Dynamic Modal Predicate Logic*

Jan van Eijck[1,2] & Giovanna Cepparello[1,3]

[1] CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands

[2] OTS, Trans 10, 3512 JK Utrecht, The Netherlands

[3] Scuola Normale Superiore, Piazza dei Cavalieri 7, 56100 Pisa, Italy

1 Introduction

The semantic processing of pieces of natural language discourse can be viewed as a continuous process of linguistic updating: every new bit of information changes the context that forms the background against which the next bit of information is to be interpreted. This process of linguistic updating has several dimensions, as was stressed by David Lewis (1979) when he compared this process with the keeping of a score-board while the 'language game' moves on.

Technically, the program of giving a dynamic 'blow by blow' account of the scorekeeping process, has been worked out in several directions, for several dimensions of the score-table. One of the logical tools with which this score-keeping aspect has been analysed in recent years is dynamic logic. Van Benthem (1991a, 1991b) gives a general appraisal of the use of the toolbox of dynamic logic in natural language analysis.

Let us look at some of the dimensions involved. For the dimension of binding ('What do the pronouns refer to that are used at the present stage in the discourse?'), the task of providing an analysis with dynamic logic was taken up by Barwise (1987). Groenendijk and Stokhof (1991a) have worked this out by proposing the elegant framework of dynamic predicate logic. Epistemic aspects of natural language use are analysed dynamically in Gärdenfors (1988) and Veltman (1991), where

*The second author was sponsored by a travel grant from project NF 102/62-356 ('Structural and Semantic Parallels in Natural Languages and Programming Languages'), funded by the Netherlands Organization for the Advancement of Research (N.W.O.).

the score with respect to the question 'Which states of the world are still compatible with what has been conveyed in the discourse so far?' is analysed dynamically. Other dimensions could be analysed in the same spirit, for instance: 'Who is the current speaker?', 'Who is the current addressee?', 'What are the boundaries of the current universe of discourse?', 'What is the current reference point in time?', and so on. A look at Lewis' score-keeping paper will convince the reader that the list of possible dimensions is virtually limitless. The most promising strategy seems to us to first develop dynamic accounts of score-keeping in the individual dimensions, and then devise general strategies for combining those accounts.

Our goal in this paper is to combine within the same logic the dynamic account of variable binding from Groenendijk and Stokhof (1991a) with the dynamic account of epistemic updating from Veltman (1991), thus combining the useful features of Dynamic Predicate Logic (DPL) with those of Update Logic (UL). At the end of the paper we will briefly look at further extensions along other score-keeping dimensions.

The DPL features provide a compositional treatment of anaphoric binding, while UL provides us with a treatment of epistemic modalities. By combining the two, our logic provides a suitable framework for the representation of natural language texts involving unbound anaphora and epistemic operators, and the interplay between those. Consider the following example texts.

(1) A man walked out. Maybe he was angry.

(2) If a man walks out, then maybe he is angry.

The semantic analysis of these example texts poses a combination of two problems. The pronoun 'he' must be linked to its antecedent; in the first example this is difficult because the antecedent is in a different sentence, while the second example poses the problem of getting the universal reading for the antecedent together with the intended anaphoric link.

The adverb 'maybe' intuitively acts as a consistency check on the piece of discourse that it has scope over. Its use in the two example texts above makes intuitive sense, but the next example illustrates that it can also serve to rule out anaphoric links.

(3) Maybe a man walked out. *He was angry.

Intuitively, the continuation with 'he' does not make sense here, as the discourse does not provide a suitable antecedent. Our aim in this paper is to develop a logic that gives an account of the interaction of the processes of anaphoric linking and epistemic updating, and then look at further possible extensions.

A further dimension for score-keeping could be the dimension of epistemic preference for certain states of affairs over others. This is needed for the semantic analysis of sentences like (4).

(4) If a man walks out, then usually he is angry.

The dynamics of preference is analysed in Van Benthem, Van Eijck and Frolova (1993); this paper also analyses its relation to epistemic updating.

Next, take the dynamics of the process of information exchange in a conversation between speakers A and B, as in the following example.

(5) A: A man walked out.

B: He was angry.

A: He wasn't angry. Maybe he was drunk.

B: He was not drunk.

Here we have a case of updating the score in the dimension of role switching between speaker and addressee in a piece of discourse. Again, there is an interaction with the other dimensions: discourse (5) illustrates that anaphoric links carry over when speaker and hearer switch roles, but epistemic context does not carry over: the same discourse when uttered by a single speaker would be inconsistent. We will return to this example at the end of the paper.

As it is clear that the list of relevant dynamic dimensions of score-keeping can be extended *ad libitum*, it does not make sense to develop dynamic logics that give an exhaustive account of score-keeping in a language game. It does make eminent sense, on the other hand, to develop strategies for combining dynamic logics for various aspects of score-keeping.

We will now quickly review the two example systems we are going to combine, dynamic predicate logic and update logic. Next we present our proposal for combining them, and we show that our approach leads to a natural dynamic version of quantified modal logic. At the end of the paper we provide some pointers to extending the account to the dimension of the dynamics of speaker switches and listener agreement in conversation.

2 Dynamic Predicate Logic

Dynamic predicate logic is just like predicate logic, except for the fact that existential quantification is replaced by random assignment, and that the static semantics in terms of one variable assignment gets replaced by a dynamic semantics, in terms of a relation between input and output variable assignments.

DPL $\pi := Rt\cdots t \mid t = t \mid v :=? \mid \pi;\pi \mid \neg\pi$.

Here t ranges over terms, v over variables, and R over relation symbols of the language. ; denotes sequential composition, and $v :=?$ random assignment of a value to variable v.

The appropriate notion of 'state' for DPL is: variable assignment function in a first order model. The semantics for DPL can be given as a function from sets of states to sets of states, but this is less than illuminating, as the semantics is relational at the single state level. If one insists on a functional formulation, the 'distribution lemma' of Groenendijk and Stokhof (1991b) formulates a trivial consequence of the fact that the set-level functional definition of the semantics in a reformulation of a single-state relational definition:

$$[\![\pi]\!]_{\mathcal{M}}(S) = \bigcup_{s\in S}[\![\pi]\!]_{\mathcal{M}}(\{s\})$$

A functional semantics at the single state level is also possible, but it takes the form of a characteristic function ${}_s[\![\cdot]\!]_u^{\mathcal{M}} : DPL \to \{0,1\}$, where $\mathcal{M} = \langle M, F\rangle$ is a first order model, s is the input state, and u the output state. If V is the set of variables of the language, then M^V is the set of states. If $s \in M^V$ and $d \in M$, then $s(x|d)$ is the state which is like s except for the possible difference that x is mapped to d. For any formula A of the form $Rt_1\cdots t_n$ or $t_1 = t_2$ the relation $\mathcal{M} \models_s A$ (for: A is true in $\mathcal{M}$ under assignment s) is defined in the usual way.

The interpretation function ${}_s[\![\cdot]\!]_u^{\mathcal{M}} : DPL \to \{0,1\}$ is defined as follows.

1. ${}_s[\![Rt_1\cdots t_n]\!]_u^{\mathcal{M}} = 1$ iff $s = u$ and $\mathcal{M} \models_s Rt_1\cdots t_n$.
2. ${}_s[\![t_1 = t_2]\!]_u^{\mathcal{M}} = 1$ iff $s = u$ and $\mathcal{M} \models_s t_1 = t_2$.
3. ${}_s[\![v :=?]\!]_u^{\mathcal{M}} = 1$ iff $s = s(x|d)$ for some $d \in M$.
4. ${}_s[\![\pi_1;\ \pi_2]\!]_u^{\mathcal{M}} = 1$ iff there is an r with ${}_s[\![\pi_1]\!]_r^{\mathcal{M}} = 1$ and ${}_r[\![\pi_2]\!]_u^{\mathcal{M}} = 1$.
5. ${}_s[\![\neg\pi]\!]_u^{\mathcal{M}} = 1$ iff $s = u$ and there is no r with ${}_s[\![\pi]\!]_r^{\mathcal{M}} = 1$.

Dynamic implication between programs is defined in terms of negation and sequential composition. $\pi_1 \Rightarrow \pi_2$ abbreviates $\neg(\pi_1;\ \pi_2)$. This gives the following semantics for dynamic implication:

${}_s[\![\pi_1 \Rightarrow \pi_2]\!]_u^{\mathcal{M}} = 1$ iff

$s = u$ and

for all r with ${}_s[\![\pi_1]\!]_u^{\mathcal{M}} = 1$ there is a p with ${}_u[\![\pi_2]\!]_p^{\mathcal{M}} = 1$.

We say that π is successful in $\mathcal{M}$ if there are s, u with ${}_s[\![\pi]\!]_u^{\mathcal{M}} = 1$. Program π_1 dynamically entails conclusion π_2 (notation $\pi_1 \models \pi_2$) if for every $\mathcal{M}$ and every $s, u \in M^V$ we have that ${}_s[\![\pi_1]\!]_u^{\mathcal{M}} = 1$ implies that there is a $p \in M^V$ with ${}_u[\![\pi_2]\!]_p^{\mathcal{M}} = 1$. Here are some examples of valid dynamic consequences (we abbreviate $x :=?; \pi$ as $\eta x : \pi$):

(6) $\eta x : man\ x \models man\ x.$

(7) $\eta x : man\ x;\ walk\ x \models \eta y : man\ y;\ walk\ y.$

(8) $(\eta x : man\ x;\ \eta y : woman\ y;\ love(x, y) \Rightarrow kiss(x, y));$
$man\ z;\ woman\ w;\ love(z, w) \models kiss(z, w).$

Here is an example of a non-valid dynamic consequence.

(9) $\eta x : man\ x \not\models man\ y.$

Dynamic meaning for DPL is called *non eliminative* in Groenendijk and Stokhof (1991b), because some DPL statements do indeed *change* variable states, rather than just weed them out.

3 Epistemic Update Logic

Following Veltman (1991), we define the language of epistemic update logic over a set of proposition letters P as follows:

UL $\pi ::= p \mid \pi; \pi \mid \pi \cup \pi \mid \neg\pi \mid \Diamond\pi.$

The semantics of UL is again given in terms 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 index set. Progams are interpreted as functions from index sets to index sets, i.e., as functions in $\mathcal{P}W \rightarrow \mathcal{P}W$. The clauses are as follows:

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

Note that negation is Boolean complement. Intuitively, a program of the form $\Diamond\pi$ does not provide information about the world but about available information. A program $\Diamond\pi$ is true, given an index set I, if there is at least one world $w \in I$ for which π is true in the sense that

$w \in [\![\pi]\!](I)$. If such a w can be found, the output index set of $\Diamond\pi$ is equal to its input index set; this agrees with the intuition that $\Diamond\pi$ 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 index set of $\Diamond\pi$ equals $\emptyset$.

The UL counterpart of the DPL notion of success is the notion of acceptance:

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

A program π is accepted in I if $I = [\![\pi]\!](I)$.

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

On this basis, we can define the following notion of entailment for UL; premiss π_1 entails conclusion π_2 (notation $\pi_1 \models \pi_2$) if for all I it holds that:

(10) $$[\![\pi_1]\!](I) = [\![\pi_2]\!]([\![\pi_1]\!](I)).$$

Note that the semantic clause for $\Diamond\pi$ refers to the full index set I. This introduces an element of *non distributivity* into the semantics, in the sense that unions of input states do not distribute over output states. In other words, a pointwise definition of the semantics will not work for UL, as there are π, I such that (11).

(11) $$[\![\pi]\!](I) \not\subseteq \bigcup_{i \in I} [\![\pi]\!](\{i\}).$$

Take for example π equal to $\Diamond p$ and let $I = \{w, w'\}$ with $p \in w$ and $p \notin w'$. Then $[\![\Diamond p]\!](\{w\}) \cup [\![\Diamond p]\!](\{w'\}) = \{w\}$, but $[\![\Diamond p]\!](I) = I = \{w, w'\}$.

On the other hand, a simple induction on the complexity of π shows that Lemma 1 holds for all expressions π of UL and all index sets I. In the terminology of Groenendijk and Stokhof (1991b): epistemic update logic is *eliminative*.

Lemma 1 (Elimination Lemma) *For all programs π of Update Logic and all index sets I: $[\![\pi]\!](I) \subseteq I$.*

4 Epistemic Modalities in Dynamic Predicate Logic

The two attempts that we know of get a system with the combined properties of DPL and UL both proceed by first getting rid of the 'context change potential' of DPL (by making DPL eliminative) and then combining the result with UL to create an eliminative logic. There are two ways of making DPL eliminative: curtailing the syntax and changing the semantics. The approach pursued by Dekker (1993) is to first curtail DPL to get an eliminative system, and then add a modal operator to the result. The disadvantage of this is that it imposes a rather unnatural restriction on the language: new assignments to already 'active' variables are forbidden. The formula, $\eta x : Px;\ \eta x : \Diamond Qx$, for instance, is not interpretable in Dekker's eliminative dynamic modal predicate logic. Vermeulen (1993) takes the other tack. Here DPL is made eliminative by developing a sequence semantics that remembers all old values of variables. This is the method of changing the semantics. Our objection to this is that the original DPL semantics was rather nice and simple.

It is not an accident, by the way, that both approaches to the problem of combining DPL and UL have concentrated on changing the features of DPL. Tampering with the non distributivity of UL does not make much sense, as non distributivity is the essence of the epistemic notion *maybe* that UL tries to analyse.

In our combined system, we want *might* to range over different possible states of affairs rather than different possible assignments. On the other hand, it is clear that random assignment ($v :=?$) should tell us something about the assignment of objects to variables. Thus, a harmonious marriage of DPL and UL seems to call for a distinction of levels or dimensions. The solution we are going to propose is the following: *might* will range over 'possible worlds', i.e., over interpretation functions over a universe, while $v :=?$ will range over assignments. In this way we will obtain a layered reconciliation of the pointwise relational, non eliminative character of DPL with the non distributive and eliminative character of UL. Our approach can be seen as an example of playing with Tarskian variations in the sense of van Benthem (1991a).

Given a set of first order models then, we need to take into account two 'dimensions' at the same time: alternative interpretations and changing assignments. Programs of the form $\Diamond\pi$ will look at the behaviour of π in different worlds, given an assignment s. Intuitively, $\Diamond\pi$ is acceptable in a set of worlds I, given an input assignment s if there is a world $w \in I$ where evaluation of the program π with input assignment s is successful. On the other hand, $v :=?$ programs will modify assignments, leaving the possible world context untouched.

This intuitive account leads to a notion of evaluation at triples consisting of an *index set* of possible worlds and pair of variable assignments. To give a functional definition of the semantics we define a function $[\![\cdot]\!]$ from index sets to index sets, given an input assignment and an output assignment.

Summing up, we make a distinction between the *index set* level, the level of alternative possible worlds which constitute a frame for the interpretation of modalities, and the *assignment* level, the level which constitutes the frame for the dynamic interpretation of assignment statements. Evaluation at the level of index sets is eliminative, and evaluation at the level of assignment functions takes place in a point-wise relational fashion.[1]

5 Semantics for DMPL

The syntax of DMPL is like the syntax of DPL, but with a construction for epistemic 'might' added:

DMPL $\pi := Rt \cdots t \mid t = t \mid v :=? \mid \pi; \pi \mid \neg\pi \mid \Diamond\pi$.

We evaluate with respect to sets of first order models over the same universe M, i,e., we consider a DMPL model $\mathcal{M}$ as a pair $\langle M, W \rangle$ where W is a set of first order interpretations over M. Variable assignments for $\mathcal{M}$ are elements of M^V. Index sets for $\mathcal{M}$ are subsets of W.

We give a functional semantics that maps triples consisting of an index set, an input assignment and an output assignment to a new index set. Intuitively, ${}^{\mathcal{M}}_{s}[\![\pi]\!]^{I}_{u} = J$ means that given input assignment s and output assignment u for $\mathcal{M}$, π maps index set I to index set J. Suppressing the parameter $\mathcal{M}$ for ease of reading, we can express the same as $[\![\pi]\!]^{s}_{u}(I) = J$. The advantage of this functional formulation is that it clearly shows the two dimensions of parametric variation, the dimension of assignments and the dimension of index sets, with their interplay.

The index set W pictures the case of complete ignorance. Of course, even in this case one has to make up one's mind about what the pronouns (free variables) are supposed to refer to, both at the start and at the end of the processing, i.e., one has to fix an input and an output assignment function to be able to compute an output index set. If the discourse gives new information, the index set that results from the semantic processing will be a proper subset of the initial index set. A state of maximal information is given by an index set $\{i\}$, indicating that i is the only state of affairs compatible with one's information. The index set $\emptyset$ pictures the case of inconsistent information: no state of affairs at all is compatible with what one knows or believes.

Definition 1 (Semantics of DMPL)

[1] We must add that a similar solution was independently noticed by Paul Dekker, who briefly discusses the 'least common product' of DPL and UL in one of the draft versions for his thesis (Dekker (1993)), although it seems to have dropped out of the final version for some reason.

1. $[\![Rt_1 \cdots t_n]\!]^s_u(I) = \{i \in I \mid s = u \textit{ and } M, i \models_s Rt_1 \cdots t_n\}$.

2. $[\![t_1 = t_n]\!]^s_u(I) = \{i \in I \mid s = u \textit{ and } M, i \models_s t_1 = t_2\}$.

3. $[\![\pi_1;\ \pi_2]\!]^s_u(I) = \{i \in I \mid \textit{there is an } r \textit{ with } i \in [\![\pi_2]\!]^r_u([\![\pi_1]\!]^s_r(I))\}$.

4. $[\![\neg\pi]\!]^s_u(I) = \{i \in I \mid s = u \textit{ and there is no } r \textit{ with } i \in [\![\pi]\!]^s_r(I)\}$.

5. $[\![v := ?]\!]^s_u(I)$

$$= \{i \in I \mid u = s(x|d) \textit{ for some } d \in M\}$$

$$= \begin{cases} I & \textit{if } u = s(x|d) \textit{ for some } d \in M, \\ \emptyset & \textit{otherwise.} \end{cases}$$

6. $[\![\Diamond\pi]\!]^s_u(I)$

$$= \{i \in I \mid s = u \textit{ and there is an } r \textit{ with } [\![\pi]\!]^s_r(I) \neq \emptyset\}$$

$$= \begin{cases} I & \textit{if } s = u \textit{ and there is an } r \textit{ with } [\![\pi]\!]^s_r(I) \neq \emptyset, \\ \emptyset & \textit{otherwise.} \end{cases}$$

The clauses for basic relations $Rt_1 \cdots t_n$ and identities $t_1 = t_2$ say that atomic predicates serve to weed out the set of indices, given what the assignment function tells us about the referents of the variables. For every input index set, the output index set simply consists of those items from the input index set that satisfy the predicate, given the assignment function, which itself remains unchanged. Note that identity is a logical relation which behaves the same in every world.

The clause for program concatenation says that the members of the output index set of the concatenated program are the output index set of the second program, when applied on the output index set of the first program for the original input index set, provided that some intermediate assignment function establishes the link between the input and the output assignment. Thus, along the index set dimension we have composition of update functions, as in UL, while along the assignment dimension we have relational composition, as in DPL.

Similarly, negation is boolean complement along the index set dimension, but computed in terms of possible assignment continuations along the assignment dimension.

As in DPL, dynamic implication $\pi_1 \Rightarrow \pi_2$, is defined as $\neg(\pi_1;\ \neg\pi_2)$. The DMPL semantics gives rise to the following clause for $\Rightarrow$:

$[\![\pi_1 \Rightarrow \pi_2]\!]^s_u(I) =$

$\{i \in I \mid s = u$ and

for all r with $i \in [\![\pi_1]\!]^s_r(I)$ there is a p with $i \in [\![\pi_2]\!]^s_p([\![\pi_1]\!]^s_r(I))\}$.

Thus, dynamic implication weeds out those indices that for some intermediate assignment function r satisfy the antecedent program but do not have an output assignment for which they also satisfy the consequent program.

The clause for random assignment to v checks whether the output assignment function is a v variant of the input assignment and returns the input index set if it does, the empty set if not.

Finally, the clause for $\Diamond$ checks whether the output assignment equals the input assignment and whether it is consistent with π.

As in UL, we have an elimination lemma:

Lemma 2 (Elimination Lemma for DMPL) *For all programs π of DMPL, all models $\mathcal{M}$, all index sets I and assignments s, u:* $[\![\pi]\!]^s_u(I) \subseteq I$.

Also, it is easy to semantically characterize the test programs. Test programs are the programs for which $[\![\pi]\!]^s_u(I) \neq \emptyset$ implies that $s = u$.

Acceptability and acceptance are defined in the spirit of UL.

Definition 2 *A program π is acceptable (in model $\mathcal{M}$) for input index set I and input assignment s if there is an u for which $[\![\pi]\!]^s_u(I) \neq \emptyset$.*

Intuitively, if the information conveyed by program π is accepted by index set I then it does not weed out any of the current epistemic alternatives:

Definition 3 *A program π is accepted (in model $\mathcal{M}$) by index set I, given input assignment s, if there is an u for which $[\![\pi]\!]^s_u(I) = I$.*

Validity is defined in terms of acceptance, as follows.

Definition 4 *A program π is valid if for all models $\mathcal{M}$, for all I, s for $\mathcal{M}$, π is accepted by I, given s.*

Notation: write $\downarrow [\![\pi]\!]^s(I)$ for $\{i \in I \mid \text{there is some } u \text{ with } i \in [\![\pi]\!]^s_u(I)\}$.

Definition 5 *π_1 dynamically entails π_2, notation $\pi_1 \models \pi_2$, if for all models $\mathcal{M}$ all index sets I and assignments s, u for $\mathcal{M}$,*

$$\downarrow [\![\pi_2]\!]^u([\![\pi_1]\!]^s_u(I)) = [\![\pi_1]\!]^s_u(I).$$

We leave it to the reader to check that this definition combines the features of DPL entailment and UL entailment, both in the appropriate dimensions.

To see that DMPL is a conservative extension of Update Logic, consider the restriction of DMPL to the update fragment, i.e., the fragment which only has 0-ary predicates and no quantification. It is easily seen that the assignment parameters in the semantics become redundant, and the definition of DMPL entailment reduces to the Update Logic definition.

If, on the other hand, we consider the DPL fragment of DMPL (all formulas without occurrences of the $\Diamond$ operator), then we may replace evaluation at an index set by evaluation at a single world, and we are back at the semantics of DPL. In this case the index set parameter becomes redundant and the definition of DMPL entailment reduces to the DPL definition.

6 Some Simple Examples

In order to gain insight in the semantics of DMPL it is useful to look at some very simple examples. Let us assume a model $\mathcal{M}$ with a set of worlds W consisting of three worlds over a universe $\{d, d'\}$, where a one place predicate P is defined as follows.

$$F_{w_1}(P) = \{d, d'\}.$$
$$F_{w_2}(P) = \{d\}.$$
$$F_{w_3}(P) = \emptyset.$$

Now suppose also we only have one variable x in the language. Then M^V consists of just two assignment functions, $x \mapsto d$ and $x \mapsto d'$, which we will refer to as s and u respectively. Here are the results of evaluating some very simple programs.

$[\![Px]\!]^s_s(W) = \{w_1, w_2\}.$

$[\![Px]\!]^u_u(W) = \{w_1\}.$

$[\![Px]\!]^s_u(W) = [\![Px]\!]^u_s(W) = \emptyset.$

$[\![\neg Px]\!]^s_s(W) = \{w_3\}.$

$[\![\neg Px]\!]^u_u(W) = \{w_2, w_3\}.$

$[\![\neg Px]\!]^s_u(W) = [\![\neg Px]\!]^u_s(W) = \emptyset$

$[\![\eta x : Px]\!]^s_s(W) = \{w_1, w_2\}.$

$[\![\eta x : Px]\!]^u_u(W) = \{w_1\}.$

$[\![\eta x : Px]\!]^s_u(W) = \{w_1\}.$

$[\![\eta x : Px]\!]^u_s(W) = \{w_1, w_2\}.$

$[\![\Diamond Px]\!]^s_s(W) = W.$

$[\![\Diamond Px]\!]^u_u(W) = W.$

$[\![\Diamond Px]\!]^s_u(W) = [\![\Diamond Px]\!]^u_s(W) = \emptyset.$

The first litmus test of the system is whether it can deal with the contrast between *'He may be present ... He isn't present.'* and *'He isn't present ...*He may be present.'*, no matter what the referent of 'he' happens to be. We illustrate with the example interpretation that the semantics of DMPL make the translation of the first acceptable, but that of the second unacceptable:

$$\begin{aligned}
[\![\Diamond Px;\ \neg Px]\!]^s_s(W) &= [\![\neg Px]\!]^s_s([\![\Diamond Px]\!]^s_s(W)) \\
&= [\![\neg Px]\!]^s_s(W) \\
&= \{w_3\}. \\
[\![\Diamond Px;\ \neg Px]\!]^u_u(W) &= [\![\neg Px]\!]^u_u([\![\Diamond Px]\!]^u_u(W)) \\
&= [\![\neg Px]\!]^u_u(W) \\
&= \{w_2, w_3\}.
\end{aligned}$$

$$
\begin{aligned}
[\![\neg Px;\ \Diamond Px]\!]^s_s(W) &= [\![\Diamond Px]\!]^s_s([\![\neg Px]\!]^s_s(W)) \\
&= [\![\Diamond Px]\!]^s_s(\{w_3\}) \\
&= \emptyset. \\
[\![\neg Px;\ \Diamond Px]\!]^u_u(W) &= [\![\Diamond Px]\!]^u_u([\![\neg Px]\!]^u_u(W)) \\
&= [\![\Diamond Px]\!]^u_u(\{w_2, w_3\}) \\
&= \emptyset.
\end{aligned}
$$

Secondly, the *maybe* operator should not eliminate any of the current epistemically possible worlds. A litmus test for this is that a sentence like *He may be present, but he may just as well not be present* should come out true in the situation of complete ignorance, and leave us in a state of complete ignorance. And this is precisely what we find, regardless of how the reference for 'he' gets fixed:

$$
\begin{aligned}
[\![\Diamond Px;\ \Diamond\neg Px]\!]^s_s(W) &= [\![\Diamond\neg Px]\!]^s_s([\![\Diamond Px]\!]^s_s(W)) \\
&= [\![\Diamond\neg Px]\!]^s_s(W) \\
&= W. \\
[\![\Diamond Px;\ \Diamond\neg Px]\!]^u_u(W) &= [\![\Diamond\neg Px]\!]^u_u([\![\Diamond Px]\!]^u_u(W)) \\
&= [\![\Diamond\neg Px]\!]^u_u(W) \\
&= W.
\end{aligned}
$$

We want to argue that if *may* in the following examples is taken in its epistemic sense, then (12) should turn out to be acceptable, but (13) should not.

(12) Everyone may have escaped from the fire ...
Oh dear! Someone hasn't made it.

(13) Someone hasn't escaped from the fire.
*Everyone may have escaped.

One can easily imagine a fire fighter uttering discourse (12) as a comment to a colleague during an inspection round. But (13) would sound very weird in those same circumstances.

It is clear, however, that *may* or *might* has more senses than the purely epistemic sense that we are focussing on here. Note, for instance, that the following discourse is intuitively acceptable.

(14) Someone hasn't escaped from the fire.
Everyone might have escaped.

The acceptability of (14) means that *might* in this example does not express a purely epistemic modality. The second sentence of the discourse does not express that a state of affairs where everyone has escaped is in accordance with one's epistemic state (it is not, witness the information contained in the first sentence). The second sentence expresses something quite different: things might have turned out different from how they in fact turned out. To account for this kind of example (with an *irrealis* flavour), we would have to introduce a new operator for alethic modality, which looks at all indices, irrespective of whether they are still 'in the game' as epistemic alternatives. This would boil down to evaluation in yet another dimension. We will not pursue this issue here.

7 Quantified Dynamic Modal Logic

To gain further insight in DMPL, our next goal is to provide an axiomatization. For this we take our cue from two existing axiomatisations for DPL and Update Logic, respectively. Van Eijck and De Vries (1992a) use Hoare logic to axiomatize DPL, and in Van Eijck and De Vries (1992b) they apply the same methods to Update Logic. The extension of DPL with the epistemic operator suggests the use of modal predicate logic as assertion language in a Hoare style calculus for DMPL programs.

Rather than confining ourselves to Hoare style *implications* we want to be able to use the full range of logical connections between static assertions from modal predicate logic and programs from DMPL. We will therefore define a version of quantified dynamic modal logic that gives us the expressive power we need. This logic is inspired by Pratt (1976). See also Goldblatt (1987) for a general survey of dynamic logics, Van Eijck (app) for a reformulation of the Hoare style calculus of Van Eijck and De Vries (1992a) in quantified dynamic logic, and Van Eijck and De Vries (1993) for a reformulation of the Hoare style calculus for Update Logic in S5 propositional dynamic logic.

QDML programs $\pi := Rt \cdots t \mid t = t \mid v :=? \mid \pi ; \pi \mid \neg\pi \mid \Diamond\pi.$

QDML formulas $\varphi ::= Rt \cdots t \mid t = t \mid \neg\varphi \mid \varphi \wedge \varphi \mid \exists v\varphi \mid \Diamond\varphi \mid \langle\pi\rangle\varphi.$

The programs of QDL are the DMPL programs, the formulas are the formulas of modal predicate logic, with an extra modality for DMPL programs added.

We use $\top$ as an abbreviation of $\forall x(x = x)$ and $\bot$ as an abbreviation of $\neg\top$. As is customary, we abbreviate $\neg(\neg\varphi \wedge \neg\psi)$ as $(\varphi \vee \psi)$. Also, we write $\neg(\varphi \wedge \neg\psi)$ as $(\varphi \rightarrow \psi)$, $\neg\Diamond\neg\varphi$ as $\Box\varphi$, $\langle\neg(\pi_1; \neg\pi_2)\rangle\varphi$ as

$\langle \pi_1 \Rightarrow \pi_2 \rangle\varphi$, $\neg\langle\pi\rangle\neg\varphi$ as $[\pi]\varphi$ and $\neg\exists x\neg\varphi$ as $\forall x\varphi$. Also, we omit outermost parentheses for readability.

We consider index sets I as pointers to universal Kripke models consisting of those worlds which are the current epistemic alternatives, with accessibility relation $I \times I$. Recall from the literature that the modal logic determined by the class of finite universal frames is S5. Moreover, for any universal model $\mathcal{M}$ (a universal frame with first order valuations in some domain M assigned to all of its worlds) there is a submodel $\mathcal{M}'$ where all worlds have different valuations, and such that $\mathcal{M}'$ validates the same formulas. $\mathcal{M}'$ 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.

We define the notion of truth in a model $\mathcal{M}$, with respect to an index set I, an index $i \in I$, and an assignment s for $\mathcal{M}$.

Definition 6 (Truth in $\mathcal{M}$ for index set I, index i, ass s)

1. $\mathcal{M}, I, i, s \models Rt_1 \cdots t_n$ *iff* $\mathcal{M}, i \models_s Rt_1 \cdots t_n$.
2. $\mathcal{M}, I, i, s \models t_1 = t_2$ *iff* $\mathcal{M}, i \models_s t_1 = t_2$.
3. $\mathcal{M}, I, i, s \models \neg\varphi$ *iff it is not the case that* $\mathcal{M}, I, i, s \models \varphi$.
4. $\mathcal{M}, I, i, s \models \varphi \wedge \psi$ *iff* $\mathcal{M}, I, i, s \models \varphi$ *and* $\mathcal{M}, I, i, s \models \psi$.
5. $\mathcal{M}, I, i, s \models \exists v\varphi$ *iff for some* $d \in M$, $\mathcal{M}, I, i, s(v|d) \models \varphi$.
6. $\mathcal{M}, I, i, s \models \Diamond\varphi$ *iff there is some* $j \in I$ *with* $\mathcal{M}, I, j, s \models \varphi$.
7. $\mathcal{M}, I, i, s \models \langle\pi\rangle\varphi$ *iff there is an assignment* u *for which* $i \in [\![\pi]\!]^s_u(I)$ *and* $\mathcal{M}, [\![\pi]\!]^s_u(I), i, u \models \varphi$.

More global notions of truth are now defined by universally quantifying over the various parameters, as usual:

Definition 7 (Truth for I and i, Truth for I, Truth, Validity)

- $\mathcal{M}, I, i \models \varphi$ *if for all assignments* s *for* $\mathcal{M}$: $\mathcal{M}, I, i, s \models \varphi$.
- $\mathcal{M}, I \models \varphi$ *if for all* $i \in I$: $\mathcal{M}, I, i \models \varphi$.
- $\mathcal{M} \models \varphi$ *if for all* $I \subseteq W$: $\mathcal{M}, I \models \varphi$.
- $\models \varphi$ *if for all* $\mathcal{M}$: $\mathcal{M} \models \varphi$.

The consequence relation for QDML is defined as follows:

Definition 8 (Consequence for QDML) $\Gamma \models \varphi$ *if for all triples* $\mathcal{M}, I, i$ *the following holds: if there is an assignment* s *for* $\mathcal{M}$ *with* $\mathcal{M}, I, i, s \models \gamma$ *for all* $\gamma \in \Gamma$, *then* $\mathcal{M}, I, i, s \models \varphi$.

It is convenient to define the following operation on index sets.

Definition 9 (Pruning of index set I by φ, given s)
$I^s_\varphi \stackrel{\text{def}}{=} \{i \in I \mid \mathcal{M}, I, i, s \models \varphi\}$.

Note that $\downarrow [\![\pi]\!]^s(I) = I^s_{\langle\pi\rangle\top}$.

We now define localisations of QDML formulas. If φ, ψ are formulas of QDML, then $\varphi{\downarrow}\psi$, the localisation of φ to ψ, is given by the following definition.

Definition 10 (Localised QDML formulas $\varphi{\downarrow}\psi$)

$$
\begin{array}{lcll}
Rt_1 \cdots t_n{\downarrow}\psi & = & Rt_1 \cdots t_n \wedge \psi & \\
t_1 = t_2{\downarrow}\psi & = & t_1 = t_2 \wedge \psi & \\
(\varphi_1 \wedge \varphi_2){\downarrow}\psi & = & (\varphi_1{\downarrow}\psi) \wedge (\varphi_2{\downarrow}\psi) & \\
(\neg\varphi){\downarrow}\psi & = & \psi \wedge \neg(\varphi{\downarrow}\psi) & \\
(\exists v\varphi){\downarrow}\psi & = & \exists w(\varphi[w/v]{\downarrow}\psi) & (w \text{ a new variable}) \\
(\Diamond\varphi){\downarrow}\psi & = & \psi \wedge \Diamond(\varphi{\downarrow}\psi) & \\
(\langle \boldsymbol{Rt_1 \cdots t_n}\rangle\varphi){\downarrow}\psi & = & (Rt_1 \cdots t_n \wedge \varphi){\downarrow}\psi & \\
(\langle \boldsymbol{t_1 = t_2}\rangle\varphi){\downarrow}\psi & = & (t_1 = t_2 \wedge \varphi){\downarrow}\psi & \\
(\langle \boldsymbol{\pi_1};\, \boldsymbol{\pi_2}\rangle\varphi){\downarrow}\psi & = & \langle \boldsymbol{\pi_1}\rangle\langle \boldsymbol{\pi_2}\rangle\varphi{\downarrow}\psi & \\
(\langle \neg\boldsymbol{\pi}\rangle\varphi){\downarrow}\psi & = & (\varphi{\downarrow}[\boldsymbol{\pi}]\bot){\downarrow}\psi & \\
(\langle \boldsymbol{v} := ?\rangle\varphi){\downarrow}\psi & = & (\exists v\varphi){\downarrow}\psi & \\
(\langle \Diamond\boldsymbol{\pi}\rangle\varphi){\downarrow}\psi & = & (\varphi \wedge \Diamond\langle\boldsymbol{\pi}\rangle\top){\downarrow}\psi. &
\end{array}
$$

The localisation operator will play an important role in the axioms to be presented in Section 8. The following lemma makes clear what localisation accomplishes.

Lemma 3 (Localisation) *For all $\mathcal{M} = \langle M, W\rangle$, all $I \subseteq W$, and all assignments s for $\mathcal{M}$: $I^s_{\varphi\downarrow\psi} = (I^s_\psi)^s_\varphi$.*

Proof: What we have to prove is that $\mathcal{M}, I, i, s \models \varphi{\downarrow}\psi$ iff $\mathcal{M}, I, i, s \Vdash \psi$ and $\mathcal{M}, I^s_\psi, i, s \vdash \varphi$. The proof uses induction on the structure of φ; we merely give some example clauses.

For atomic formulas $R\bar{t}$ we have: $\mathcal{M}, I, i, s \models R\bar{t}{\downarrow}\psi$ iff $\mathcal{M}, I, i, s \models R\bar{t} \wedge \psi$ iff both $\mathcal{M}, I, i, s \models \psi$ and $\mathcal{M}, I^s_\psi, i, s \models R\bar{t}$.

The case of existential quantification:

$$
\begin{array}{ll}
 & \mathcal{M}, I, i, s \models (\exists v\varphi){\downarrow}\psi \\
\text{iff} & \mathcal{M}, I, i, s \models \exists w(\varphi[w/v]{\downarrow}\psi), \text{ with } w \text{ new} \\
\text{iff} & \text{there is some } d \in M \text{ with } \mathcal{M}, I, i, s(w|d) \models \varphi[w/v]{\downarrow}\psi \\
\text{iff (ind hyp)} & \mathcal{M}, I, i, s(w|d) \models \psi \text{ and } \mathcal{M}, I^{s(w|d)}_\psi, i, s(w|d) \models \varphi[w/v] \\
\text{iff } (w \text{ fresh}) & \mathcal{M}, I, i, s \models \psi \text{ and } \mathcal{M}, I^s_\psi, i, s(w|d) \models \varphi[w/v] \\
\text{iff} & \mathcal{M}, I, i, s \models \psi \text{ and } \mathcal{M}, I^s_\psi, i, s \models \exists v\varphi.
\end{array}
$$

The modal operator case:

	$\mathcal{M}, I, i, s \models (\Diamond\varphi)\downarrow\psi$
iff	$\mathcal{M}, I, i, s \models \psi \wedge \Diamond(\varphi\downarrow\psi)$
iff	$\mathcal{M}, I, i, s \models \psi$ and there is a $j \in I$ such that $\mathcal{M}, I, j, s \models \varphi\downarrow\psi$
iff (ind hyp)	$\mathcal{M}, I, i, s \models \psi$ and there is a $j \in I$ such that $\mathcal{M}, I, j, s \models \psi$ and $\mathcal{M}, I^s_\psi, j, s \models \varphi$
iff	$\mathcal{M}, I, i, s \models \psi$ and $M, I^s_\psi, i, s \models \Diamond\varphi$.

The definition of localisations for formulas starting with a program modality decomposes this operator, so in the program modality case all we have to do is check that the decomposition agrees with the semantic clause for the program construct and apply the induction hypothesis. This completes the induction argument and the proof. ∎

8 A Calculus for QDML

The calculus for QDML to be presented in this section provides the explicit link between static meaning and dynamic meaning for DMPL. The calculus has five sets of axiom schemata: (i) propositional and quantificational schemata, (ii) S5 schemata for the epistemic modality $\Box$, (iii) K-schemata for the program modalities, (iv) atomic test schemata and an assignment schema for the atomic program modalities, and (v) program composition schemata.

Propositional and Quantificational Schemata We start by taking the axiom schemata of propositional logic and first order quantification:

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

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

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

A 4 $\forall v\varphi \rightarrow [t/v]\varphi$, *provided t is free for v in φ.*

A 5 $\varphi \rightarrow \forall v\varphi$, *provided v has no free occurrences in φ.*

A 6 $\forall v(\varphi \rightarrow \psi) \rightarrow (\forall v\varphi \rightarrow \forall v\psi)$.

A 7 $v = v$.

A 8 $v = w \rightarrow (\varphi \rightarrow \varphi')$, *where φ' results from replacing some v-occurrence(s) in φ by w.*

See e.g. Enderton (1972) for discussion and motivation.

The S5 Schemata for $\Box$

A 9 $\Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$.

A 10 $\Box\varphi \to \varphi$.

A 11 $\Box\varphi \to \Box\Box\varphi$.

A 12 $\Diamond\Box\varphi \to \varphi$.

These are the propositional S5 modalities. The next axiom gives the Barcan Schema, which takes care of the interaction of quantifiers and the S5 modal operator.

A 13 $\forall v \Diamond\varphi \to \Diamond\forall v\varphi$.

The K-schema for the program modalities

A 14 $[\boldsymbol{\pi}](\varphi \to \psi) \to ([\boldsymbol{\pi}]\varphi \to [\boldsymbol{\pi}]\psi)$.

Atomic Test Schemata

A 15 $\langle \boldsymbol{R t_1 \cdots t_n} \rangle\varphi \leftrightarrow \varphi{\downarrow}Rt_1 \cdots t_n$.

A 16 $\langle \boldsymbol{t_1 = t_2} \rangle\varphi \leftrightarrow \varphi{\downarrow}t_1 = t_2$.

Assignment Schema

A 17 $\langle \boldsymbol{v := ?} \rangle\varphi \leftrightarrow \exists v\varphi$.

Program Composition Schemata The schemata for complex programs.

A 18 $\langle \boldsymbol{\pi_1 ; \pi_2} \rangle\varphi \leftrightarrow \langle \boldsymbol{\pi_1} \rangle\langle \boldsymbol{\pi_2} \rangle\varphi$.

A 19 $\langle \boldsymbol{\neg\pi} \rangle\varphi \leftrightarrow \varphi{\downarrow}[\boldsymbol{\pi}]\bot$.

A 20 $\langle \boldsymbol{\Diamond\pi} \rangle\varphi \leftrightarrow \varphi \wedge \Diamond\langle \boldsymbol{\pi} \rangle\top$.

Rules of inference The rules of inference of the calculus are Universal Generalization (conclude from $\vdash \varphi$ to $\vdash \forall v\varphi$), Necessitation for $\Box$ (conclude from $\vdash \varphi$ to $\vdash \Box\varphi$) and Modus Ponens (conclude from $\vdash \varphi \rightarrow \psi$ and $\vdash \varphi$). It turns out that necessitation for program modality can be derived (Proposition 4). The notions of theoremhood ($\vdash \varphi$) and derivability from a set of premisses ($\Gamma \vdash \varphi$) in the calculus are defined in the standard way.

Proposition 4 (Necessitation for program modalities)
If $\vdash \varphi$, *then* $\vdash [\pi]\varphi$.

Proof: Induction on the complexity of π. ■

Theorem 5 (Soundness) *If* $\Gamma \vdash \varphi$ *then* $\Gamma \models \varphi$.

Proof: Checking of axioms and rules, in the usual manner. ■

Completeness of the calculus is also straightforward, as is to be expected from the fact that there are no difficult iterative phenomena (no Kleene star among the program operators).

Theorem 6 (Completeness) *If* $\Gamma \models \varphi$ *then* $\Gamma \vdash \varphi$.

Proof: First observe that the following translation function * from QDML to S5 modal predicate logic preserves truth for index set, index and assignment.

$$
\begin{array}{lcl}
(Rt_1 \cdots t_n)^* & = & Rt_1 \cdots t_n \\
(t_1 = t_2)^* & = & t_1 = t_2 \\
(\varphi \wedge \psi)^* & = & \varphi^* \wedge \psi^* \\
(\neg\varphi)^* & = & \neg\varphi^* \\
(\exists v\varphi)^* & = & \exists v\varphi^* \\
(\langle \boldsymbol{Rt_1} \cdots \boldsymbol{t_n} \rangle\varphi)^* & = & \varphi^* \downarrow Rt_1 \cdots t_n \\
(\langle \boldsymbol{t_1} = \boldsymbol{t_2} \rangle\varphi)^* & = & \varphi^* \downarrow t_1 = t_2 \\
(\langle \boldsymbol{\pi_1}; \boldsymbol{\pi_2} \rangle\varphi)^* & = & (\langle \boldsymbol{\pi_1} \rangle \langle \boldsymbol{\pi_2} \rangle\varphi)^* \\
(\langle \neg\boldsymbol{\pi} \rangle\varphi)^* & = & \varphi^* \downarrow ([\boldsymbol{\pi}]\bot)^* \\
(\langle \boldsymbol{v} :=? \rangle\varphi)^* & = & (\exists v\varphi)^* \\
(\langle \Diamond\boldsymbol{\pi} \rangle\varphi)^* & = & \varphi^* \wedge \Diamond(\langle \boldsymbol{\pi} \rangle\top)^*
\end{array}
$$

Thus, it follows from $\Gamma \models \varphi$ that $\Gamma \models \varphi^*$. Next, use the completeness of S5 modal predicate logic to conclude from $\Gamma \models \varphi^*$ that $\Gamma \vdash \varphi^*$. Finally, note that the translation steps and their inverses in the definition of * are licenced by the atomic test schemata and the program composition schemata of the calculus. This allows us to conclude from $\Gamma \vdash \varphi^*$ that $\Gamma \vdash \varphi$. ■

9 Calculating Meanings

The translation function * from Theorem 6 derives more or less directly from our calculus. We will now demonstrate that it can be used for calculating the meanings of DMPL programs as formulas of S5 modal predicate logic.

Please note that in this paper we are not concerned with giving a translation algorithm from natural language to our representation language, although it is clear that this can be done using standard techniques from Montague grammar; see, e.g., Bouchez, Van Eijck and Istace (1993) or Muskens (1991). All that we want to establish here is that DMPL is a reasonable representation language for those aspects of natural language meaning that involve epistemic modalities and anaphoric linking, by showing that the translation function * that is implied by our calculus can be used to derive truth conditions for natural language texts in S5 modal predicate logic.

The demonstration will proceed by analyzing the example natural language texts from Section 1, repeated here for convenience.

(15) A man walked out. Maybe he was angry.

A reasonable DMPL translation is the following (note that we ignore tense).

(16) $\eta x : \textit{man}\ x;\ \textit{walk-out}\ x;\ \Diamond \textit{angry}\ x.$

We want to find a specification of the conditions under which this DMPL program succeeds, for these are the truth conditions of the natural language example (in the intended reading, as specified by the DMPL translation). In other words, we want the truth conditions of the following QDML formula.

(17) $\langle \boldsymbol{\eta x} : \textit{man}\ \boldsymbol{x};\ \textit{walk-out}\ \boldsymbol{x};\ \Diamond \textit{angry}\ \boldsymbol{x}\rangle\top.$

We apply the translation function.

(18) $(\langle \boldsymbol{\eta x} : \textit{man}\ \boldsymbol{x};\ \textit{walk-out}\ \boldsymbol{x};\ \Diamond \textit{angry}\ \boldsymbol{x}\rangle\top)^*.$

What we get is:

$$
\begin{array}{ll}
(\langle \boldsymbol{\eta x} : \textit{man}\ \boldsymbol{x};\ \textit{walk-out}\ \boldsymbol{x};\ \Diamond \textit{angry}\ \boldsymbol{x}\rangle\top)^* & = \\
(\langle \boldsymbol{\eta x} : \textit{man}\ \boldsymbol{x}\rangle\langle \textit{walk-out}\ \boldsymbol{x}\rangle\langle \Diamond \textit{angry}\ \boldsymbol{x}\rangle\top)^* & = \\
\exists x(\langle \textit{man}\ \boldsymbol{x}\rangle\langle \textit{walk-out}\ \boldsymbol{x}\rangle\langle \Diamond \textit{angry}\ \boldsymbol{x}\rangle\top)^*) & = \\
\exists x((\langle \textit{walk-out}\ \boldsymbol{x}\rangle\langle \Diamond \textit{angry}\ \boldsymbol{x}\rangle\top)^* \downarrow \textit{man}\ x) & = \\
\exists x(\textit{man}\ x \wedge (\langle \textit{walk-out}\ \boldsymbol{x}\rangle\langle \Diamond \textit{angry}\ \boldsymbol{x}\rangle\top)^*) & = \\
\exists x(\textit{man}\ x \wedge (\langle \Diamond \textit{angry}\ \boldsymbol{x}\rangle\top)^* \downarrow \textit{angry}\ x) & = \\
\exists x(\textit{man}\ x \wedge \textit{walk-out}\ x \wedge (\langle \Diamond \textit{angry}\ \boldsymbol{x}\rangle\top)^*) & = \\
\exists x(\textit{man}\ x \wedge \textit{walk-out}\ x \wedge \Diamond(\langle \textit{angry}\ \boldsymbol{x}\rangle\top)^*) & = \\
\exists x(\textit{man}\ x \wedge \textit{walk-out}\ x \wedge \Diamond(\top \downarrow \textit{angry}\ x) & = \\
\exists x(\textit{man}\ x \wedge \textit{walk-out}\ x \wedge \Diamond \textit{angry}\ x). &
\end{array}
$$

Before we proceed to the next example, it is useful to list some derived translation instructions.

$$
\begin{array}{lcl}
([\boldsymbol{R t_1 \cdots t_n}]\varphi)^* & = & Rt_1 \cdots t_n \rightarrow (\varphi^* \downarrow Rt_1 \cdots t_n) \\
([\boldsymbol{t_1 = t_2}]\varphi)^* & = & t_1 = t_2 \rightarrow (\varphi^* \downarrow t_1 = t_2) \\
([\boldsymbol{\pi_1}; \boldsymbol{\pi_2}]\varphi)^* & = & ([\boldsymbol{\pi_1}][\boldsymbol{\pi_2}]\varphi)^* \\
([\neg\boldsymbol{\pi}]\varphi)^* & = & [\boldsymbol{\pi}]\bot \rightarrow \varphi^* \downarrow ([\boldsymbol{\pi}]\bot)^* \\
(\langle\boldsymbol{\pi_1} \Rightarrow \boldsymbol{\pi_2}\rangle\varphi)^* & = & \varphi^* \downarrow ([\boldsymbol{\pi_1}]\langle\boldsymbol{\pi_2}\rangle\top)^* \\
([\boldsymbol{\pi_1} \Rightarrow \boldsymbol{\pi_2}]\varphi)^* & = & ([\boldsymbol{\pi_1}]\langle\boldsymbol{\pi_2}\rangle\top)^* \rightarrow \varphi^* \downarrow ([\boldsymbol{\pi_1}]\langle\boldsymbol{\pi_2}\rangle\top)^* \\
([\eta\boldsymbol{v} : \boldsymbol{\pi}]\varphi)^* & = & \forall v([\boldsymbol{\pi}]\varphi)^* \\
([\Diamond\boldsymbol{\pi}]\varphi)^* & = & \Diamond(\langle\boldsymbol{\pi}\rangle\top)^* \rightarrow \varphi^*
\end{array}
$$

We can now tackle the second example of Section 1.

(19) If a man walks out, then maybe he is angry.

A reasonable DMPL translation:

(20) $(\eta x : \mathit{man}\ x;\ \mathit{walk\text{-}out}\ x) \Rightarrow \Diamond \mathit{angry}\ x.$

We have to calculate the truth conditions of the following QDML sentence.

(21) $\langle(\eta\boldsymbol{x} : \mathit{man}\ \boldsymbol{x};\ \mathit{walk\text{-}out}\ \boldsymbol{x}) \Rightarrow \Diamond \mathit{angry}\ \boldsymbol{x}\rangle\top.$

Again the * function allows us to translate this into S5 modal predicate logic.

$$
\begin{array}{ll}
(\langle(\eta\boldsymbol{x} : \mathit{man}\ \boldsymbol{x};\ \mathit{walk\text{-}out}\ \boldsymbol{x}) \Rightarrow \Diamond \mathit{angry}\ \boldsymbol{x}\rangle\top)^* & = \\
\top\downarrow([\eta\boldsymbol{x} : \mathit{man}\ \boldsymbol{x};\ \mathit{walk\text{-}out}\ \boldsymbol{x}]\langle\Diamond \mathit{angry}\ \boldsymbol{x}\rangle\top)^* & = \\
([\eta\boldsymbol{x} : \mathit{man}\ \boldsymbol{x};\ \mathit{walk\text{-}out}\ \boldsymbol{x}]\langle\Diamond \mathit{angry}\ \boldsymbol{x}\rangle\top)^* & = \\
([\eta\boldsymbol{x} : \mathit{man}\ \boldsymbol{x}][\mathit{walk\text{-}out}\ \boldsymbol{x}]\langle\Diamond \mathit{angry}\ \boldsymbol{x}\rangle\top)^* & = \\
\forall x([\mathit{man}\ \boldsymbol{x}][\mathit{walk\text{-}out}\ \boldsymbol{x}]\langle\Diamond \mathit{angry}\ \boldsymbol{x}\rangle\top)^* & = \\
\forall x(\mathit{man}\ x \rightarrow ([\mathit{walk\text{-}out}\ \boldsymbol{x}]\langle\Diamond \mathit{angry}\ \boldsymbol{x}\rangle\top)^*) & = \\
\forall x(\mathit{man}\ x \rightarrow (\mathit{walk\text{-}out}\ x \rightarrow (\langle\Diamond \mathit{angry}\ \boldsymbol{x}\rangle\top)^*)) & = \\
\forall x(\mathit{man}\ x \rightarrow (\mathit{walk\text{-}out}\ x \rightarrow \Diamond \mathit{angry}\ x)). &
\end{array}
$$

10 Towards an Analysis of Conversation

We have investigated the logic of a representation language for natural language meaning that can handle dynamic binding of variables and dynamically interpreted epistemic modalities. It would be a straightforward extension of the representation language to add generalized quantifiers with a mechanism for internal dynamic bindings, along the lines of Van Eijck and De Vries (1992a). Another useful extension would be a modification of the language to handle puzzles of modal subordination (Roberts (1989)).

One flaw of this proposal should be mentioned straight away. It inherits all the puzzles of reference and modality from modal predicate logic. As Groenendijk, Stokhof and Veltman remark in (1993), examples like (22) and (23) get handled in an implausible way.

(22) Someone$_1$ did it. Maybe it$_1$ is him$_2$. Therefore, it$_1$ is him$_2$.

(23) Someone$_1$ did it. It$_1$ could be anyone.

If 'Maybe it is him' in example (22) gets translation $\Diamond x = y$, then the DMPL rendering of this example becomes valid. Not very plausible. Similarly, if 'It could be anyone' gets DMPL rendering $\neg\eta x : \neg(\Diamond x = y)$, where y is the referent of 'it', then we are in trouble, for identities like $x = y$ are true at every index iff they are true at a single index, for a given assignment, as the index sets and the assignments vary independently. This makes example (23) inconsistent in every model with more than one individual in its domain.

We have to plead guilty to this charge. But then, we believe that this whole issue of reference and modality is orthogonal to our main topic of many-dimensional dynamic analysis. One way out would be to argue that the problem rests with the translations of 'Maybe it is him' and 'It could be anyone.' Isn't a rendering that translates 'it' as *the one who did it.*, i.e., as a description, more plausible? The denotation of the predicate P for 'having done it' varies per index, so the translation $\neg\eta x : \neg\Diamond(x = \iota y : Py)$ does not suffer from the identity crisis. Another possible way out would be to replace objects in the semantics by something more complex, such as growing families of functions through models, in the style of Ghilardi (1991), say.

We will leave this issue for future research and close off with an issue more central to many-dimensional dynamic analysis, namely the extension of DMPL with operators for role switching and agreement in information exchange conversations among a given set of participants A. Here is a suitable representation language for that.

programs $\pi := Rt \cdots t \mid t = t \mid v := ? \mid \pi;\pi \mid \neg\pi \mid \Diamond\pi$.

agreements $\alpha ::= \Lambda \mid a : ok; \alpha$.

conversations $C := a : \pi; \alpha \mid C; C$.

The programs of this language are precisely the DMPL programs, but now these programs can be ingredients of conversations, by being asserted by a participant and agreed to by other participants. The conversational item $a : \pi; b_1 : ok; \cdots; b_n : ok$ is supposed to mean that speaker a contributes π to the discourse, and participants $b_1, \ldots, b_n$ agree to accept this piece of information. Of course, appending explicit agreement statements to every contribution to the conversation is something of an idealization; in real-life conversations the participants often agree or disagree implicitly. Still, this idealization is very useful. Consider conversation example (5) from the introduction. One would like to know which readings of the example are consistent. The reading where B agrees with A's first statement but not with A's second statement, and A does not agree with anything B says is given by the following translation in our conversation representation medium:

(24) $a : \eta x : man\ x; walk\text{-}out\ x; b : ok;$
$b : angry\ x;$
$a : \neg angry\ x; \Diamond drunk\ x;$
$b : \neg drunk\ x.$

It is easily seen that this reading is consistent in the sense that there are possible initial epistemic states of the participants such that after updating with this conversation both participants end up with non-empty sets of indices representing their new epistemic states. For readings of the conversation where either A or B agrees with everything the other says, this is not the case, so such readings are much less plausible.

A suitable assertion language for analysing conversations, as defined above, can readily be had as an extension of QDML:

$$\varphi ::= Rt \cdots t \mid t = t \mid \neg\varphi \mid \varphi \wedge \varphi \mid \exists v \varphi \mid \langle a \rangle \varphi \mid \langle C \rangle \varphi.$$

The formulas of this language are the formulas of QDML, with the single modal operator $\Diamond$ replaced by a set of modal operators $\{\langle a \rangle \mid a \in A\}$, to be interpreted as the epistemic accessibility relations for each discourse participant a, and the modal operator $\langle \pi \rangle$ replaced by $\langle C \rangle$, for the result of updating with conversation C.

Rather than working out a full axiomatisation in this language, we confine ourselves to some formal details of the dynamic processing of conversations. Evaluation takes place, as before, with respect to models $\mathcal{M}$ consisting of a single domain and a set of indexed interpretations over this domain, but we replace index sets with functions from the set of discourse participants to index sets. Let K and K' range over such functions, i.e., $K, K' : A \rightarrow \mathcal{P}W$. The semantic interpretation clause for a single item of conversation becomes (we again suppress the model parameter $\mathcal{M}$):

$$[\![a : \pi; b_1 : ok; \cdots b_n : ok]\!]_u^s(K) = K',$$

where $K' = \lambda x.K'(x)$ is given by:

- if $x = a$ and $K(x) = [\![\pi]\!]_u^s(K(x))$, then $K'(x) = K(x)$,
- if $x = a$ and $K(x) \neq [\![\pi]\!]_u^s(K(x))$, then $K'(x) = \emptyset$,

- otherwise, if $x \in \{b_1, \ldots, b_n\}$, then $K'(x) = [\![\pi]\!]^s_u(K(x))$,
- otherwise $K'(x) = K(x)$.

This clause expresses that the contribution that a speaker a makes to a conversation should be in agreement with a's current epistemic state, i.e., it should be accepted in the set of indices $K(a)$. Also, the index sets for all discourse participants agreeing with a get updated with the contents of a's contribution, but interpreted from the perspective of that participant. I.e., if A says 'Maybe he was drunk', then A says this about a set D of individuals compatible with what has been going on in previous discourse, against the background of the epistemic state of A, but if B agrees with what A says, then B is talking about a set D' of individuals compatible with what has been going on in previous discourse, against the background of B's epistemic state.

Note, by the way, that someone with an inconsistent epistemic state can still make a meaningful contribution to a conversation, provided the contribution π by itself is not contradictory. For instance, A's utterance of 'Maybe he was drunk' may be incompatible with A's epistemic state, but nevertheless B could agree without ending up in an inconsistent epistemic state himself, simply because B's and A's initial epistemic states may be different.

Here is the semantic clause for composite conversations:

$$[\![C_1; C_2]\!]^s_u(K) = K',$$

where K' is given by:

$$K'(x) = \{i \in K(x) \mid i \in ([\![C_2]\!]^r_u([\![C_1]\!]^s_r(K)))(x) \text{ for some } r\}.$$

What this says is that for given initial and final variable assignments representing (possible) initial and final anaphoric linkings, the conversational update with two items consists of an update with the first item, leading to an intermediate assignment, followed with an update leading from the intermediate update result, given the intermediate assignment, to the final update result and final assignment. The independent role played in the clause by the assignments explains how it is possible that discourse participants may disagree about 'the same thing.'

This semantic analysis of what happens in conversation gives a dynamic account of the processes of role switching and agreement about cnversational statements among discourse participants. Of course, it is still very much a simplification of what is really going on in conversation. For instance, it does not take into account that participant A may also update her beliefs about B's beliefs about the world, or her beliefs about B's beliefs about C's beliefs about the world, and so on. But nevertheless it seems to us that it provides a convincing example of how a further dimension of score-keeping in a language game can get accounted for in a truly general dynamic framework.

Acknowledgements

This paper has benefited from helpful comments by Johan van Benthem, Paul Dekker, Tim Fernando, Makoto Kanazawa, Wilfried Meyer Viol, Kees Vermeulen, and Albert Visser, plus remarks from at most two others, acting as anonymous referees for this volume. Thank you all.

References

Barwise, J. 1987. Noun phrases, generalized quantifiers and anaphora. In P. Gärdenfors, ed., *Generalized Quantifiers: linguistic and logical approaches*, 1–30. Dordrecht: D. Reidel Publishing Company.

van Benthem, J. 1991. General dynamics. *Theoretical Linguistics* 17, 159–201.

van Benthem, J. 1991. *Language in Action: categories, lambdas and dynamic logic*. Studies in Logic 130. Amsterdam: Elsevier.

van Benthem, J., J. van Eijck, and A. Frolova. 1993. Changing preferences. Technical Report CS-R9310, CWI.

Bouchez, O., J. van Eijck, and O. Istace. 1993. A strategy for dynamic interpretation: a fragment and an implementation. In S. Krauwer, M. Moortgat, and Louis des Tombe, eds., *Sixth Conference of the European Chapter of the Association for Computational Linguistics — Proceedings of the Conference*, 61–70. ACL.

Dekker, P. 1993. *Transsentential Meditations*. PhD thesis, Department of Philosophy, University of Amsterdam.

van Eijck, J., and F.J. de Vries. 1992a. Dynamic interpretation and Hoare deduction. *Journal of Logic, Language, and Information* 1, 1–44.

van Eijck, J. and F.J. de Vries. 1992b. A sound and complete calculus for update logic. In P. Dekker and M. Stokhof, eds., *Proceedings of the Eighth Amsterdam Colloquium*, 133–152. ILLC, Amsterdam.

van Eijck, J. and F.J. de Vries. 1993. Reasoning about update logic. Technical Report CS-R9312, CWI, Amsterdam. To appear in the *Journal of Philosophical Logic.*

van Eijck, J. app. Axiomatizing dynamic predicate logic with quantified dynamic logic. In J. van Eijck and A. Visser, eds., *Logic and Information Flow.* MIT Press, Cambridge, Mass., to appear.

Enderton, H.B. 1972. *A Mathematical Introduction to Logic.* Academic Press.

Gärdenfors, P. 1988. *Knowledge in Flux: Modelling the Dynamics of Epistemic States.* MIT Press.

Ghilardi, S. 1991. Incompleteness results in Kripke semantics. *Journal of Symbolic Logic* 56, 517–538.

Goldblatt, R. 1987. *Logics of Time and Computation, Second Edition, Revised and Expanded*, volume 7 of *CSLI Lecture Notes.* Stanford: CSLI, 1992 (first edition 1987). Distributed by University of Chicago Press.

Groenendijk, J. and M. Stokhof. 1991a. Dynamic predicate logic. *Linguistics and Philosophy* 14, 39–100.

Groenendijk, J. and M. Stokhof. 1991b. Two theories of dynamic semantics. In J. van Eijck, ed., *Logics in AI—European Workshop JELIA '90*, 55–64. Lecture Notes in Artificial Intelligence. Berlin: Springer.

Groenendijk, J., M. Stokhof, and F. Veltman. 1993. Coreference and modality. Hand-out Workshop on Dynamic Semantics, ESLLI, August, 1993.

Lewis, D. 1979. Score keeping in a language game. *Journal of Philosophical Logic* 8, 339–359.

Muskens, R. 1991. Anaphora and the logic of change. In J. van Eijck, ed., *Logics in AI / European Workshop JELIA '90 / Amsterdam, The Netherlands, September 1990 / Proceedings*, Lecture Notes in Artificial Intelligence 478, 412–427. Springer Verlag.

Pratt, V. 1976. Semantical considerations on Floyd–Hoare logic. *Proceedings 17th IEEE Symposium on Foundations of Computer Science*, 109–121.

Roberts, C. 1989. Modal subordination and pronominal anaphora in discourse. *Linguistics and Philosophy* 12, 683–721.

Veltman, F. 1991. Defaults in update semantics. Technical report, Department of Philosophy, University of Amsterdam. To appear in the *Journal of Philosophical Logic.*

Vermeulen, C.F.M. 1993. Sequence semantics for dynamic predicate logic. *Journal of Logic, Language, and Information* 2, 217–254.