approach is dynamic in that information change is at the core of the approach. In the dynamic perspective, the meaning of a sentence is equated with its information change potential, with the effect that it has on a given state of information. Meanings are functions from information states to information states.

If one applies this to the semantics of natural language, (see for example Karttunen [15], Stalnaker [21], Kamp [14], Heim [12] and Barwise [1]) then the meaning of a text is the change it brings about in the information state of anyone who accepts the information conveyed by it. One perspective on dynamic semantics for natural language is to view this approach as a proposal to represent the meanings of natural language sentences not by means of formulae from static logic but by means of expressions from a dynamic action language. The action languages that have been proposed display an intriguing mix of features of programming languages and features of logical languages.

It has been demonstrated by example in [1] that under a dynamic regime compositional translations become feasible for fragments of natural language which include features that have resisted compositional treatment in the static representation approach (the most notable features being the handling of 'donkey' pronouns and pronominal binding across sentence boundaries).

The move from static logic to dynamic logic raises some interesting questions. In the first place, due to this transition we seem to have lost the deduction systems that static logical languages carry with them. This was noted by Barwise in [1], one of the first papers to propose an action language as representation medium for natural language meaning. Here the quest for a complete set of axioms for the dynamic inference notion that is engendered by the action language is put forward as an open problem. Next it can be asked what is the precise relation between the static semantics and the dynamic semantics of natural language. Are there systematic ways to derive the static meaning of a sentence from its dynamic representation?

These questions are intimately connected. Our contention is that they should be tackled together and moreover that theoretical computer science can guide the way. In computer science the view that the meaning of a program is a function from information states to information states is common ground. In the case of imperative programming languages this perspective has led to Hoare logic [13] as a successful means to construct deductive systems for reasoning about imperative programs.

To apply this to information processing in a very general sense, consider a reader of a text π as an agent who uses π to update her or his knowledge φ. Unless there is a consistency clash the agent will end up with more specific knowledge φ'. Taking our cue from Hoare logic we ask the following question. What is the weakest formula φ such that any knowledge implying φ remains consistent during the process of absorbing the information from text π? This weakest precondition φ for successful processing represents the static meaning of text π. It is the careful analysis of these weakest preconditions that leads to Hoare deduction systems which are sound and complete for a given dynamic semantics.

In this paper we will study Veltman's [22] update logic from this perspective. The simplest version of update logic is exemplary for dynamic approaches to information. Information states are just sets of valuations to a set P of proposition letters, i.e. sets of subsets of P. Valuations to proposition letters might be called possible worlds, so information states are sets of possible worlds. The set W of all possible worlds is $\mathcal{P}P$, the information state W is the state of complete ignorance (no possibility is excluded), for any $w \in W$, the state $\{w\}$ is a state of complete information (all possibilities except w have been excluded), $\emptyset$ is the absurd information state (nothing is compatible with the information). A further simplifying assumption of Veltman's update logic is the disregard of information revision (a central aspect of, e.g., Gärdenfors [8]).

Update logic can be used for the analysis of the epistemic sense of *maybe* or *might*. If I say *Maybe it rains*, or *Mary might be at home* then I wish to convey that the possibility of rain cannot be excluded on the basis of what I know, or that my evidence about Mary's whereabouts does not exclude her being at home (she took a day off from work, or I see light at her window).

It is hoped that the simplicity of update logic will help us to clarify our more general points about the relation between static and dynamic concepts in theories of information processing. We are in fact convinced that the link between the statics and dynamics of update logic by means of a Hoare/Pratt style analysis can be generalised to more complex systems of information flow logic.

To end this introduction, here is an overview of the contents and structure of the paper. In Section 2 Veltman's update logic is presented. Section 3 consists of a brief review of the tools from propositional modal logic that we will need. Section 4 contains the definition of validity for the assertion logic that is the foundation of our adaptation of Hoare/Pratt assertion reasoning to update semantics. The key notion of this section is the notion of a weakest precondition for a program of update logic. Section 5 links weakest preconditions to the next state conditions for update logic that were proposed by Van Benthem [2].

Section 6 contains the Hoare/Pratt calculus engendered by the notion of weakest preconditions of update programs. In Section 7 we prove the soundness and in Section 8 the completeness of the Hoare/Pratt calculus. In Section 9 we illustrate how the weakest preconditions analysis and the link to modal propositional logic can be used for reasoning about update logic. Section 10 demonstrates how weakest preconditions and the Hoare/Pratt calculus that is based on them can be used for reasoning about consistency of update programs. Finally, in Section 11 we wind up our story by connecting our program with related work and listing some directions for future research.

2. UPDATE LOGIC: SYNTAX AND SEMANTICS

The characteristic feature of Veltman's update logic (see Veltman [22]) is the epistemic modal operator **might**. Due to the presence of this operator the meanings of 'update programs' have to be phrased in terms of input information *sets*, and have to be phrased dynamically. The formulae of update logic have to be distinguished from the formulae of the static language used to make assertions about update logic. Because of the dynamic flavour of the former we will refer to these as 'update programs'. We will see that sequential composition of update programs does not in general reduce to Boolean conjunction.

An update program π maps an information state I to a new information state $[\![\pi]\!](I)$. To see that **might** is the key feature, note that the semantics for the fragment of update logic without **might** can be given by means of a dynamic yes/no function for individual propositional valuations, which reduces the semantics immediately to ordinary static propositional logic.

Following Veltman [22] (and in fact, slightly simplifying his syntax), we can define the language of epistemic update logic over a set of proposition letters P as the smallest set L_P such that the following hold:

DEFINITION 1 (Syntax of Update Logic L_P).

1. $\bot \in L_P$.

2. If $p \in P$ then $p \in L_P$.
3. If π and $\pi' \in L_P$, then $(\pi; \pi') \in L_P, (\pi \cup \pi') \in L_P$.
4. If $\pi \in L_P$, then $\neg\pi \in L_P$, **might** $\pi \in L_P$.
5. Nothing else is in L_P.

The semantics of L_P is given in tefms of input-output behaviour. We take the set W of worlds over P to be the set $\mathcal{P}P$. Any subset of W is an information state. Progams are interpreted as functions from information states to information states, i.e., as functions in $\mathcal{P}W \rightarrow \mathcal{P}W$. The clauses are as follows:

DEFINITION 2 (Semantics of Update Logic.)

1. $[\![\bot]\!](I) = \emptyset$.
2. $[\![p]\!](I) = I \cap \{w \mid p \in w\}$.
3. $[\![\pi; \pi']\!](I) = [\![\pi']\!]([\![\pi]\!](I))$.
4. $[\![\pi \cup \pi']\!](I) = [\![\pi]\!](I) \cup [\![\pi']\!](I)$.
5. $[\![\neg\pi]\!](I) = I - [\![\pi]\!](I)$.
6. $[\![\mathbf{might}\ \pi]\!](I) = \begin{cases} I & \text{if } [\![\pi]\!](I) \neq \emptyset, \\ \emptyset & \text{otherwise.} \end{cases}$

We will follow the usual conventions and drop outermost parentheses as much as possible. Also, since sequential composition is associative we will write both $\pi_1; (\pi_2; \pi_3)$ and $(\pi_1; \pi_2); \pi_3$ as $\pi_1; \pi_2; \pi_3$.

Intuitively, a program of the form $\mathtt{might}\ \pi$ does not provide information about the world but about available information. A program $\mathtt{might}\ \pi$ is acceptable, given an information state I, if there is at least one world $w \in I$ for which π is accepted in the sense that $w \in [\![\pi]\!](I)$. If such a w can be found, the output information state of $\mathtt{might}\ \pi$ is equal to its input information state; this agrees with the intuition that $\mathtt{might}\ \pi$ does not say anything at all about what the world is like. In the other case, i.e., the case were $[\![\pi]\!](I) = \emptyset$, the output information state of $\mathtt{might}\ \pi$ equals $\emptyset$.

As was mentioned already, the $\mathtt{might}$ operator is the key feature of update logic. Yet another way to see this is to note that the semantic clause for $\mathtt{might}\ \pi$ introduces an element of *non distributivity* (this terminology is taken from [11]) into the semantics, in the sense that unions of input states do not distribute over output states: (1) does not in general hold.

(1) $$[\![\pi]\!](I) = \bigcup_{i \in I} [\![\pi]\!](\{i\}).$$

More specifically, it does not in general hold that $[\![\pi]\!](I) \subseteq \bigcup_{i \in I} [\![\pi]\!](\{i\})$. Counterexample: take π equal to $\mathbf{might}\ p$ and let $I = \{w, w'\}$ with $p \in w$ and $p \notin w'$. Then $[\![\mathbf{might}\ p]\!](\{w\}) \cup [\![\mathbf{might}\ p]\!](\{w'\}) = \{w\}$, but $[\![\mathbf{might}\ p]\!](I) = I = \{w, w'\}$.

On the other hand, a simple induction on the complexity of π shows that Lemma 1 holds for all $\pi \in L_P$ and all information states I. In the terminology of Groenendijk & Stokhof [11]: epistemic update logic is *eliminative*.

LEMMA 1 (Elimination Lemma.) *For all I: $[\![\pi]\!](I) \subseteq I$.*

3. ASSERTION LOGIC

The presence of the modal **might** operator in a dynamic setting which is otherwise fully propositional strongly suggests the use a modal propositional logic as language to make static assertions about update programs in, i.e., as assertion language. But we also want to be able to talk about execution results, so we add the update programs themselves as a second kind of modality. The syntax of our assertion language *mal*$_P$ is as follows:

DEFINITION 3 (Syntax of *mal*$_P$.)

1. $\bot \in mal_P$.
2. If $p \in P$ then $p \in mal_P$.
3. If $\varphi, \psi \in mal_P$, then $(\varphi \wedge \psi), \neg\varphi, \Diamond\varphi \in mal_P$.
4. If $\varphi \in mal_P$ and $\pi \in L_P$ then $\langle\pi\rangle\varphi \in mal_P$.
5. Nothing else is in *mal*$_P$.

As is customary, we abbreviate $\neg\bot$ as $\top$, $\neg(\neg\varphi \wedge \neg\psi)$ as $(\varphi \vee \psi)$, $\neg(\varphi \wedge \neg\psi)$ as $(\varphi \rightarrow \psi)$, $\neg\Diamond\neg\varphi$ as $\Box\varphi$, $\neg\langle\pi\rangle\neg\varphi$ as $[\pi]\varphi$, Also, we omit outermost parentheses for readability. We will refer to propositional modal logic (which is defined by omitting clause 4 from the definition of mal_P) as ml_P.

We consider information states $I \in \mathcal{P}W$ as universal Kripke models; thus, I is considered as the Kripke model with accessibility relation $I \times I$. Recall from the literature (see e.g. [9]) that the modal logic determined by the class of finite universal frames is S5. Moreover, for any finite universal model (a universal frame with valuations assigned to all of its worlds) there is a finite subset I of W validating the same formulae. I can be got by throwing away the extra copies of the worlds with identical valuations: because of the universal accessibility this makes no difference to validity.

It is convenient to define the interpretation of an mal_P formula with respect to an information state.

DEFINITION 4 (Interpretation of φ with respect to I.)

1. $||\bot||_I = \emptyset$.
2. $||p||_I = \{w \in I \mid p \in w\}$.
3. $||\varphi \wedge \psi||_I = ||\varphi||_I \cap ||\psi||_I$.
4. $||\neg\varphi||_I = I - ||\varphi||_I$.
5. $||\Diamond\varphi||_I = \begin{cases} I & \text{if } ||\varphi||_I \neq \emptyset, \\ \emptyset & \text{otherwise.} \end{cases}$
6. $||\langle\pi\rangle\varphi||_I = ||\varphi||_{[\![\pi]\!](I)}$.

We can now can define the notion $I, w \Vdash \varphi$ (world w forces formula φ in information state I) as $w \in ||\varphi||_I$.

In Hoare style reasoning about update logic the notions of relativisation and localisation of modal formulae play an important role. Localisations of modal formulae are defined in Kracht [16]. If φ, ψ are in mal_P, then $\varphi{\downarrow}\psi$, the localisation of φ to ψ, is given by the following definition.

DEFINITION 5 (Localised modal formulae $\varphi{\downarrow}\psi$.)

$$\begin{array}{ll}
\bot{\downarrow}\psi & = \bot \\
p{\downarrow}\psi & = p \wedge \psi \\
(\varphi_1 \wedge \varphi_2){\downarrow}\psi & = (\varphi_1{\downarrow}\psi) \wedge (\varphi_2{\downarrow}\psi) \\
(\neg\varphi){\downarrow}\psi & = \psi \wedge \neg(\varphi{\downarrow}\psi) \\
(\Diamond\varphi){\downarrow}\psi & = \psi \wedge \Diamond(\psi \wedge (\varphi{\downarrow}\psi)). \\
(\langle\pi\rangle\varphi){\downarrow}\psi & = \psi \wedge \langle\pi\rangle(\psi \wedge (\varphi{\downarrow}\psi)).
\end{array}$$

Localisation is closely related to the usual notion of a relativised modal formula.

DEFINITION 6 (Relativised modal formulae φ^ψ).

$$\begin{array}{ll}
\bot^\psi & = \bot \\
p^\psi & = \psi \rightarrow p \\
(\varphi_1 \wedge \varphi_2)^\psi & = \varphi_1^\psi \wedge \varphi_2^\psi \\
(\neg\varphi)^\psi & = \neg(\varphi^\psi) \\
(\Diamond\varphi)^\psi & = \Diamond(\psi \wedge \varphi^\psi). \\
(\langle\pi\rangle\varphi)^\psi & = \langle\pi\rangle(\psi \wedge \varphi^\psi).
\end{array}$$

The connection between the two notions is given by the following lemma.

LEMMA 2 (Van Benthem.) $\varphi{\downarrow}\psi$ *iff* $\varphi^\psi \wedge \psi$.

Proof. Induction on the structure of φ. For example, in the case of negation the reasoning is as follows.

$$\begin{array}{llll}
\neg\varphi{\downarrow}\psi & = & {\downarrow}\ \textit{def} & \psi \wedge \neg(\varphi{\downarrow}\psi) \\
 & = & \textit{ind hyp} & \psi \wedge \neg(\varphi^\psi \wedge \psi) \\
 & = & \textit{prop logic} & \psi \wedge \neg(\varphi^\psi) \\
 & = & {\downarrow}\ \textit{def} & \psi \wedge (\neg\varphi)^\psi.
\end{array}$$ ■

Given this connection, the following lemma will not come as a surprise (the first item is from the modal folklore, the second from Kracht [16]).

LEMMA 3 (Relativisation and Localisation).

1. $||\varphi^\psi||_I = (I - ||\psi||_I) \cup ||\varphi||_{||\psi||_I}$.

2. $||\varphi\downarrow\psi||_I = ||\varphi||_{||\psi||_I}$.

Proof. Both assertions are proved by induction on the complexity of φ. ■

4. CORRECTNESS STATEMENTS FOR UPDATE LOGIC

Once we have a notion of validity for assertions, the assertion language can be used to make correctness assertions about update logic. Here is the validity notion.

DEFINITION 7. Assume $\varphi \in mal_P$. Then $\models \varphi$ if for all $I \subseteq W$, $I = ||\varphi||_I$.

Now we immediately have the following.

LEMMA 4. $\models \varphi \leftrightarrow \langle\pi\rangle\psi$ *iff for all I:* $||\varphi||_I = ||\psi||_{[\![\pi]\!](I)}$.

Proof. Immediate from Definition 4 and Definition 7. ■

Note that it follows from Lemma 4 that $\models \varphi \leftrightarrow \langle\pi\rangle\top$ iff for all I: $||\varphi||_I = [\![\pi]\!](I)$.

The correctness statements suggest the following notion of weakest precondition for update logic.

DEFINITION 8. A formula $\varphi \in ml_P$ is a weakest precondition (WP) of the program $\pi \in L_P$ and the formula $\psi \in mal_P$ if for all I: $||\varphi||_I = ||\psi||_{[\![\pi]\!](I)}$.

It is not obvious at first sight that WPs of an L_P program π and an mal_P formula ψ always exist (as formulae of ml_P). We will demonstrate now that they do, by inductively defining a function $\mathbf{wp}(\pi, \psi)$, of which we will show that it expresses a WP of π and ψ.

DEFINITION 9 (wp.)

1. $\mathbf{wp}(\bot, \psi) = \bot$.
2. $\mathbf{wp}(p, \psi) = \psi\downarrow p$.
3. $\mathbf{wp}(\pi_1; \pi_2, \psi) = \mathbf{wp}(\pi_1, \mathbf{wp}(\pi_2, \psi))$.

4. $\mathbf{wp}(\pi_1 \cup \pi_2, \psi) = \psi{\downarrow}(\mathbf{wp}(\pi_1, \top) \vee \mathbf{wp}(\pi_2, \top))$.
5. $\mathbf{wp}(\neg\pi, \psi) = \psi{\downarrow}\neg\mathbf{wp}(\pi, \top)$.
6. $\mathbf{wp}(\mathbf{might}\ \pi, \psi) = \Diamond\mathbf{wp}(\pi, \top) \wedge \psi$.

An easy induction shows that $\mathbf{wp}(\pi, \psi) \in ml_P$, for $\pi \in L_P$ and $\psi \in mal_P$. Lemma 5 shows that the function $\mathbf{wp}(\pi, \psi)$ does indeed express a WP of a program π and a modal propositional formula ψ.

LEMMA 5 (wp adequacy.) $||\mathbf{wp}(\pi, \psi)||_I = ||\psi||_{[\![\pi]\!](I)}$.

Proof. We prove the claim with induction on the structure of π.

$$\begin{aligned}
||\mathbf{wp}(\bot, \psi)||_I &= \textit{wp def} & &||\bot||_I \\
&= ||\ ||_I \textit{ def} & &\emptyset \\
&= [\![\ ]\!] \textit{ def} & &||\psi||_{[\![\bot]\!](I)}.
\end{aligned}$$

$$\begin{aligned}
||\mathbf{wp}(p, \psi)||_I &= \textit{wp def} & &||\psi{\downarrow}p||_I \\
&= \textit{loc lemma} & &||\psi||_{||p||_I} \\
&= [\![\ ]\!] \textit{ def} & &||\psi||_{[\![p]\!](I)}.
\end{aligned}$$

$$\begin{aligned}
||\mathbf{wp}(\pi_1; \pi_2, \psi)||_I &= \textit{wp def} & &||\mathbf{wp}(\pi_1, \mathbf{wp}(\pi_2, \psi))||_I \\
&= \textit{ind hyp} & &||\mathbf{wp}(\pi_2, \psi)||_{[\![\pi_1]\!](I)} \\
&= \textit{ind hyp} & &||\psi||_{[\![\pi_2]\!]([\![\pi_1]\!](I))} \\
&= [\![\ ]\!] \textit{ def} & &||\psi||_{[\![\pi_1;\pi_2]\!](I)}.
\end{aligned}$$

$$\begin{aligned}
||\mathbf{wp}(\pi_1 \cup \pi_2, \psi)||_I &= \textit{wp def} & &||\psi{\downarrow}(\mathbf{wp}(\pi_1, \top) \vee \mathbf{wp}(\pi_2, \top))||_I \\
&= \textit{loc lemma} & &||\psi||_{||\mathbf{wp}(\pi_1, \top) \vee \mathbf{wp}(\pi_2, \top)||_I} \\
&= \textit{ind hyp}, ||\ ||_I \textit{ def} & &||\psi||_{[\![\pi_1]\!](I) \cup [\![\pi_2]\!](I)} \\
&= [\![\ ]\!] \textit{ def} & &||\psi||_{[\![\pi_1 \cup \pi_2]\!](I)}.
\end{aligned}$$

$$\begin{aligned}
||\mathbf{wp}(\neg\pi, \psi)||_I &= \textit{wp def} & &||\psi{\downarrow}\neg\mathbf{wp}(\pi, \top)||_I \\
&= \textit{loc lemma} & &||\psi||_{||\neg\mathbf{wp}(\pi, \top)||_I} \\
&= ||\ ||_I \textit{ def} & &||\psi||_{I - ||\mathbf{wp}(\pi, \top)||_I} \\
&= \textit{ind hyp}, ||\ ||_I \textit{ def} & &||\psi||_{I - [\![\pi]\!](I)} \\
&= [\![\ ]\!] \textit{ def} & &||\psi||_{[\![\neg\pi]\!](I)}.
\end{aligned}$$

$$\begin{aligned}
||\mathbf{wp}(\mathbf{might}\ \pi, \psi)||_I &= \textit{wp def} & &||\Diamond\mathbf{wp}(\pi, \top) \wedge \psi||_I \\
&= ||\ ||_I \textit{ def} & &||\Diamond\mathbf{wp}(\pi, \top)||_I \cap ||\psi||_I \\
&= \Diamond \textit{ def in S5} & &\begin{cases} ||\psi||_I & \text{if } ||\mathbf{wp}(\pi, \top)||_I \neq \emptyset, \\ \emptyset & \text{otherwise} \end{cases} \\
&= \textit{ind hyp}, ||\ ||_I \textit{ def} & &\begin{cases} ||\psi||_I & \text{if } [\![\pi]\!](I) \neq \emptyset, \\ \emptyset & \text{otherwise} \end{cases} \\
&= [\![\ ]\!] \textit{ def} & &||\psi||_{[\![\mathbf{might}\ \pi]\!](I)}.
\end{aligned}$$

This completes the proof of the lemma. ■

LEMMA 6.

1. $||\mathbf{wp}(\pi, \top)||_I = [\![\pi]\!](I)$.
2. $\models \varphi \leftrightarrow \langle\pi\rangle\top$ *iff for all I:* $||\mathbf{wp}(\pi, \top)||_I = ||\varphi||_I$.

Proof. The first item:

$$\begin{aligned} ||\mathbf{wp}(\pi, \top)||_I &= \text{wp adeq} \quad ||\top||_{[\![\pi]\!](I)} \\ &= || \, ||_I \text{ def} \quad [\![\pi]\!](I). \end{aligned}$$

The second item follows from Lemma 4 and the first item. ■

LEMMA 7. $||\mathbf{wp}(\pi, \psi)||_I = ||\psi{\downarrow}\mathbf{wp}(\pi, \top)||_I$.

Proof.

$$\begin{aligned} ||\mathbf{wp}(\pi, \psi)||_I &= \textit{wp adeq} \quad ||\psi||_{[\![\pi]\!](I)} \\ &= \textit{Lemma 6} \quad ||\psi||_{||\mathbf{wp}(\pi,\top)||_I} \\ &= \textit{loc lemma} \quad ||\psi{\downarrow}\mathbf{wp}(\pi, \top)||_I. \end{aligned}$$ ■

5. WEAKEST PRECONDITIONS VERSUS NEXT STATE CONDITIONS

In [2] and [3], Van Benthem has studied update logic by looking at update programs π as functions of the form $\lambda I \cdot \text{NEXT STATE}(I, \pi)$, were NEXT STATE is the function producing the information state which results from processing π in information state I, i.e., π is considered as $\lambda I \cdot [\![\pi]\!](I)$. The investigation in [2, 3] was carried out in semantic terms, without reference to a specific assertion language, but it can easily be transposed in a setting of assertions from modal propositional logic. Some illuminating conversations between Johan van Benthem and the authors, backed up by an exchange of letters of explanation and consecutive drafts of the present paper, have fully cleared up the connection between his perspective and ours. His generous help in clarifying the issues raised in this section is herewith gratefully acknowledged.

DEFINITION 10. A formula $\psi \in ml_P$ is a next state condition (NSC) of the formula $\varphi \in ml_P$ and the program $\pi \in L_P$ if for all I: $||\psi||_I = [\![\pi]\!](||\varphi||_I)$.

The following function **nsc** is a reformulation in modal logic of Van Benthem's characterisation of the next state function.

DEFINITION 11 (nsc.)

1. $\mathbf{nsc}\,(\varphi, \bot) = \bot$.
2. $\mathbf{nsc}\,(\varphi, p) = p \wedge \varphi$.
3. $\mathbf{nsc}\,(\varphi, \pi_1; \pi_2) = \mathbf{nsc}\,(\mathbf{nsc}\,(\varphi, \pi_1), \pi_2)$.
4. $\mathbf{nsc}\,(\varphi, \pi_1 \cup \pi_2) = \mathbf{nsc}\,(\varphi, \pi_1) \vee \mathbf{nsc}\,(\varphi, \pi_2)$.
5. $\mathbf{nsc}\,(\varphi, \neg\pi) = \neg\mathbf{nsc}\,(\varphi, \pi) \wedge \varphi$.
6. $\mathbf{nsc}\,(\varphi, \mathbf{might}\ \pi) = \Diamond\mathbf{nsc}\,(\varphi, \pi) \wedge \varphi$.

LEMMA 8 (nsc adequacy.) $||\mathbf{nsc}\,(\varphi, \pi)||_I = [\![\pi]\!](||\varphi||_I)$.

Proof. Induction on the structure of π.

$$\begin{aligned} ||\mathbf{nsc}\,(\varphi, \bot)||_I &= \textit{nsc def} \quad ||\bot||_I \\ &= ||\ ||_I\ \textit{def} \quad \emptyset \\ &= [\![\ ]\!]\ \textit{def} \quad [\![\bot]\!](||\varphi||_I). \end{aligned}$$

$$\begin{aligned} ||\mathbf{nsc}\,(\varphi, p)]_I &= \textit{nsc def} \quad ||p \wedge \psi||_I \\ &= ||\ ||_I\ \textit{def} \quad ||p||_I \cap ||\varphi||_I \\ &= [\![\ ]\!]\ \textit{def} \quad [\![p]\!](||\varphi||_I). \end{aligned}$$

$$\begin{aligned} ||\mathbf{nsc}\,(\varphi, \pi_1; \pi_2)||_I &= \textit{nsc def} \quad ||\mathbf{nsc}\,(\mathbf{nsc}\,(\varphi, \pi_1), \pi_2)||_I \\ &= \textit{ind hyp} \quad [\![\pi_2]\!](||\mathbf{nsc}\,(\varphi, \pi_1)||_I) \\ &= \textit{ind hyp} \quad [\![\pi_2]\!]([\![\pi_1]\!](||\varphi||_I)) \\ &= [\![\ ]\!]\ \textit{def} \quad [\![\pi_1; \pi_2]\!](||\varphi||_I). \end{aligned}$$

$$\begin{aligned} ||\mathbf{nsc}\,(\varphi, \pi_1 \cup \pi_2)||_I &= \textit{nsc def} \quad ||\mathbf{nsc}\,(\varphi, \pi_1) \vee \mathbf{nsc}\,(\varphi, \pi_2)||_I \\ &= \textit{ind hyp} \quad [\![\pi_1]\!](||\varphi||_I) \cup [\![\pi_2]\!](||\varphi||_I) \\ &= [\![\ ]\!]\ \textit{def} \quad [\![\pi_1 \cup \pi_2]\!](||\varphi||_I). \end{aligned}$$

$$\begin{aligned} ||\mathbf{nsc}\,(\varphi, \neg\pi)||_I &= \textit{nsc def} \quad ||\neg\mathbf{nsc}\,(\varphi, \pi) \wedge \varphi||_I \\ &= ||\ ||_I\ \textit{def} \quad (I - ||\mathbf{nsc}\,(\varphi, \pi)||_I) \cap ||\varphi||_I \\ &= \textit{ind hyp} \quad (I - [\![\pi]\!](||\varphi||_I)) \cap ||\varphi||_I \\ &= \textit{elim lemma} \quad (||\varphi||_I - [\![\pi]\!](||\varphi||_I)) \cap ||\varphi||_I \\ &= [\![\ ]\!]\ \textit{def} \quad [\![\neg\pi]\!](||\varphi||_I). \end{aligned}$$

$$\begin{aligned} ||\mathbf{nsc}\,(\varphi, \mathbf{might}\ \pi)||_I &= \textit{nsc def} \quad ||\Diamond\mathbf{nsc}\,(\varphi, \pi) \wedge \varphi||_I \\ &= ||\ ||_I\ \textit{def} \quad ||\Diamond\mathbf{nsc}\,(\varphi, \pi)||_I \cap ||\varphi||_I \\ &= \Diamond\ \textit{def in S5} \quad \begin{cases} ||\varphi||_I & \text{if } ||\mathbf{nsc}\,(\varphi, \pi)||_I \neq \emptyset, \\ \emptyset & \text{otherwise} \end{cases} \\ &= \textit{ind hyp} \quad \begin{cases} ||\varphi||_I & \text{if } [\![\pi]\!](||\varphi||_I) \neq \emptyset, \\ \emptyset & \text{otherwise} \end{cases} \\ &= [\![\ ]\!]\ \textit{def} \quad [\![\mathbf{might}\ \pi]\!](||\varphi||_I). \end{aligned}$$

This completes the proof of the lemma. ∎

LEMMA 9. $||\mathbf{nsc}\,(\top, \pi)||_I = [\![\pi]\!](I)$.

Proof.

$$\begin{array}{rcll} ||\mathbf{nsc}\,(\top, \pi)||_I & = & \textit{nsc adeq} & [\![\pi]\!](||\top||_I) \\ & = & || \; ||_I \textit{ def} & [\![\pi]\!](I). \end{array}$$ ∎

The following theorem gives the precise connections between WPs and NSCs.

THEOREM 10.

1. $||\mathbf{nsc}\,(\varphi, \pi)||_I = ||\mathbf{wp}(\pi, \top){\downarrow}\varphi||_I$.
2. $||\mathbf{wp}(\pi, \psi)||_I = ||\psi{\downarrow}\mathbf{nsc}\,(\top, \pi)||_I$.

Proof. The first item:

$$\begin{array}{rcll} \mathbf{nsc}\,(\varphi, \pi)||_I & = & \textit{nsc adeq} & [\![\pi]\!](||\varphi||_I) \\ & = & \textit{Lemma 6} & ||\mathbf{wp}(\pi, \top)||_{||\varphi||_I} \\ & = & \textit{loc lemma} & ||\mathbf{wp}(\pi, \top){\downarrow}\varphi||_I. \end{array}$$

The second item:

$$\begin{array}{rcll} ||\mathbf{wp}(\pi, \psi)||_I & = & \textit{wp adeq} & ||\psi||_{[\![\pi]\!](I)} \\ & = & \textit{Lemma 9} & ||\psi||_{||\mathbf{nsc}\,(\top,\pi)||_I} \\ & = & \textit{loc lemma} & ||\psi{\downarrow}\mathbf{nsc}\,(\top, \pi)||_I. \end{array}$$ ∎

6. A HOARE/PRATT CALCULUS FOR UPDATE LOGIC

We now present the axioms and rules of a deduction system for update logic based on the concept of WPs from Section 4. We start with the axiom schemata for propositional logic.

A 1 $\varphi \rightarrow (\psi \rightarrow \varphi)$.

A 2 $(\varphi \rightarrow (\psi \rightarrow \chi)) \rightarrow ((\varphi \rightarrow \psi) \rightarrow (\varphi \rightarrow \chi))$.

A 3 $(\neg\varphi \rightarrow \neg\psi) \rightarrow (\psi \rightarrow \varphi)$.

Next, we take the axiom schemata of S5 modal logic for the $\Box$ modality.

A 4 $\Box(\varphi \rightarrow \psi) \rightarrow (\Box\varphi \rightarrow \Box\psi)$.

A 5 $\Box\varphi \rightarrow \varphi$.

A 6 $\Box\varphi \rightarrow \Box\Box\varphi$.

A 7 $\Diamond\Box\varphi \rightarrow \varphi$.

These are the propositional S5 modalities. Here are the axiom schemata for the program modalities.

A 8 $\bot \leftrightarrow \langle\bot\rangle\varphi$.

A 9 $\varphi{\downarrow}p \leftrightarrow \langle p\rangle\varphi$.

A 10 $\langle\pi_1\rangle\langle\pi_2\rangle\varphi \leftrightarrow \langle\pi_1;\pi_2\rangle\varphi$.

A 11 $\varphi{\downarrow}(\langle\pi_1\rangle\top \vee \langle\pi_2\rangle\top) \leftrightarrow \langle\pi_1 \cup \pi_2\rangle\varphi$.

A 12 $\varphi{\downarrow}[\pi]\bot \leftrightarrow \langle\neg\pi\rangle\varphi$.

A 13 $(\Diamond\langle\pi\rangle\top \wedge \varphi) \leftrightarrow \langle\mathbf{might}\pi\rangle\varphi$.

The rules of inference are as follows.

R 1 (Necessitation for $\Box$.) *Conclude from $\vdash \varphi$ to $\vdash \Box\varphi$.*

R 2 (Necessitation for program modalities.) *For every program π of update logic: conclude from $\vdash \varphi$ to $\vdash [\pi]\varphi$.*

R 3 (Modus Ponens.) *Conclude from $\vdash \varphi \rightarrow \psi$ and $\vdash \varphi$ to $\vdash \psi$.*

The notion of theoremhood in the calculus is standard.

DEFINITION 12. Formula φ is a theorem of the calculus, notation $\vdash \varphi$, if φ fits one of the axiom schemata or φ follows from theorems in the calculus by an application of one of the inference rules.

Here is an example of a derived schema.

PROPOSITION 11. *For every update program π, the K schema is derivable:*

$$\vdash ([\pi](\varphi \rightarrow \psi) \wedge [\pi]\varphi) \rightarrow [\pi]\psi.$$

Proof. Induction on the complexity of π. ■

7. SOUNDNESS OF THE CALCULUS

To prove that the calculus is sound, we have to prove that if $\vdash \varphi$, then φ is valid in the sense defined in Section 4. As usual, soundness is proved by induction on the length of the derivation of φ. For this, we have to check that every axiom of the calculus is valid and that the rules of the calculus preserve validity.

THEOREM 12 (Soundness.) *For all $\varphi \in mal_P$: If $\vdash \varphi$ then $\models \varphi$.*

Proof. First, it is obvious that the axiom schemata of propositional logic are valid. Next, observe that it follows from the definition of $||\ ||_I$ that

$$||\Box\varphi||_I = \begin{cases} I & \text{if } ||\varphi||_I = I, \\ \emptyset & \text{otherwise.} \end{cases}$$

For the validity of Axiom 4 we have to show (2).

(2) *For all I,* $||\Box(\varphi \rightarrow \psi) \rightarrow (\Box\varphi \rightarrow \Box\psi)||_I = I.$

This is equivalent to (3).

(3) *For all I,* $||\Box(\varphi \rightarrow \psi)||_I \subseteq ||\Box\varphi \rightarrow \Box\psi||_I.$

Two cases. If $||\varphi \rightarrow \psi||_I \neq I$, then $||\Box(\varphi \rightarrow \psi)||_I = \emptyset$, and (3) trivially holds. Assume therefore that $||\varphi \rightarrow \psi||_I = I$. This is equivalent to

$||\varphi||_I \subseteq ||\psi||_I$ $(*)$. Now if $||\varphi||_I \neq I$, then $||\Box\varphi||_I = \emptyset$ and $||\Box\varphi||_I \subseteq ||\Box\psi||_I$, in other words $||\Box\varphi \rightarrow \Box\psi||_I = I$, and (3) holds. If, on the other hand, $||\varphi||_I = I$, then by $*$, $||\psi||_I = I$, and again (3) holds because $||\Box\varphi||_I \subseteq ||\Box\psi||_I$.

For Axiom 5, observe that $||\Box\varphi \rightarrow \varphi||_I = I$ iff $||\Box\varphi||_I \subseteq ||\varphi||_I$ iff it holds that if $||\varphi||_I = I$ then $I \subseteq ||\varphi||_I$, which is always true.

For Axiom 6, we have to show that $||\Box\varphi \rightarrow \Box\Box\varphi||_I = I$, or equivalently, $||\Box\varphi||_I \subseteq ||\Box\Box\varphi||_I$. If $||\varphi||_I \neq I$ then $||\Box\varphi||_I = \emptyset$, and the claim holds. If $||\varphi||_I = I$ then $||\Box\varphi||_I = I$, and so $||\Box\Box\varphi||_I = I$, and the the claim holds in this case too.

For Axiom 7, observe: $||\Diamond\Box\varphi \rightarrow \varphi||_I = I$ iff $||\Diamond\Box\varphi||_I \subseteq ||\varphi||_I$ iff if $||\Box\varphi||_I \neq \emptyset$ then $I \subseteq ||\varphi||_I$ iff if $||\varphi||_I = I$ then $I \subseteq ||\varphi||_I$ iff true.

Axiom 8: $||\bot \leftrightarrow \langle\bot\rangle\varphi||_I = I$ iff $||\bot||_I = ||\langle\bot\rangle\varphi||_I$ iff $\emptyset = ||\langle\bot\rangle\varphi||_I$ iff true.

The reasoning for Axiom 9: $||\varphi{\downarrow}p \leftrightarrow \langle p\rangle\varphi||_I = I$ iff $||\varphi{\downarrow}p||_I = ||\langle p\rangle\varphi||_I$ iff (Lemma 3) true.

Axiom 10:

$||\langle\pi_1\rangle\langle\pi_2\rangle\varphi \leftrightarrow \langle\pi_1;\pi_2\rangle\varphi||_I = I$
iff
$||\langle\pi_1\rangle\langle\pi_2\rangle\varphi||_I = ||\langle\pi_1;\pi_2\rangle\varphi||_I$
iff
$||\langle\pi_2\rangle\varphi||_{[\![\pi_i]\!]I} = ||\langle\pi_1;\pi_2\rangle\varphi||_I$
iff
$||\varphi||_{[\![\pi_2]\!]([\![\pi_i]\!]I)} = ||\langle\pi_1;\pi_2\rangle\varphi||_I$
iff true.

Axiom 11:

$||\varphi{\downarrow}(\langle\pi_1\rangle\top \vee \langle\pi_2\rangle\top) \leftrightarrow \langle\pi_1 \cup \pi_2\rangle\varphi||_I = I$
iff
$||\varphi{\downarrow}(\langle\pi_1\rangle\top \vee \langle\pi_2\rangle\top)||_I = ||\langle\pi_1 \cup \pi_2\rangle\varphi||_I$
iff

$||\varphi||_{||\langle\pi_1\rangle\top\vee\langle\pi_2\rangle\top||_I} = ||\langle\pi_1 \cup \pi_2\rangle\varphi||_I$
iff
$||\varphi||_{||\langle\pi_1\rangle\top||_I\cup||\langle\pi_2\rangle\top||_I} = ||\langle\pi_1 \cup \pi_2\rangle\varphi||_I$
iff
$||\varphi||_{[\![\pi_1]\!]I\cup[\![\pi_2]\!]I} = ||\langle\pi_1 \cup \pi_2\rangle\varphi||_I$
iff true.

Axiom 12:
$||\varphi\downarrow[\pi]\bot \leftrightarrow \langle\neg\pi\rangle\varphi||_I = I$
iff
$||\varphi\downarrow[\pi]\bot||_I = ||\langle\neg\pi\rangle\varphi||_I$
iff
$||\varphi||_{||[\pi]\bot||_I} = ||\langle\neg\pi\rangle\varphi||_I$
iff
$||\varphi||_{I-[\![\pi]\!]I} = ||\langle\neg\pi\rangle\varphi||_I$
iff true.

Axiom 13:
$||(\Diamond\langle\pi\rangle\top \wedge \varphi) \leftrightarrow \langle\mathbf{might}\pi\rangle\varphi||_I = I$
iff
$||\Diamond\langle\pi\rangle\top \wedge \varphi||_I = ||\langle\mathbf{might}\pi\rangle\varphi||_I$
iff
$||\Diamond\langle\pi\rangle\top||_I \cap ||\varphi||_I = ||\langle\mathbf{might}\pi\rangle\varphi||_I$
iff
if $||\langle\pi\rangle\top||_I \neq \emptyset$ then $||\langle\mathbf{might}\pi\rangle\varphi||_I = ||\varphi||_I$,
otherwise $||\langle\mathbf{might}\pi\rangle\varphi||_I = \emptyset$.
iff
if $[\![\pi]\!]I \neq \emptyset$ then $||\langle\mathbf{might}\pi\rangle\varphi||_I = ||\varphi||_I$,
otherwise $||\langle\mathbf{might}\pi\rangle\varphi||_I = \emptyset$.
iff (semantic clause for $\mathtt{might}\pi$) true.

This establishes that all axiom schemata are valid. We now check the validity of the rules of inference.

Rule 1:
Observe that if for all I, $||\varphi||_I = I$, then for all I, $||\Box\varphi||_I = I$.

Rule 2:

If for all I, $||\varphi||_I = I$, then for all I, $||[\pi]\varphi||_I = (I - [\![\pi]\!]I) \cup ||\varphi||_{[\![\pi]\!]I} = (I - [\![\pi]\!]I) \cup [\![\pi]\!]I = I$.

Rule 3:

If for all I, $||\varphi \rightarrow \psi||_I = I$ $(*)$ and for all I, $||\varphi||_I = I$ $(**)$, then for all I, $||\varphi|||I \subseteq ||\psi||_I$ (from $*$) and thus, by $**$, for all I, $I \subseteq ||\psi||_I$, i.e., $||\psi||_I = I$.

This concludes the checking of the inference rules and the soundness proof. ■

8. COMPLETENESS OF THE CALCULUS

THEOREM 13. *The calculus is complete, i.e., for all mal$_P$ formulae φ, if $\models \varphi$ then $\vdash \varphi$.*

Proof. First observe that the following translation function $*$ from *mal$_P$* to *ml$_P$* preserves validity.

$$\begin{array}{ll}
(\varphi \wedge \psi)^* & = \varphi^* \wedge \psi^* \\
(\neg\varphi)^* & = \neg\varphi^* \\
(\Diamond\varphi)^* & = \Diamond\varphi^* \\
(\langle\bot\rangle\varphi)^* & = \bot \\
(\langle p\rangle\varphi)^* & = \varphi^*{\downarrow}p \\
(\langle\pi_1;\pi_2\rangle\varphi)^* & = (\langle\pi_1\rangle\langle\pi_2\rangle\varphi)^* \\
(\langle\pi_1 \cup \pi_2\rangle\varphi)^* & = \varphi^*{\downarrow}((\langle\pi_1\rangle\top)^* \vee (\langle\pi_2\rangle\top)^*) \\
(\langle\neg\pi\rangle\varphi)^* & = \varphi^*{\downarrow}([\pi]\bot)^* \\
(\langle\mathbf{might}\pi\rangle\varphi)^* & = (\Diamond(\langle\pi\rangle\top)^* \wedge \varphi^*)
\end{array}$$

Thus, it follows from $\models \varphi$ that $\models \varphi^*$. Next, use the completeness of S5 to conclude from $\models \varphi^*$ that $\vdash \varphi^*$. Finally, note that the translation steps and their inverses in the definition of $*$ are licensed by Schema 8 through Schema 13 of the calculus. This allows us to conclude from $\vdash \varphi^*$ that $\vdash \varphi$. ■

9. REASONING ABOUT UPDATE LOGIC VIA S5

Just for the record we mention a fact about update logic which follows immediately from our 'reduction to S5' (but note that this fact was already proved in [2]).

THEOREM 14. *Update logic is decidable.*

Proof. The decision problem for update logic is the question: which $\pi \in L_P$ have the property that they are valid (accepted in every input state I)? In other words: which π have the property that for all I it holds that $[\![\pi]\!](I) = I$? The decision procedure for π is as follows. Use the definition of **wp** to find $\mathbf{wp}(\pi, \top)$. By Lemma 6 we know that $[\![\pi]\!](I) = ||\mathbf{wp}(\pi, \top)||_I$, so the decision problem for π reduces to the question whether $\mathbf{wp}(\pi, \top)$ is S5-valid. Use the decision procedure for S5 to settle this question. ∎

In update logic there is a distinction between acceptable and accepted information, witness the following definition.

DEFINITION 13.

1. A program π is accepted in I if $I = [\![\pi]\!](I)$.
2. A program π is acceptable in I if $[\![\pi]\!](I) \neq \emptyset$.

It is the universal version of the first of these which is taken as the notion for universal validity, but one might consider the universal version of the second one just as well.

DEFINITION 14.

1. A program π is always accepted (or valid) if for all I it holds that $[\![\pi]\!](I) = I$.
2. A program π is always acceptable if for all $I \neq \emptyset$ it holds that $[\![\pi]\!](I) \neq \emptyset$.

An obvious question suggests itself: are the notions of being always accepted and being always acceptable equivalent? Using the S5 connection it is easy to see that they are not and to clarify the relation between them . We need not concern ourselves with the case of $I = \emptyset$, for the elimination lemma forces $[\![\pi]\!](\emptyset) = \emptyset$ for every π. Thus, there is no harm in adopting the usual convention that S5 models have a non-empty set of worlds. The 'static' version of π *is always accepted* is (4).

(4) $S5 \models \mathbf{wp}(\pi, \top)$.

The 'static' version for π *is always acceptable*, on the other hand, is (5). Note that this translation hinges on the assumption of non-emptyness of S5 models.

(5) $S5 \models \Diamond\mathbf{wp}(\pi, \top)$.

So the notion of being always acceptable is decidable as well, but it does not coincide with the notion of being always accepted. Indeed, we have that $S5 \models \varphi$ implies $S5 \models \Diamond\varphi$, because of the reflexivity of accessibility, so (4) implies (5), but not the other way around. Take φ equal to $\Diamond p \rightarrow p$ for a simple counterexample. We have $S5 \not\models \Diamond p \rightarrow p$ (take a non-p world in a model containing both p and non-p worlds), but $S5 \models \Diamond(\Diamond p \rightarrow p)$. To see this latter fact, take an arbitrary w in an arbitrary universal S5 model I. If there are no p worlds, then $I, w \Vdash \Diamond(\Diamond p \rightarrow p)$; if there are p worlds, then there is a p world w' for which $I, w' \models \Diamond p \rightarrow p$, so by the fact that accessibility is universal again $I, w \Vdash \Diamond(\Diamond p \rightarrow p)$. Note, by the way, that $\Diamond(\Diamond p \rightarrow p)$ is the modal counterpart of a predicate logical sentence that philosophical logicians sometimes refer to as 'Plato's principle': $\exists x(\exists x Px \rightarrow Px)$.

The S5 counterexample can be transposed to update logic, of course: $p \cup \neg\mathbf{might}\ p$ is an example of a program which is always acceptable but not always accepted.

For a next illustration of reasoning about update logic via S5 we take a quick look at valid consequence in update logic. In his paper [22] Veltman discusses various notions of valid consequence. He distinguishes the following three definitions.

DEFINITION 15.

1. $\pi_1 \models_1 \pi_2$ if for all I it holds that $[\![\pi_1]\!](I) = I$ implies $[\![\pi_2]\!](I) = I$.
2. $\pi_1 \models_2 \pi_2$ if for all I it holds that $[\![\pi_1]\!](I) = [\![\pi_2]\!]([\![\pi_1]\!](I))$.

3. $\pi_1 \models_3 \pi_2$ if $[\![\pi_1]\!](W) = [\![\pi_2]\!]([\![\pi_1]\!](W))$.

The following proposition reduces these notions to S5.

PROPOSITION 15.

1. $\pi_1 \models_1 \pi_2$ *iff S5* $\vdash \Box\mathbf{wp}(\pi_1, \top) \rightarrow \Box\mathbf{wp}(\pi_2, \top)$.
2. $\pi_1 \models_2 \pi_2$ *iff S5* $\vdash \mathbf{wp}(\pi_1, \top) \leftrightarrow \mathbf{wp}(\pi_1; \pi_2, \top)$.
3. $\pi_1 \models_3 \pi_2$ *iff S5* $\vdash (\bigwedge\Diamond\varphi_i) \rightarrow (\mathbf{wp}(\pi_1, \top) \leftrightarrow \mathbf{wp}(\pi_1; \pi_2, \top))$, *where the* φ_i *are all conjunctions of the form* $(\neg)p_1 \wedge \cdots \wedge (\neg)p_n$, *with* $p_1, \ldots, p_n$ *the list of proposition letters occurring in* π_1 *or* π_2.

Proof. The first item:

For all I: $[\![\pi_1]\!](I) = I$ *implies* $[\![\pi_2]\!](I) = I$
iff for all I: if for all $w \in I$*:* $I, w \Vdash \mathrm{wp}(\pi_1, \top)$
then for all $w \in I$*:* $I, w \Vdash \mathrm{wp}(\pi_2, \top)$
iff for all I: $I \models \Box\mathrm{wp}(\pi_1, \top) \rightarrow \Box\mathrm{wp}(\pi_2, \top)$
iff $S5 \vdash \Box\mathbf{wp}(\pi_1, \top) \rightarrow \Box\mathbf{wp}(\pi_2, \top)$.

The second item is immediate from the definitions of the validity notions, the **wp** adequacy lemma and the completeness of S5.

For the third item, note that $I \models \bigwedge\Diamond\varphi_i$ (where $\models$ denotes S5 validity) for precisely those information sets I that express total ignorance with respect to all proposition letters in π_1 and π_2, i.e., for the sets I that are indistinguishable from W as far as π_1 and π_2 are concerned. The S5 formula expresses that for such I, all worlds in $[\![\pi_1; \pi_2]\!](I)$ are worlds in $[\![\pi_2]\!](I)$ and vice versa. This is precisely what the validity notion $\models_3$ expresses. ■

As Willem Groeneveld pointed out to us, this reduction to S5 can be simplified somewhat by defining $\mathrm{wp}(\pi, \top)$ directly, as follows.

DEFINITION 16.

1. $\mathbf{wp}(\bot, \top) = \bot$.
2. $\mathbf{wp}(p, \top) = p$.
3. $\mathbf{wp}(\pi_1; \pi_2, \top) = \mathbf{wp}(\pi_2, \top){\downarrow}\mathbf{wp}(\pi_1, \top)$.
4. $\mathbf{wp}(\pi_1 \cup \pi_2, \top) = \mathbf{wp}(\pi_1, \top) \vee \mathbf{wp}(\pi_2, \top)$.

5. $\mathbf{wp}(\neg\pi, \top) = \neg\mathbf{wp}(\pi, \top)$.
6. $\mathbf{wp}(\mathbf{might}\,\pi, \top) = \Diamond\mathbf{wp}(\pi, \top)$.

10. CALCULATIONS OF CONSISTENCY

Veltman calls a program π of L_P consistent if there is some information state I for which $[\![\pi]\!](I) \neq \emptyset$. Intuitively, consistent programs are programs that can be used to convey information. By the soundness of the Hoare/Pratt calculus, consistency of an update program π boils down to the question whether there is some S5 consistent $\varphi \in ml_P$ such that $\vdash \varphi \leftrightarrow \langle\pi\rangle top$. We illustrate how to check consistency for two examples taken from Veltman [22]. We calculate with WPs, but by virtue of the fact that WP reasoning and NSC reasoning are equivalent (Theorem 10), calculations with NSCs work just as well.

EXAMPLE 1. **might** $p; \neg p$ *is consistent.*

Proof.

$$\begin{aligned}
\mathrm{wp}(\mathbf{might}\,p; \neg p\,, \top) &= \mathrm{wp}(\mathbf{might}\,p; \mathrm{wp}(\neg p\,, \top)) \\
&= \mathrm{wp}(\mathbf{might}\,p; \neg\mathrm{wp}(p, \top)) \\
&= \mathrm{wp}(\mathbf{might}\,p; \neg p) \\
&= \Diamond p \wedge \neg p
\end{aligned}$$

Since $\Diamond p \wedge \neg p$ does have S5 models, so it is not S5-provably equivalent to $\bot$. ■

EXAMPLE 2. $\neg p$; **might** p *is not consistent.*

Proof.

$$\begin{aligned}
\mathrm{wp}(\neg p\,;\ \mathbf{might}\,p, \top) &= \mathrm{wp}(\neg p; \mathrm{wp}(\mathbf{might}\,p, \top)) \\
&= \mathrm{wp}(\neg p, \Diamond\mathrm{wp}(p, \top)) \\
&= \mathrm{wp}(\neg p, \Diamond p) \\
&= \Diamond p{\downarrow}\neg\mathrm{wp}(p, \top) \\
&= \Diamond p{\downarrow}\neg p
\end{aligned}$$

$$\begin{aligned}&= \neg p \wedge \Diamond(p \wedge (p{\downarrow}\neg p))\\&= \neg p \wedge \Diamond(p \wedge \neg p))\\&= \neg p \wedge \bot\\&= \bot\end{aligned}$$

■

Of course, these results can also be established in our weakest precondition Hoare/Pratt calculus. By using the matching axiom schemata we derive that $\vdash (\Diamond p \wedge \neg p) \leftrightarrow \langle \textbf{might}\, p;\ \neg p\rangle\top$, and that $\vdash (\Diamond p{\downarrow}\neg p) \leftrightarrow \langle \neg p;\ \textbf{might}\, p\rangle\top$. In short, by our construction of a calculus for update logic we claim to have established a clean connection between a species of information flow logic and good old static S5.

11. CONCLUSION

There is scope for quite a bit of further work. In the first place, one could explore Veltman's extended versions of update logic in the same spirit. More specifically, a modal study of the preference relation on information states that Veltman proposes seems to be worthwhile: this would lead to a link to a trimodal system with one modal operator $\boxed{1}$ reflecting the dynamics of discarding possibilities (basically, our S5 box $\Box$), a modal operator $\boxed{2}$ interpreted in terms of the preference order on the set of all worlds (the 'normally' relation), and finally a modal operator $\boxed{3}$ interpreted in terms of the preference relation restricted to the current input information set (the 'presumably' relation). Thus, the modal perspective on defaults would use a relation of 'being as least as likely as' between worlds. $\boxed{2}\,\varphi$ (for: φ holds by default) would hold in a world $w \in W$ if in all worlds $w' \in W$ that are at least as likely as w, φ holds. $\boxed{3}\,\varphi$ (for: φ presumably holds) would hold in a world $w \in W$, given a current information set I, if in all worlds $w' \in I$ that are at least as likely as w, φ holds.

In a different direction, one may study the combination of the calculus given here with the calculus from [7] in a system of quantificational update logic satisfying the desiderata which Groenendijk and Stokhof list in [11]. In such a system one would be able to handle the combination of epistemic operators like *might* or *maybe* (the province of update logic) and pronominal bindings across sentence boundaries (the key application area of dynamic predicate logic and dynamic Montague grammar [10]), as in the following example sentence.

(6) *A man walked out. Maybe he was angry.*

A suitable representation medium for such examples is a system of dynamic assignment logic with epistemic modalities. Such a system is developed in Van Eijck and Cepparello [6] and axiomatized in a similar way to the approach of the present paper, but now with modal predicate logic instead of modal propositional logic as assertion language.

The more general moral of the paper, however, is in the demonstration that techniques from theoretical computer science can be applied fruitfully to information logic, broadly conceived. A dynamic logic in the spirit of Hoare and Pratt geared to this application was proposed by Van Benthem in [5], and worked out further in De Rijke [20].

In this logic there is an explicit modality $\sqsubseteq$ for 'becoming more specific about what on assumes to be the case', or 'increasing one's information'. The process of expanding one's set of assumptions to make it include φ, for example, is given by the program $\sqsubseteq; \varphi?$. The process of purging one's set of assumptions to take one back to a state where φ fails is given by $\sqsubseteq^{\smile}; \neg\varphi?$ (here $^{\smile}$ is the operator which takes a program to its converse).

Dynamic modal logics have a procedural part and a propositional part which are connected by modes (expand to φ, retract to φ, test for φ) and projections (being in the domain of π, being in the range of π, being a fixpoint for π). As is demonstrated in De Rijke [19], such systems can be used to analyse information logics which have operations for both 'updating' and 'downdating' (retracting information). The central point of their use remains the Hoare/Pratt style analysis of the connection between procedural notions (properties of programs) and static notions (properties of states), in the spirit that was demonstrated above.

ACKNOWLEDGEMENT

This paper has benefited from helpful comments by Johan van Benthem, Giovanna Cepparello, Tim Fernando, Willem Groeneveld, Marcus Kracht, Frank Veltman, Kees Vermeulen, Jørgen Villadsen, Albert Visser, and an anonymous referee of this journal. Thank you all.

REFERENCES

1. J. Barwise. Noun phrases, generalized quantifiers and anaphora. In P. Gärdenfors, editor, *Generalized Quantifiers: linguistic and logical approaches*, pages 1–30. D. Reidel Publishing Company, Dordrecht, 1987.
2. J. van Benthem. Semantic parallels in natural language and computation. In H.-D. Ebbinghaus et al., editors, *Logic Colloquium, Granada, 1987*, pages 331–375, Amsterdam, 1989. Elsevier.
3. J. van Benthem. General dynamics. *Theoretical Linguistics*, 17:159–201, 1991.
4. J. van Benthem. *Language in Action: categories, lambdas and dynamic logic*. Studies in Logic 130. Elsevier, Amsterdam, 1991.
5. J. van Benthem. Logic and the flow of information. Technical Report LP-91-10, ILLC, University of Amsterdam, 1991.
6. J. van Eijck and G. Cepparello. Dynamic modal predicate logic. Technical Report OTS-WP-CL-93-005, OTS, Utrecht, October 1993. Also in M. Kanazawa and C.J. Piñon (eds.), *Dynamics, Polarity, and Quantification*, CSLI, Stanford 1994.
7. J. van Eijck and F.J. de Vries. Dynamic interpretation and Hoare deduction. *Journal of Logic, Language, and Information*, 1:1–44, 1992.
8. P. Gärdenfors. *Knowledge in Flux: Modelling the Dynamics of Epistemic States*. MIT Press, 1988.
9. R. Goldblatt. *Logics of Time and Computation, Second Edition, Revised and Expanded*, volume 7 of *CSLI Lecture Notes*. CSLI, Stanford, 1992 (first edition 1987). Distributed by University of Chicago Press.
10. J. Groenendijk and M. Stokhof. Dynamic predicate logic. *Linguistics and Philosophy*, 14:39–100, 1991.
11. J. Groenendijk and M. Stokhof. Two theories of dynamic semantics. In J. van Eijck, editor, *Logics in AI—European Workshop JELIA '90*, pages 55–64, Berlin, 1991. Springer Lecture Notes in Artificial Intelligence.
12. I. Heim. *The Semantics of Definite and Indefinite Noun Phrases*. PhD thesis, University of Massachusetts, Amherst, 1982.
13. C.A.R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):567–580, 583, 1969.
14. H. Kamp. A theory of truth and semantic representation. In J. Groenendijk *et al.*, editors, *Formal Methods in the Study of Language*. Mathematisch Centrum, 1981.

15. L. Karttunen. Discourse referents. In J. McCawley, editor, *Syntax and Semantics 7*, pages 363–385. Academic Press, 1976.
16. M. Kracht. Splittings and the finite model property. *Journal of Symbolic Logic*, 58, 1993, pp 139–157.
17. D. Lewis. Score keeping in a language game. *Journal of Philosophical Logic*, 8:339–359, 1979.
18. V. Pratt. Semantical considerations on Floyd–Hoare logic. *Proceedings 17th IEEE Symposium on Foundations of Computer Science*, pages 109–121, 1976.
19. M. de Rijke. Meeting some neighbours. Technical Report LP-92-10, ILLC, University of Amsterdam, 1992. Also in Van Eijck and Visser (eds.), *Logic and Information Flow*, MIT Press, 1994.
20. M. de Rijke. A system of dynamic modal logic. Technical Report LP-92-08, ILLC, University of Amsterdam, 1992.
21. R. Stalnaker. Pragmatics. In D. Davidson and G. Harman, editors, *Semantics of Natural Language*, pages 380–397. Reidel, 1972.
22. F. Veltman. Defaults in update semantics. Technical report, Department of Philosophy, University of Amsterdam, 1991. To appear in the Journal of Philosophical Logic.

Jan van Eijck
CWI, P.O. Box 4079,
1009 AB Amsterdam
The Netherlands

Fer-Jan de Vries,
NTT Laboratories,
Hikaridai, Seika-cho,
Soraku-gun, Kyoto 619–02,
Japan