Presupposition Failure—A Comedy of Errors

Jan van Eijck
Centrum voor Wiskunde en Informatica (CWI), Amsterdam
and Onderzoeksinstituut voor Taal en Spraak (OTS), Utrecht

May 1994

Abstract

Presuppositions of utterances are the pieces of information you convey with an utterance no matter whether your utterance is true or not. We first study presupposition in a very simple framework of updating propositional information, with examples of how presuppositions of complex propositional updates can be calculated. Next we move on to presuppositions and quantification, in the context of a dynamic version of predicate logic, suitably modified to allow for presupposition failure. In both the propositional and the quantificational case, presupposition failure can be viewed as error abortion of procedures. Thus, a dynamic assertion logic which describes the preconditions for error abortion is the suitable tool for analysing presupposition.

1 Introduction

David Lewis in an influential paper [22] once compared the interpretation process that takes place during natural language understanding with the process of keeping a score during a game. The score board gets updated continuously with new answers to questions like 'Who is doing the talking?', 'What is the topic of conversation?' (or more concretely: what is the reference of *this* use of the demonstrative 'this'?), 'What are the common assumptions of speaker and listener?', etcetera.

The score board is used for keeping track of changes in the context. Adopting a dynamic perspective on language understanding, one can set up the semantics of natural language in such a way that one keeps track of these changes of context. Dynamic logic and assertion logic are eminently suited for this purpose, as I will try to illustrate in this paper for the notion of presupposition.

A presupposition of an utterance is a piece of information which I convey with my utterance no matter whether what I am saying is true or false. If I utter the sentence *John's wife was angry*, for example, and it turns out that my claim is false, then I have still conveyed the information to my listener that John is married. In many cases, the presupposition of an utterance will typically be a bit of information that the listerer knows already, but not invariably so. If I inform my customers that I want them to pay in cash for my services, I can do so by means of the polite message *We regret that credit cards are not accepted.* But my use of this message does not assume that my customers already know the presupposition of this statement, for the fact that credit cards are not accepted is precisely what I want to convey.

2 Dynamic Logic and Natural Language Analysis

The background metaphor of all uses of dynamic logic in natural language analysis and programming language analysis can be summed up in the following picture:

$$\boxed{\text{Old information state}} \rightarrow \text{update/action} \rightarrow \boxed{\text{new information state}}.$$

In proposals to employ this metaphor for the investigation of natural languages and programming languages, this picture has to be made more specific. The exploitation of the metaphor consists of making clear what a *state* is and what an *update* or *action* is.

In the prototypical case of use of dynamic logic in the analysis of imperative programming [25], states are taken to be states of memory of a machine, and actions are the manipulations of these memory states by running programs.

In the application to natural language, several different (and sometimes conflicting) ways of making the metaphor precise have been explored simultaneously. In a natural language context, it makes sense to view the states as states of mind of the discourse participants, or as the stock of knowledge of a hearer attending a lecture, or as a list of referents a speaker has in mind for the referring expressions she uses, etcetera.

In this paper I will locate a point where the explorations in natural language and computer science meet, by focussing on some applications of dynamic logic to cases where states are viewed from the natural language perspective as sets of propositional valuations or as lists of speaker references, and from the programming language perspective as lists of values for certain registers, and where the fact is exploited that these perspectives are very similar.

Analysing presupposition in natural language with the tools of dynamic logic is *prima facie* attractive because of the intuitive fact that presuppositions in natural language are intimately tied to sequential processing aspects, and dynamic logic was developed for the analysis of sequential processes. Moreover, dynamic logic has proved its mettle already in the semantics of natural language. A general appraisal of the dynamic perspective on natural language analysis can be found in Van Benthem [5, 6]. Dynamic logic is used for analysing anaphoric binding phenomena by Barwise [1], Groenendijk and Stokhof [16], Van Eijck and De Vries [12]. Epistemic aspects of natural language are analysed dynamically in Veltman [31] and Van Eijck and De Vries [13]. Finally, the dynamics of the interplay between anaphora and epistemics is the theme of Van Eijck and Cepparello [11].

A theory of presupposition in natural language has to give a general account of the reasons why *John loves his wife* presupposes that John has a wife, whereas *If John is married, then he loves his wife* has no such presupposition. The account that I give in this paper will capitalize on the fact that a similar phenomenon is observed in imperative programming, where (1) aborts with error in states where x has a value $\geq$ the value of `MaxInt`, while (2) never aborts.

```
x := x + 1                          (1)
IF x < MaxInt THEN x := x + 1       (2)
```

My analysis of presupposition can be viewed as a rational reconstruction of theories in the tradition of Karttunen [18], Karttunen and Peters [20, 24] and Heim [17]. It is in the same spirit as Beaver [2, 3], Krahmer [21], and Zeevat [33], to name but a few. My aim is not to cover as many linguistic data pertaining to presupposition as possible, but rather to isolate the essence of the phenomenon of semantic presupposition and analyse it with the toolbox of dynamic logic, in order to (i) get a clear sense of the role of the dynamics of context change in the account of the phenomenon and (ii) to explore and exploit the similarity to the case of presupposition failure (i.e., error abortion) in programming.

My approach to the problem will be one of 'piecemeal engineering'. I first concentrate on simpler problems for which I build tools, and use the insights thus gained for the construction of more complex tools for more complex problems. I will first present a system of assertion statements for a logic of updating propositional information (update logic) and a system of assertion statements for partial update logic. I will show that the latter system can be used for the analysis of presuppositions at the propositional level. Next, I turn to the interaction of presupposition and quantification, and I present a system of quantified dynamic error logic. In the final part of the paper, the use of this analysis tool for analysing the interplay between presupposition and quantification will be demonstrated.

3 Assertions for Update Logic

The notion of presupposition is intimately connected with that of information updating, which is a very simple form of context change. Let us therefore first look at information updating in a very simple system, basic Update Logic, due to Veltman [31]. In this logic, current information is represented as the set of propositional valuations ('worlds') that agree with what I know or believe. In the following language description, p ranges over a set P of atomic propositions.

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

The semantics for Update Logic is given in terms of updates of context sets, where a context set I is a subset of the set W of all propositional valuations for P. We write the result of updating context set I with the information π as $I[\pi]$.

$$\begin{array}{lcl} I[p] & = & \{w \in I \mid w \models p\} \\ I[\pi_1;\pi_2] & = & I[\pi_1][\pi_2] \\ I[\pi_1 \cup \pi_2] & = & I[\pi_1] \cup I[\pi_2] \\ I[\neg\pi] & = & I - I[\pi] \\ I[M\pi] & = & \begin{cases} I & \text{if } I[\pi] \neq \emptyset \\ \emptyset & \text{otherwise.} \end{cases} \end{array}$$

The state change effected by π in state I can be pictured as $I \xrightarrow{\pi} I[\pi]$. A simple induction argument shows that for all I and all π: $I[\pi] \subseteq I$ (eliminativity of Update Logic). Also, the transition system for information updating is deterministic: for a given I, π there is always precisely one J with $I \xrightarrow{\pi} J$, to wit $J = I[\pi]$. The determinism is built into the notation.

What makes Update Logic interesting is the 'consistency' operator M (for: 'maybe'). The following examples illustrate that M causes non-commutativity of sequential updating:

$$\{pq, p\bar{q}, \bar{p}q, \bar{p}\bar{q}\} \xrightarrow{\neg p} \{\bar{p}q, \bar{p}\bar{q}\} \xrightarrow{Mp} \emptyset.$$

$$\{pq, p\bar{q}, \bar{p}q, \bar{p}\bar{q}\} \xrightarrow{Mp} \{pq, p\bar{q}, \bar{p}q, \bar{p}\bar{q}\} \xrightarrow{\neg p} \{\bar{p}q, \bar{p}\bar{q}\}.$$

It is quite easy to build an assertion logic for Update Logic, linking the dynamic perspective on information updating to a static perspective (see Van Benthem [4, 5] and Van Eijck and De Vries [13]).

AUL $\varphi ::= p \mid (\varphi \wedge \varphi) \mid \neg\varphi \mid \Diamond\varphi \mid NS(\varphi, \pi).$

Here p ranges over the set P of propositional variables for UL, and π over the set of UL procedures. We assume the usual abbreviations: $\varphi \vee \psi := \neg(\neg\varphi \wedge \neg\psi)$,

$\varphi \rightarrow \psi := \neg(\varphi \wedge \neg\psi)$, $\varphi \leftrightarrow \psi := (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$, $\perp := p \wedge \neg p$, $\top := \neg\perp$, $\Box\varphi := \neg\Diamond\neg\varphi$.

The *NS* assertion ties the knot between modal propositional logic and Update Logic. We define the semantics of AUL as a function $||\varphi||_I$ that gives the interpretation of φ in state I (the reference set I is necessary to make sense of the modality).

$$
\begin{array}{lcl}
||p||_I & = & \{w \in I \mid w \models p\} \\
||\neg\varphi||_I & = & I - ||\varphi||_I \\
||\varphi \wedge \psi||_I & = & ||\varphi||_I \cap ||\psi||_I \\
||\Diamond\varphi||_I & = & \begin{cases} I & \text{if } ||\varphi||_I \neq \emptyset \\ \emptyset & \text{otherwise,} \end{cases} \\
||NS(\varphi, \pi)||_I & = & ||\varphi||_I[\pi].
\end{array}
$$

In particular, it follows from this definition that $||NS(\top, \pi)||_I = I[\pi]$. *NS* is short for 'next state': $NS(\varphi, \pi)$ is the formula characterizing the next state that one gets by updating a φ state with information π.

Here are the axioms schemes linking UL and AUL:

A 3.1 $NS(\varphi, p) \leftrightarrow (\varphi \wedge p)$

A 3.2 $NS(\varphi, \pi_1; \pi_2) \leftrightarrow (NS(NS(\varphi, \pi_1), \pi_2)$.

A 3.3 $NS(\varphi, \pi_1 \cup \pi_2) \leftrightarrow (NS(\varphi, \pi_1) \vee NS(\varphi, \pi_2))$.

A 3.4 $NS(\varphi, \neg\pi) \leftrightarrow (\varphi \wedge \neg NS(\varphi, \pi))$.

A 3.5 $NS(\varphi, \mathcal{M}\pi) \leftrightarrow (\varphi \wedge \Diamond NS(\varphi, \pi))$.

It is left to the reader to check that these axiom schemes are sound, in the sense that for every scheme $\varphi \leftrightarrow \psi$ and for every information state I it holds that $||\varphi||_I = ||\psi||_I$.

Here are two examples of applications of these schemes:

$$
\begin{array}{lll}
NS(\top, Mp; \neg p) & & \\
& \leftrightarrow & NS(NS(\top, Mp), \neg p) \\
& \leftrightarrow & NS(\Diamond NS(\top, p), \neg p) \\
& \leftrightarrow & NS(\Diamond p, \neg p) \\
& \leftrightarrow & \Diamond p \wedge \neg(NS(\Diamond p, p) \\
& \leftrightarrow & \Diamond p \wedge \neg(\Diamond p \wedge p) \\
& \leftrightarrow & \Diamond p \wedge \neg p.
\end{array}
$$

For given I, the clauses for $|| \; ||$ now give:

$$||\Diamond p \wedge \neg p||_I = \begin{cases} I - \{w \in I \mid w \models p\} & \text{if } \{w \in I \mid w \models p\} \neq \emptyset \\ \emptyset & \text{otherwise.} \end{cases}$$

$$\begin{array}{ll} NS(\top, \neg p; Mp) & \\ & \leftrightarrow \quad NS(NS(\top, \neg p), Mp) \\ & \leftrightarrow \quad NS(\neg NS(\top, p), Mp) \\ & \leftrightarrow \quad NS(\neg p, Mp) \\ & \leftrightarrow \quad \neg p \wedge \Diamond NS(\neg p, p) \\ & \leftrightarrow \quad \neg p \wedge \Diamond(\neg p \wedge p) \\ & \leftrightarrow \quad \neg p \wedge \Diamond \bot \\ & \leftrightarrow \quad \neg p \wedge \bot \\ & \leftrightarrow \quad \bot. \end{array}$$

For every I, $||\bot||_I = \emptyset$.

Note that in this analysis, M gets related to the $S5$ modality $\Diamond$ ($S5$ is the modality with the universal accessibility relation).

4 Presuppositions for Partial Update Logic

Now we move to the analysis of presuppositions in Update Logic. Assume that the presupposition of an atomic proposition p is given as the set of partial valuations where p is either true or false. A partial valuation for p divides the set of (partial) possible worlds in three:

- the worlds in W where p is true (notation: $\{w \in W \mid w \models p\}$),
- the worlds in W where p is false (notation: $\{w \in W \mid w \mathrel{=\!\!|} p\}$),
- the worlds in W where p is neither true nor false, i.e., the worlds where the presupposition of p doesn't hold (notation: $\{w \in W \mid w \not\models p \text{ and } w \not\mathrel{=\!\!|} p\}$).

We will assume that $w \models p$ and $w \mathrel{=\!\!|} p$ are mutually exclusive.

A theory about change of presuppositional context is nothing but a calculus of so-called presupposition projection, answering the following question. How can the presupposition of a complex update be given in terms of semantics of its components?

Here is the semantics for Partial Update Logic, the version of Update Logic that monitors changes in presuppositional context.

$$
\begin{array}{lcl}
I[p]^{+} & = & \{w \in I \mid w \models p\} \\
I[p]^{-} & = & \{w \in I \mid w =\!\mid p\} \\
I[\pi_1; \pi_2]^{+} & = & I[\pi_1]^{+}[\pi_2]^{+} \\
I[\pi_1; \pi_2]^{-} & = & I[\pi_1]^{-} \cup I[\pi_1]^{+}[\pi_2]^{-} \\
I[\pi_1 \cup \pi_2]^{+} & = & I[\pi_1]^{+} \cup I[\pi_2]^{+} \\
I[\pi_1 \cup \pi_2]^{-} & = & I[\pi_1]^{-} \cap I[\pi_2]^{-} \\
I[\neg\pi]^{+} & = & I[\pi]^{-} \\
I[\neg\pi]^{-} & = & I[\pi]^{+} \\
I[M\pi]^{+} & = & \begin{cases} I[\pi]^{+} \cup I[\pi]^{-} & \text{if } I[\pi]^{+} \neq \emptyset \\ \emptyset & \text{otherwise} \end{cases} \\
I[M\pi]^{-} & = & \begin{cases} I[\pi]^{-} & \text{if } I[\pi]^{+} = \emptyset \\ \emptyset & \text{otherwise.} \end{cases}
\end{array}
$$

I do not present the definitions of the semantics of 'maybe' as the last word on how epistemic possibility interacts with presupposition, but just as an example of how such interaction may be conceived. The idea behind the clauses is that *Maybe Jan is a bachelor* seems to imply that Jan is the kind of entity that qualifies for bachelorhood, to wit an adult man. Also, the denial of *Maybe Jan is a bachelor* seems to imply the same.

A similar *caveat* applies to the definition of negation. Another possible definition of negation is the following clause for strong or exclusive negation (assume we extend the language definition with a construct $\tilde{\pi}$):

$$
\begin{array}{lcl}
I[\tilde{\pi}]^{+} & = & I - I[\pi]^{+} \\
I[\tilde{\pi}]^{-} & = & I[\pi]^{+}
\end{array}
$$

This is the kind of negation that calcels presuppositions. There is a vast literature on the interaction of presupposition and negation. See, e.g., Van der Sandt [27] and Seuren [28].

For a better idea of what these clauses have to do with the intuitive notion of presupposition, we consider a few simple examples. Suppose we want to know what happens to the presupposition of *Jan is a bachelor*, when this proposition occurs embedded in more complex natural language contexts, such as conjunctions or implications of propositions, like *Jan is not an adult, or he is a bachelor.*

Assume *Jan is a bachelor* is an atomic proposition, say b, of which the assertion $ass(b)$ and the presupposition $pres(b)$ are given. Of course, the assertion should imply the presupposition: you can only assert that Jan is a bachelor if you assume that the presupposition that Jan is male and unmarried does also hold.

The assertion *Jan is a bachelor* consists of three parts: (i) Jan is male, (ii) Jan is adult, and (iii) Jan is not married. Formally: $ass(b) = p \wedge q \wedge \neg r$. The presupposition consists of only the parts (i) and (ii). So: $pres(b) = p \wedge q$. It immediately follows that $ass(b) \rightarrow pres(b)$ does hold. So far, so good.

Now we can picture the update of the state of complete ignorance concerning Jan's bachelorhood with the assertion of b as follows:

$$\{pqr, p\bar{q}r, pq\bar{r}, p\bar{q}\bar{r}, \bar{p}qr, \bar{p}\bar{q}r, \bar{p}q\bar{r}, \bar{p}\bar{q}\bar{r}\} \xrightarrow{b+} \{pq\bar{r}\}.$$

Updating the state of complete ignorance with the denial of b:

$$\{pqr, p\bar{q}r, pq\bar{r}, p\bar{q}\bar{r}, \bar{p}qr, \bar{p}\bar{q}r, \bar{p}q\bar{r}, \bar{p}\bar{q}\bar{r}\} \xrightarrow{b-} \{pqr\}.$$

And finally, updating the state of complete ignorance with the presupposition of b:

$$\{pqr, p\bar{q}r, pq\bar{r}, p\bar{q}\bar{r}, \bar{p}qr, \bar{p}\bar{q}r, \bar{p}q\bar{r}, \bar{p}\bar{q}\bar{r}\} \xrightarrow{b\pm} \{pqr, pq\bar{r}\}.$$

This picture squares with our intuition. Updating with the presupposition of b leads to the set of worlds where Jan is male and adult.

Now suppose we also have a proposition to express adulthood of Jan, say a. Assume that this proposition does not carry a presupposition of its own, so we have: $ass(a) = p$, $pres(a) = \top$ (this expresses that the presupposition of a is always fulfilled. We look at the presupposition of $a; b$, i.e., of the complex proposition *Jan is adult; he is a bachelor.* First, here is the the effect of the assertion of this proposition:

$$\{pqr, p\bar{q}r, pq\bar{r}, p\bar{q}\bar{r}, \bar{p}qr, \bar{p}\bar{q}r, \bar{p}q\bar{r}, \bar{p}\bar{q}\bar{r}, \} \xrightarrow{(a;b)+} \{pq\bar{r}\}.$$

Here is the effect of its denial:

$$\{pqr, p\bar{q}r, pq\bar{r}, p\bar{q}\bar{r}, \bar{p}qr, \bar{p}\bar{q}r, \bar{p}q\bar{r}, \bar{p}\bar{q}\bar{r}\} \xrightarrow{(a;b)-} \{pqr, \bar{p}qr, \bar{p}\bar{q}r, \bar{p}q\bar{r}, \bar{p}\bar{q}\bar{r}\}.$$

The first element of the result gives the effect of $a-$, the rest that of $a+; b-$ (compare the definition of the negative update with $\pi_1; \pi_2$ above). Combining these we get:

$$\{pqr, p\bar{q}r, pq\bar{r}, p\bar{q}\bar{r}, \bar{p}qr, \bar{p}\bar{q}r, \bar{p}q\bar{r}, \bar{p}\bar{q}\bar{r}\} \xrightarrow{(a;b)\pm} \{pqr, pq\bar{r}, \bar{p}qr, \bar{p}\bar{q}r, \bar{p}q\bar{r}, \bar{p}\bar{q}\bar{r}\}.$$

In the next section I present a calculus for expressing the presuppositions of complex update procedures in terms of assertions and presuppositions of their components.

5 Reasoning about Presuppositions

To reason about presuppositions of complex update procedures, we again introduce an assertion language. We assume that assertions and denials of atomic propositions are given, so that for every $p \in P$ we can use $ass(p)$ to express the assertion, and $ass(p) \vee den(p)$ to express the presupposition. Next, we introduce relations $ASS(\varphi, \pi)$ and $DEN(\varphi, \pi)$ which are intended to characterize the sets of partial worlds that are the result of the $[\pi]^+$ update and the $[\pi]^-$ update in a φ state. The presupposition of π, given φ, is then expressed as $ASS(\varphi, \pi) \vee DEN(\varphi, \pi)$.

APUL $\varphi ::= ass(p) \mid den(p) \mid (\varphi \wedge \varphi) \mid \neg\varphi \mid \Diamond\varphi \mid ASS(\varphi, \pi) \mid DEN(\varphi, \pi)$.

It is useful to have abbreviations for presuppositions of atomic and complex propositions, so we will write $pres(p)$ for $ass(p) \vee den(p)$, and $PRES(\varphi, \pi)$ for $ASS(\varphi, \pi) \vee DEN(\varphi, \pi)$. Here are the semantic clauses for this assertion language:

$$
\begin{array}{lcl}
||ass(p)||_I & = & \{w \in I \mid w \models p\} \\
||den(p)||_I & = & \{w \in I \mid w \mathrel{=\!\!|} p\} \\
||\neg\varphi||_I & = & I - ||\varphi||_I \\
||\varphi \wedge \psi||_I & = & ||\varphi||_I \cap ||\psi||_I \\
||\Diamond\varphi||_I & = & \begin{cases} I & \text{if } ||\varphi||_I \neq \emptyset \\ \emptyset & \text{otherwise,} \end{cases} \\
||ASS(\varphi, \pi)||_I & = & ||\varphi||_I[\pi]^+ \\
||DEN(\varphi, \pi)||_I & = & ||\varphi||_I[\pi]^-.
\end{array}
$$

If Γ is a set of formulas from APUL, we use $\Gamma \models \varphi$ ('φ follows logically from Γ') for $G \subseteq ||\varphi||_W$, where $G = \cap\{||\gamma||_W \mid \gamma \in \Gamma\}$, and $\models \varphi$ for $\emptyset \models \varphi$, i.e., $\models \varphi$ iff $||\varphi||_W = W$.

The following simple lemma expresses that PUL is eliminative:

Lemma 1 (Eliminativity) *For all φ, π:*

1. $\models ASS(\varphi, \pi) \rightarrow \varphi$.
2. $\models DEN(\varphi, \pi) \rightarrow \varphi$.

The link between PUL and the assertion language over it is given by the following axiom schemes:

A 5.1 $ASS(\varphi, p) \leftrightarrow (\varphi \wedge ass(p))$

A 5.2 $DEN(\varphi, p) \leftrightarrow (\varphi \wedge den(p))$

A 5.3 $ASS(\varphi, \pi_1; \pi_2) \leftrightarrow (ASS(ASS(\varphi, \pi_1), \pi_2)$.

A 5.4 $DEN(\varphi, \pi_1; \pi_2) \leftrightarrow (DEN(\varphi, \pi_1) \vee DEN(ASS(\varphi, \pi_1), \pi_2)$.

A 5.5 $ASS(\varphi, \pi_1 \cup \pi_2) \leftrightarrow (ASS(\varphi, \pi_1) \vee ASS(\varphi, \pi_2))$.

A 5.6 $DEN(\varphi, \pi_1 \cup \pi_2) \leftrightarrow (DEN(\varphi, \pi_1) \wedge DEN(\varphi, \pi_2))$.

A 5.7 $ASS(\varphi, \neg\pi) \leftrightarrow DEN(\varphi, \pi)$.

A 5.8 $DEN(\varphi, \neg\pi) \leftrightarrow ASS(\varphi, \pi)$.

A 5.9 $ASS(\varphi, \tilde{\pi}) \leftrightarrow (\varphi \wedge \neg ASS(\varphi, \pi))$.

A 5.10 $DEN(\varphi, \tilde{\pi}) \leftrightarrow ASS(\varphi, \pi)$.

A 5.11 $ASS(\varphi, M\pi) \leftrightarrow (PRES(\varphi, \pi) \wedge \Diamond ASS(\varphi, \pi))$.

A 5.12 $DEN(\varphi, M\pi) \leftrightarrow (DEN(\varphi, \pi) \wedge \neg\Diamond ASS(\varphi, \pi))$.

We write $\Gamma \vdash \varphi$ if φ follows from the axiom schemes of the propositional modal logic $S5$ plus the above axiom schemes, plus the formulas from Γ, with Necessitation and Modus Ponens (for more information about $S5$, see e.g. Goldblatt [15]), and we use $\vdash \varphi$ as shorthand for $\emptyset \vdash \varphi$.

Then to check that the above axiom schemes are sound, we have to establish that for every scheme $\varphi \leftrightarrow \psi$ it holds that $\models \varphi \leftrightarrow \psi$, i.e., that $||\varphi||_W = ||\psi||_W$. We leave this to the reader. The soundness of the calculus then follows from this fact, plus the soundness of the calculus for propositional $S5$.

Theorem 2 (Soundness) *For all* Γ, φ*:* $\Gamma \vdash \varphi$ *implies* $\Gamma \models \varphi$*.*

Conversely, the completeness of the calculus follows from the fact that the following translation from APUL to $S5$ preserves logical consequence:

$$
\begin{array}{lcl}
(ass(p))^\bullet & = & ass(p) \\
(den(p))^\bullet & = & den(p) \\
(\varphi \wedge \psi)^\bullet & = & \varphi^\bullet \wedge \psi^\bullet \\
(\neg\varphi)^\bullet & = & \neg\varphi^\bullet \\
(\Diamond\varphi)^\bullet & = & \Diamond\varphi^\bullet \\
(ASS(\varphi, p))^\bullet & = & \varphi^\bullet \wedge ass(p) \\
(DEN(\varphi, p))^\bullet & = & \varphi^\bullet \wedge den(p) \\
(ASS(\varphi, \pi_1; \pi_2))^\bullet & = & (ASS(ASS(\varphi, \pi_1), \pi_2))^\bullet \\
(DEN(\varphi, \pi_1; \pi_2))^\bullet & = & (DEN(\varphi, \pi_1) \vee DEN(ASS(\varphi, \pi_1), \pi_2))^\bullet \\
(ASS(\varphi, \pi_1 \cup \pi_2))^\bullet & = & (ASS(\varphi, \pi_1) \vee ASS(\varphi, \pi_2))^\bullet \\
(DEN(\varphi, \pi_1 \cup \pi_2))^\bullet & = & (DEN(\varphi, \pi_1) \wedge DEN(\varphi, \pi_2))^\bullet \\
(ASS(\varphi, \neg\!\!\neg\pi))^\bullet & = & (DEN(\varphi, \pi))^\bullet \\
(DEN(\varphi, \neg\!\!\neg\pi))^\bullet & = & (ASS(\varphi, \pi))^\bullet \\
(ASS(\varphi, \tilde{\pi}))^\bullet & = & (\varphi \wedge \neg ASS(\varphi, \pi))^\bullet \\
(DEN(\varphi, \tilde{\pi}))^\bullet & = & (ASS(\varphi, \pi))^\bullet \\
(ASS(\varphi, M\pi))^\bullet & = & (PRES(\varphi, \pi) \wedge \Diamond ASS(\varphi, \pi))^\bullet \\
(DEN(\varphi, M\pi))^\bullet & = & (DEN(\varphi, \pi) \wedge \neg\Diamond ASS(\varphi, \pi))^\bullet.
\end{array}
$$

Write $\Gamma^\bullet$ for $\{\gamma^\bullet \mid \gamma \in \Gamma\}$. Then, from the above, $\Gamma \models \varphi$ implies $\Gamma^\bullet \models \varphi^\bullet$. From this and the completeness of $S5$, $\Gamma^\bullet \vdash \varphi^\bullet$. From this and the fact that schemes A–5.1 through A–5.12 can be used to make the $\bullet$ translation steps and their converses, $\Gamma \vdash \varphi$. Thus we have established:

Theorem 3 (Completeness of APUL) $\Gamma \models \varphi$ *implies* $\Gamma \vdash \varphi$.

This calculus can be used as a calculus of presupposition projection for PUL. For example, the presupposition of $\pi_1; \pi_2$, in a state satisfying φ, is given by:

$$
\begin{array}{l}
PRES(\varphi, \pi_1; \pi_2) \\
\leftrightarrow (ASS(\varphi, \pi_1; \pi_2) \vee DEN(\varphi, \pi_1; \pi_2)) \\
\leftrightarrow (ASS(ASS(\varphi, \pi_1), \pi_2) \vee DEN(\varphi, \pi_1) \vee DEN(ASS(\varphi, \pi_1), \pi_2)) \\
\leftrightarrow (DEN(\varphi, \pi_1) \vee PRES(ASS(\varphi, \pi_1), \pi_2)) \\
\leftrightarrow (DEN(\varphi, \pi_1) \vee (PRES(ASS(\varphi, \pi_1), \pi_2)) \\
\leftrightarrow (DEN(\varphi, \pi_1) \vee (ASS(\varphi, \pi_1) \wedge PRES(ASS(\varphi, \pi_1), \pi_2))) \\
\leftrightarrow ((DEN(\varphi, \pi_1) \vee ASS(\varphi, \pi_1)) \wedge \\
\qquad (DEN(\varphi, \pi_1) \vee PRES(ASS(\varphi, \pi_1), \pi_2))) \\
\leftrightarrow (PRES(\varphi, \pi_1) \wedge (ASS(\varphi, \pi_1) \rightarrow PRES(ASS(\varphi, \pi_1), \pi_2))).
\end{array}
$$

Compare this with the example at the end of the previous section.

Similarly, we can express the presupposition of constructions like $\neg\!\!\neg\pi$, $\tilde{\pi}$, $M\pi$, $\pi_1 \cup \pi_2$.

$$\begin{aligned}
&PRES(\varphi, \neg\pi) \\
&\leftrightarrow (ASS(\varphi, \neg\pi) \vee DEN(\varphi, \neg\pi)) \\
&\leftrightarrow (ASS(\varphi, \pi) \vee DEN(\varphi, \pi)) \\
&\leftrightarrow PRES(\varphi, \pi).
\end{aligned}$$

$$\begin{aligned}
&PRES(\varphi, \tilde{\pi}) \\
&\leftrightarrow (ASS(\varphi, \tilde{\pi}) \vee DEN(\varphi, \tilde{\pi})) \\
&\leftrightarrow ((\varphi \wedge \neg ASS(\varphi, \pi)) \vee ASS(\varphi, \pi)) \\
&\leftrightarrow \varphi.
\end{aligned}$$

$$\begin{aligned}
&PRES(\varphi, M\pi) \\
&\leftrightarrow (ASS(\varphi, M\pi) \vee DEN(\varphi, M\pi)) \\
&\leftrightarrow (((ASS(\varphi, \pi) \vee DEN(\varphi, \pi)) \wedge \Diamond ASS(\varphi, \pi)) \\
&\qquad \vee (DEN(\varphi, \pi) \wedge \neg \Diamond ASS(\varphi, \pi))) \\
&\leftrightarrow ((ASS(\varphi, \pi) \vee DEN(\varphi, \pi)) \vee DEN(\varphi, \pi)) \\
&\leftrightarrow (ASS(\varphi, \pi) \vee DEN(\varphi, \pi)) \\
&\leftrightarrow PRES(\varphi, \pi).
\end{aligned}$$

$$\begin{aligned}
&PRES(\varphi, \pi_1 \cup \pi_2) \\
&\leftrightarrow (ASS(\varphi, \pi_1 \cup \pi_2) \vee DEN(\varphi, \pi_1 \cup \pi_2)) \\
&\leftrightarrow (ASS(\varphi, \pi_1) \vee ASS(\varphi, \pi_2) \vee (DEN(\varphi, \pi_1) \wedge DEN(\varphi, \pi_2))) \\
&\leftrightarrow (ASS(\varphi, \pi_1) \vee ASS(\varphi, \pi_2) \\
&\qquad \vee (PRES(\varphi, \pi_1) \wedge \neg ASS(\varphi, \pi_1) \wedge PRES(\varphi, \pi_2) \wedge \neg ASS(\varphi, \pi_2))).
\end{aligned}$$

To express the presuppositions of implications, define $\pi_1 \Rightarrow \pi_2$ as $\neg(\pi_1; \neg\pi_2)$. The presupposition of $\pi_1 \Rightarrow \pi_2$ is now given by:

$$\begin{aligned}
&PRES(\varphi, \pi_1 \Rightarrow \pi_2) \\
&\leftrightarrow PRES(\varphi, \neg(\pi_1; \neg\pi_2)) \\
&\leftrightarrow PRES(\varphi, \pi_1; \neg\pi_2) \\
&\leftrightarrow (PRES(\varphi, \pi_1) \wedge (ASS(\varphi, \pi_2) \rightarrow PRES(ASS(\varphi, \pi_1), \neg\pi_2))) \\
&\leftrightarrow (PRES(\varphi, \pi_1) \wedge (ASS(\varphi, \pi_2) \rightarrow PRES(ASS(\varphi, \pi_1), \pi_2))).
\end{aligned}$$

If one restricts attention to assertions over the language fragment of PUL without the M operator, these calculations can be simplified further, as the following proposition shows.

Proposition 4 *For all φ, for all π without occurrences of M:*

1. $\models ASS(\varphi, \pi) \leftrightarrow (\varphi \wedge ASS(\top, \pi))$.

2. $\models DEN(\varphi, \pi) \leftrightarrow (\varphi \wedge DEN(\top, \pi))$.

Here is a table of presuppositions for the connectives of this restricted language (with $ASS(\pi)$, $DEN(\pi)$, $PRES(\pi)$ as shorthand for $ASS(\top, \pi)$, $DEN(\top, \pi)$, and $PRES(\top, \pi)$, respectively; also, I give the case of $\pi_1 \cup (\neg\pi_1; \pi_2)$ rather than $\pi_1 \cup \pi_2$, as a dynamic version of 'or'):

update	presupposition
$\pi_1; \pi_2$	$PRES(\pi_1) \wedge (ASS(\pi_1) \rightarrow PRES(\pi_2))$
$\neg\pi$	$PRES(\pi)$
$\tilde{\pi}$	$\top$
$\pi_1 \cup (\neg\pi_1; \pi_2)$	$PRES(\pi_1) \wedge (\neg ASS(\pi_1) \rightarrow PRES(\pi_2))$
$\pi_1 \Rightarrow \pi_2$	$PRES(\pi_1) \wedge (ASS(\pi_1) \rightarrow PRES(\pi_2))$

In fact, the presupposition calculus for updates without the M operator is equivalent with the presupposition calculus presented in Van Eijck [10]. This calculus, in turn, agrees with the propositional part of the description of presupposition facts in Karttunen and Peters [20], and with the algebraic theory of presupposition developed by Visser in [32].

6 Presuppositions and Quantification

In the presupposition literature it was not very common until quite recently to present theories of presupposition in the form of explicit compositional frameworks. One classic example is the theory of Karttunen and Peters [20]. This theory separates out assertions from presuppositions, and gives separate instructions for constructing these compositionally from assertions and presuppositions of component parts of speech. This is different from the approach which I have followed above: assertion and presupposition of an update are the success condition and the non-abort condition of that update, and we have as a matter of definition of these concepts that the success condition implies the non-abort condition.

Still, the separation approach works quite well in the propositional case. In fact, the system of Karttunen and Peters covers quite a large range of phenomena, including almost all of those covered by the APUL calculus. (Karttunen and Peters do not cover presuppositions induced by 'maybe', but they do cover the presuppositions of phrases with occurrences of 'even $\langle$*proper name*$\rangle$', which I have not bothered to deal with. A proper treatment of the presuppositions of *even* calls for a worked-out theory of focus and is outside the scope of the present paper.) However, the present approach has the great advantage that it allows a straightforward generalization to the quantificational case.

In a very interesting note at the end of [20], Karttunen and Peters mention the fact that the separation between presupposition and assertion which is the cornerstone of their theory of presupposition, becomes awkard in cases where presuppositions involve quantification or indefinite description (see also Beaver [2] for discussion). Consider example (3).

A bachelor placed a small add. (3)

If you wanted to separate the presupposition from the assertion here, you would probably have to say something like this. The presupposition of (3) is that there is an adult male in the domain of discourse, and the assertion is that *that* person placed a small add. The problem is that, as the anaphoric link (by means of the pronoun *that*) between the presupposition and the assertion already suggests, these two aspects of sentence meaning cannot really be separated.

As I intend to show next, the dynamic approach to presupposition generalizes neatly to the quantificational case, for we can capitalize on the fact that, dynamically speaking, calculating a presupposition is nothing but calculating the disjunction of a success condition and a failure condition, quantifiers and all.

Let us first look at some other quantificational cases of presupposition, which arise in connection with existence and/or uniqueness presuppositions for definite descriptions. Consider the following examples.

John's wife is annoyed. (4)

Presupposition of (4): John is married.

John's wife is not annoyed. (5)

Presupposition of (5): John is married.

John is married. His wife is annoyed with him now. (6)

Presupposition of (6): $\top$.

If John is married, then his wife will be annoyed with him now. (7)

Presupposition of (7): $\top$.

If a woman is married, then her husband looks after her. (8)

Presupposition of (8): $\top$.

A fat man is riding his bicycle. (9)

Presupposition of (9): at least one fat man has a bicycle.

$$\textit{A woman entered. Her husband followed her.} \tag{10}$$

Presupposition of (10): the woman who entered is married.

All these presupposition judgements were intuitive, of course, and indeed, in the presupposition literature such judgements are hotly debated, for here, as elsewhere in linguistics, theoretical bias tends to influence perception of the linguistic facts. See Gazdar [14], Karttunen [19, 18], Van der Sandt [27], Soames [29] for more presupposition data and for further illustration of the diverging judgements on them.

7 Assertions for Dynamic Predicate Logic

Dynamic predicate logic (DPL) is a dynamic variant of first order predicate logic that has been proposed as a medium for natural language representation because of its dynamic way of handling variable binding (Groenendijk and Stokhof [16]). Basically, DPL is the result of replacing existential quantification over a variable x by random assignment to variable x, and conjunction of formulas by sequential composition. *A man walked in. He sat down* can be translated as $x := ?; Mx; Wx; Sx$. This avoids the problem of variable binding that occurs if existential quantifiers and ordinary conjunction are used: $\exists x(Mx \wedge Wx) \wedge S\underline{x}$. Apart from these changes, DPL procedures are like first order formulas, which means that they can be negated. To a computer scientist the concept of the negation of a procedure π may seem strange, but in fact, $\neg\pi$ simply expresses a test: $\neg\pi$ succeeds in precisely those states from which there are no π transitions.

Let C be a set of constants, V a set of variables, and assume $C \cap V = \emptyset$, $c \in C, v \in V$.

DPL terms $t ::= c \mid v$.

Assume a set of relation symbols R with arities.

DPL procedures $\pi ::= t = t \mid Rt \cdots t \mid (\pi;\pi) \mid (\neg\pi) \mid \eta v \mid \iota v : \pi$.

ηv is my notation for $v := ?$. An indefinite noun phrase like *a man* will be represented in DPL as a sequential composition of ηx and Mx. I use $\iota v : \pi$ for definite assignment, a procedure to interpret definite descriptions (see below). While *a man* can be decomposed with ';', such a decomposition is impossible for $\iota x : Kx$ (the DPL translation of the indefinite noun phrase *the king*), as the reader can check below.

DPL procedures are interpreted in ordinary first order models, with assignments (mappings from the set of variables into the universe of the model) functioning as states. Let $M = \langle U, I \rangle$ be a first order model, and let $S = U^V$ (the set of assignments for the model). We use $s(v|d)$ for the assignment which is like s except for the possible difference that v is mapped to d.

The semantic clauses are dynamic, i.e., they define a two-place relation on the set of states:

1. $M, s, s' \models Rt_1 \cdots t_n$ iff $s = s'$ and $M \models_s Rt_1 \cdots t_n$.
2. $M, s, s' \models t_1 = t_2$ iff $s = s'$ and $M \models_s t_1 = t_2$.
3. $M, s, s' \models \pi_1; \pi_2$ iff there is an s'' with $M, s, s'' \models \pi_1$ and $M, s'', s' \models \pi_2$.
4. $M, s, s' \models \neg\pi$ iff $s = s'$ and there is no s'' with $M, s, s'' \models \pi$.
5. $M, s, s' \models \eta v$ iff there is some $d \in U$ with $s' = s(v|d)$.
6. $M, s, s' \models \iota v : \pi$ iff
   - there is a $d \in U$ for which $M, s(v|d), s' \models \pi$,
   - there is a unique $d \in U$ for which $M, s(v|d), s'' \models \pi$ for some s''.

Note that atomic procedures $Rt_1 \cdots t_n$ and $t_1 = t_2$ are interpreted as tests. The procedures ηv and $\iota v : \pi$ do change assignments. Of these, ηv is indeterministic (provided the model has size ≥ 2). The procedure $\iota v : \pi$ is deterministic iff π is deterministic.

A procedure for so-called 'dynamic implication' is defined as follows: $\pi_1 \Rightarrow \pi_2$ abbreviates $\neg(\pi_1; \neg\pi_2)$. It follows from the semantic clauses above that its semantics is given by:

- $M, s, s' \models \pi_1 \Rightarrow \pi_2$ iff
  - $s = s'$, and
  - for all s'' with $M, s, s'' \models \pi_1$ there is an s''' with $M, s'', s''' \models \pi_2$.

An assertion logic of DPL now relates the DPL procedures to first order logic. A suitable syntax for this logic is (terms and basic relations are the same as for the DPL language under consideration; π ranges over DPL procedures):

QDL $\varphi ::= t = t \mid Rt \cdots t \mid \varphi \wedge \varphi) \mid (\neg\varphi) \mid \exists v\varphi \mid \langle \boldsymbol{\pi} \rangle\varphi.$

We use the customary abbrevations for the boolean connectives, plus: $\forall v\varphi := \neg\exists v\neg\varphi$, $\exists! v\varphi := \exists v\forall w(w = v \leftrightarrow \varphi[w/v])$, $[\boldsymbol{\pi}]\varphi := \neg\langle\boldsymbol{\pi}\rangle\neg\varphi$, $x \neq y := (\neg x = y)$, $\bot := \exists x(x \neq x)$, $\top := \neg\bot$.

The semantic clauses for QDL are as for first order logic, with the following addition for the DPL modality:

- $M \models_s \langle\boldsymbol{\pi}\rangle\varphi$ iff there is a state s' with $M, s, s' \models \pi$ and $M \models_{s'} \varphi$.

Axiom schemes relating DPL to this assertion logic take the following shape:

A 7.1 $\langle \boldsymbol{Rt_1 \cdots t_n}\rangle\varphi \leftrightarrow (Rt_1 \cdots t_n \wedge \varphi)$.

A 7.2 $\langle \boldsymbol{t_1 = t_2}\rangle\varphi \leftrightarrow (t_1 = t_2 \wedge \varphi)$.

A 7.3 $\langle \boldsymbol{\pi_1}; \boldsymbol{\pi_2}\rangle\varphi \leftrightarrow \langle\boldsymbol{\pi_1}\rangle\langle\boldsymbol{\pi_2}\rangle\varphi$.

A 7.4 $\langle \boldsymbol{\neg\pi}\rangle\varphi \leftrightarrow ([\boldsymbol{\pi}]\bot \wedge \varphi)$.

A 7.5 $\langle \boldsymbol{\eta v}\rangle\varphi \leftrightarrow \exists v\varphi$.

A 7.6 $\langle \boldsymbol{\iota v} : \boldsymbol{\pi}\rangle\varphi \leftrightarrow (\exists! v\langle\boldsymbol{\pi}\rangle\top \wedge \exists v\langle\boldsymbol{\pi}\rangle\varphi)$.

This axiom system for DPL is analysed in Van Eijck [9]. In fact, the schemes can be used to relate DPL procedures to first order formulas, by computing the success condition $\langle\boldsymbol{\pi}\rangle\top$ of a DPL procedure π. The reader is invited to check that the following scheme for $\Rightarrow$ is derivable from the schemes above:

$$\langle\boldsymbol{\pi_1} \Rightarrow \boldsymbol{\pi_2}\rangle\varphi \leftrightarrow ([\boldsymbol{\pi_1}]\langle\boldsymbol{\pi_2}\rangle\top \wedge \varphi).$$

For the following example, it is also useful to work out some duals of the schemes. Here is a calculation of the success condition of the DPL translation of *If a farmer owns a donkey, he beats it*:

$$\begin{aligned}
&\langle(\boldsymbol{\eta x}; \boldsymbol{Fx}; \boldsymbol{\eta y}; \boldsymbol{Dy}; \boldsymbol{Oxy}) \Rightarrow \boldsymbol{Bxy}\rangle\top \\
&\leftrightarrow [\boldsymbol{\eta x}; \boldsymbol{Fx}; \boldsymbol{\eta y}; \boldsymbol{Dy}; \boldsymbol{Oxy}]\langle\boldsymbol{Bxy}\rangle\top \\
&\leftrightarrow [\boldsymbol{\eta x}][\boldsymbol{Fx}][\boldsymbol{\eta y}][\boldsymbol{Dy}][\boldsymbol{Oxy}]\langle\boldsymbol{Bxy}\rangle\top \\
&\leftrightarrow \forall x(Fx \rightarrow \forall y(Dy \rightarrow (Oxy \rightarrow Bxy))).
\end{aligned}$$

This is indeed a first order rendering of the meaning of the example sentence. Modulo a compositional definition of the DPL meaning representation for the example (which is beyond the scope of this paper, but can easily be given using standard techniques; see e.g. [7, 23]), this illustrates how the assertion logic relates the dynamic meaning of natural language sentences to their static meaning.

The assertion logic for DPL also illustates that the treatment of definite descriptions is just a dynamic version of Russell's well known description theory [26]. Here is a computation of the success condition for the DPL translation of *Some farmer beats his donkey*:

$$\begin{aligned}
&\langle \boldsymbol{\eta x}; \boldsymbol{Fx}; \boldsymbol{\iota y} : (\boldsymbol{Dy}; \boldsymbol{Oxy}); \boldsymbol{Bxy}\rangle\top \\
\leftrightarrow\ &\langle \boldsymbol{\eta x}; \boldsymbol{Fx}; \boldsymbol{\iota y} : (\boldsymbol{Dy}; \boldsymbol{Oxy}); \boldsymbol{Bxy}\rangle\top \\
\leftrightarrow\ &\langle \boldsymbol{\eta x}\rangle\langle \boldsymbol{Fx}\rangle\langle \boldsymbol{\iota y} : (\boldsymbol{Dy}; \boldsymbol{Oxy})\rangle\langle \boldsymbol{Bxy}\rangle\top \\
\leftrightarrow\ &\exists x(Fx \wedge \exists! y(Dy \wedge Oxy) \wedge \exists y(Dy \wedge Oxy \wedge Bxy)).
\end{aligned}$$

8 Dynamic Predicate Logic With Presuppositions

In order to handle the interaction of presupposition and quantification, we 'partialize' the semantics of the representation language, just like in the case of Update Logic. To keep things simple, I start out with just the uniqueness presuppositions of definite descriptions, by assuming that presupposition failures are always introduced by failure of unique reference for definite assignment procedures. In other words, I will assume that e.g. *The king of France is eating frog legs* presupposes that there is a unique thing satisfying the property of being king of France. But nothing hinges on this; it is a trivial modification of the semantics and of the axioms to be developed later to just give them existence presuppositions (so that the example would only presuppose that something has the property of being king of France).

We still interpret in ordinary first order models, with assignments (mappings from the set of variables into the universe of the model) functioning as states, but we now distinguish between success, failure and error abortion of a DPL procedure π for a given input state s in model M, as follows:

π **succeeds for** s There is a state s' with $M, s, s' \models \pi$.

π **fails for** s $M, s, s \mathrel{=\!\!|} \pi$.

π **aborts for** s Neither $M, s, s \mathrel{=\!\!|} \pi$, nor is there a state s' with $M, s, s' \models \pi$.

Here are the semantic clauses for 'DPL with error abortion':

1. $M, s, s' \models Rt_1 \cdots t_n$ iff $s = s'$ and $M \models_s Rt_1 \cdots t_n$.
2. $M, s, s' \mathrel{=\!\!|} Rt_1 \cdots t_n$ iff $s = s'$ and $M \not\models_s Rt_1 \cdots t_n$.

3. $M, s, s' \models t_1 = t_2$ iff $s = s'$ and $M \models_s t_1 = t_2$.
4. $M, s, s' \mathrel{=\!\!|} t_1 = t_2$ iff $s = s'$ and $M \not\models_s t_1 = t_2$.
5. $M, s, s' \models \pi_1; \pi_2$ iff there is an s'' with $M, s, s'' \models \pi_1$ and $M, s'', s' \models \pi_2$.
6. $M, s, s' \mathrel{=\!\!|} \pi_1; \pi_2$ iff $s = s'$ and either $M, s, s \mathrel{=\!\!|} \pi_1$, or
   - there is an s'' with $M, s, s'' \models \pi_1$, and
   - for every s'' with $M, s, s'' \models \pi_1$ it holds that $M, s'', s'' \mathrel{=\!\!|} \pi_2$.
7. $M, s, s' \models \neg\pi$ iff $s = s'$ and $M, s, s \mathrel{=\!\!|} \pi$.
8. $M, s, s' \mathrel{=\!\!|} \neg\pi$ iff $s = s'$ and there is an s'' with $M, s, s'' \models \pi$.
9. $M, s, s' \models \eta x : \pi$ iff there is some $d \in U$ with $M, s(x|d), s' \models \pi$.
10. $M, s, s' \mathrel{=\!\!|} \eta x : \pi$ iff $s = s'$ and $U = \emptyset$.
11. $M, s, s' \models \iota x : \pi$ iff
    - there is a $d \in U$ for which $M, s(x|d), s' \models \pi$, and
    - there is a unique $d \in U$ for which $M, s(x|d), s'' \models \pi$ for some s''.
12. $M, s, s' \mathrel{=\!\!|} \iota x : \pi$ never.

A simple induction establishes the fact that for all π, $M, s, s' \mathrel{=\!\!|} \pi$ implies that $s = s'$.

It is important to note that the above clauses are just one way of 'partializing' DPL. These clauses are equivalent to the error state semantics for DPL given in Van Eijck [8]. Krahmer [21] relates this semantics to other possible ways of partializing DPL. The only difference between the above and what Krahmer calls the 'middle Kleene system of PDPL' is the different convention for encoding failure.

The clauses for $\Rightarrow$ that follow from the above are:

- $M, s, s' \models \pi_1 \Rightarrow \pi_2$ iff
  - $s = s'$,
  - either $M, s, s \mathrel{=\!\!|} \pi_1$ or there is an s'' with $M, s, s'' \models \pi_1$,
  - for all s'' with $M, s, s'' \models \pi_1$ there is an s''' with $M, s'', s''' \models \pi_2$.
- $M, s, s' \mathrel{=\!\!|} \pi_1 \Rightarrow \pi_2$ iff
  - $s = s'$,
  - for some s'' with $M, s, s'' \models \pi_1$ it holds that $M, s'', s'' \mathrel{=\!\!|} \pi_2$.

We also get: $M, s, s' \models \neg\pi$ iff $M, s, s' \models \pi \Rightarrow \textbf{fail}$, and $M, s, s' \mathrel{=\!\!|} \neg\pi$ iff $M, s, s' \mathrel{=\!\!|} \pi \Rightarrow \textbf{fail}$, where **fail** is the procedure which always fails, i.e., which has semantic clauses:

- $M, s, s' \models \textbf{fail}$ never,
- $M, s, s' \mathrel{=\!\!|} \textbf{fail}$ iff $s = s'$.

Here is a different partialization of dynamic implication (this is the connective analyzed in Van Eijck [8]):

- $M, s, s' \models \pi_1 \dot{\Rightarrow} \pi_2$ iff
  - $s = s'$,
  - either $M, s, s \mathrel{=\!\!|} \pi_1$ or there is an s'' with $M, s, s'' \models \pi_1$,
  - for all s'' with $M, s, s'' \models \pi_1$ there is an s''' with $M, s'', s''' \models \pi_2$.
- $M, s, s' \mathrel{=\!\!|} \pi_1 \dot{\Rightarrow} \pi_2$ iff
  - $s = s'$,
  - either $M, s, s \mathrel{=\!\!|} \pi_1$ or there is an s'' with $M, s, s'' \models \pi_1$,
  - for all s'' with $M, s, s'' \models \pi_1$ it holds that $M, s'', s'' \mathrel{=\!\!|} \pi_2$.

9 Reasoning About DPL Presupposition

To reason about presupposition in DPL, we have to extend the assertion language to allow for the possibility of error abortion.

QDLE $\varphi ::= t = t \mid Rt \cdots t \mid \varphi \wedge \varphi) \mid (\neg\varphi) \mid \exists v\varphi \mid \langle\boldsymbol{\pi}\rangle\varphi \mid A(\pi)$.

Abbreviations: as before. Intuitively, $A(\pi)$ expresses that π aborts with error. The semantics for QDLE is as for QDL, with the following clause for $A(\pi)$ added:

- $M \models_s A(\pi)$ iff for all s': $M, s, s' \not\mathrel{=\!\!|} \pi$ and $M, s, s' \not\models \pi$.

Note that the dual $[\boldsymbol{\pi}]\varphi$ of $\langle\boldsymbol{\pi}\rangle\varphi$ is interpreted like this:

- $M \models_s [\boldsymbol{\pi}]\varphi$ iff for all s' with $M, s, s' \models \pi$ it holds that $M \models_{s'} \varphi$.

Thus, from $[\boldsymbol{\pi}]\varphi$ it does *not* follow that π does not abort with error! On the other hand, $\langle\boldsymbol{\pi}\rangle\varphi$ does entail that π does not abort with error, because it asserts the existence of a $\pi+$ transition to a state where φ holds. It should be mentioned here that this assertion language is different from the one used in [8], where $[\boldsymbol{\pi}]$ is given a stronger interpretation, and where $A(\pi)$ is a derived notion.

The addition of $A(\pi)$ assertions provides us with the tool to distinguishing failure from error abortion in the assertion logic, for we have:

- $\langle\boldsymbol{\pi}\rangle\top$ iff π succeeds in the current state,
- $[\boldsymbol{\pi}]\bot \wedge \neg A(\pi)$ iff π fails in the current state,
- $A(\pi)$ iff π aborts in the current state.

The presupposition of a procedure π is now given as $\neg A(\pi)$.

Here are axiom schemes for DPL with error abortion:

A 9.1 $\langle \boldsymbol{Rt_1 \cdots t_n}\rangle\varphi \leftrightarrow (Rt_1 \cdots t_n \wedge \varphi)$.

A 9.2 $A(Rt_1 \cdots t_n) \leftrightarrow \bot$.

A 9.3 $\langle \boldsymbol{t_1 = t_2}\rangle\varphi \leftrightarrow (t_1 = t_2 \wedge \varphi)$.

A 9.4 $A(t_1 = t_2) \leftrightarrow \bot$.

A 9.5 $\langle \boldsymbol{\pi_1; \pi_2}\rangle\varphi \leftrightarrow \langle\boldsymbol{\pi_1}\rangle\langle\boldsymbol{\pi_2}\rangle\varphi$.

A 9.6 $A(\pi_1;\pi_2) \leftrightarrow (A(\pi_1) \vee ([\boldsymbol{\pi_1}][\boldsymbol{\pi_2}]\bot \wedge \langle\boldsymbol{\pi_1}\rangle A(\pi_2)))$.

A 9.7 $\langle \boldsymbol{\neg\pi}\rangle\varphi \leftrightarrow (\neg A(\pi) \wedge [\boldsymbol{\pi}]\bot \wedge \varphi)$.

A 9.8 $A(\neg\pi) \leftrightarrow A(\pi)$.

A 9.9 $\langle \boldsymbol{\eta v}\rangle\varphi \leftrightarrow \exists v\varphi$.

A 9.10 $A(\eta v) \leftrightarrow \neg\exists v\top$.

A 9.11 $\langle \boldsymbol{\iota v : \pi}\rangle\varphi \leftrightarrow (\exists! v\langle\boldsymbol{\pi}\rangle\top \wedge \exists v\langle\boldsymbol{\pi}\rangle\varphi)$.

A 9.12 $A(\iota v : \pi) \leftrightarrow \neg\exists! v\langle\boldsymbol{\pi}\rangle\top$.

Theorem 5 *The calculus is sound for DPL with error abortion.*

Proof. Check that the schemes hold in all models.

The reasoning for the abortion scheme for sequential composition:

$M \models_s A(\pi_1; \pi_2)$ iff for all s': $M, s, s' \not\dashv \pi_1; \pi_2$ and $M, s, s' \not\models \pi_1; \pi_2$, iff either for all s': $M, s, s' \not\dashv \pi_1$ and $M, s, s' \not\models \pi_1$, or

- there are no s', s'' with $M, s, s' \models \pi_1$ and $M, s', s'' \models \pi_2$, and
- there is an s' with $M, s, s' \models \pi_1$ and for all s'': $M, s', s'' \not\dashv \pi_2$ and $M, s', s'' \not\models \pi_2$.

iff (induction hypothesis) either $M \models_s A(\pi_1)$ or $M \models_s [\boldsymbol{\pi}_1][\boldsymbol{\pi}_2]\bot \wedge \langle \boldsymbol{\pi}_1 \rangle A(\pi_2)$, iff $M \models_s A(\pi_1) \vee ([\boldsymbol{\pi}_1][\boldsymbol{\pi}_2]\bot \wedge \langle \boldsymbol{\pi}_1 \rangle A(\pi_2))$.

The reasoning for the negation axioms: $M \models_s \langle \neg \boldsymbol{\pi} \rangle \varphi$ iff there is an s' with $M, s, s' \models \neg\pi$ and $M \models_{s'} \varphi$ iff $M, s, s \dashv \pi$ and $M \models_s \varphi$ iff (induction hypothesis) $M \models_s \neg A(\pi) \wedge [\boldsymbol{\pi}]\bot$ and $M \models_s \varphi$ iff $M \models_s \neg A(\pi) \wedge [\boldsymbol{\pi}]\bot \wedge \varphi$.

$M \models_s A(\neg\pi)$ iff for all s', $M, s, s' \not\dashv \neg\pi$ and $M, s, s' \not\models \neg\pi$, iff for all s', $M, s, s' \not\dashv \pi$ and $M, s, s' \not\models \pi$, iff (induction hypothesis) $M \models_s A(\pi)$.

And here is the case of definite assignment: $M \models_s \langle \boldsymbol{\iota v} : \boldsymbol{\pi} \rangle \varphi$ iff there is an s' with $M, s, s' \models \iota v : \pi$ and $M \models_{s'} \varphi$ iff there is an s' and there is a $d \in U$ with $M, s(v|d), s' \models \pi$ and $M \models_{s'} \varphi$, and moreover, there is a unique $d \in U$ for which $M, s(v|d), s'' \models \pi$ for some s''. This is the case iff $M \models_{s(v|d)} \langle \boldsymbol{\pi} \rangle \varphi$ and therefore $M \models_s \exists v \langle \boldsymbol{\pi} \rangle \varphi$, and moreover $M \models_s \exists! v \langle \boldsymbol{\pi} \rangle \top$, iff $M \models_s \exists! v \langle \boldsymbol{\pi} \rangle \top \wedge \exists v \langle \boldsymbol{\pi} \rangle \varphi$.

$M \models_s A(\iota v : \pi)$ iff for all s': $M, s, s' \not\dashv \iota v : \pi$ and $M, s, s' \not\models \iota v : \pi$, iff for all s': $M, s, s' \not\dashv \iota v : \pi$, iff there is no unique $d \in U$ with $M, s(v|d), s'$ (for some s'), iff (induction hypothesis) there is no unique $d \in U$ with $M \models_{s(v|d)} \langle \boldsymbol{\pi} \rangle \top$, iff $M \models_s \neg \exists! v \langle \boldsymbol{\pi} \rangle \top$. □

To reason with this presupposition calculus, it is useful to derive some additional principles:

T 9.1 $[\neg \boldsymbol{\pi}]\varphi \leftrightarrow (A(\pi) \vee ([\boldsymbol{\pi}]\bot \to \varphi))$.

T 9.2 $\langle \boldsymbol{\pi}_1 \Rightarrow \boldsymbol{\pi}_2 \rangle \varphi \leftrightarrow (\neg A(\pi_1) \wedge [\boldsymbol{\pi}_1]\langle \boldsymbol{\pi}_2 \rangle \top \wedge \varphi)$.

T 9.3 $[\boldsymbol{\pi}_1 \Rightarrow \boldsymbol{\pi}_2]\varphi \leftrightarrow (A(\pi_1) \vee ([\boldsymbol{\pi}_1]\langle \boldsymbol{\pi}_2 \rangle \top \to \varphi))$.

T 9.4 $A(\pi_1 \Rightarrow \pi_2) \leftrightarrow (A(\pi_1) \vee ([\boldsymbol{\pi}_1](\neg A(\pi_2) \to \langle \boldsymbol{\pi}_2 \rangle \top) \wedge \langle \boldsymbol{\pi}_1 \rangle A(\pi_2)))$.

T 9.5 $[\boldsymbol{\iota v} : \boldsymbol{\pi}]\varphi \leftrightarrow (\exists! v\langle\boldsymbol{\pi}\rangle\top \rightarrow \forall v[\boldsymbol{\pi}]\varphi)$.

The calculus can be used to calculate presuppositions and abort conditions for natural language examples with definite descriptions. We will give calculations for two examples:

$$\textit{A fat man is riding his bicycle.} \tag{11}$$

$$\textit{If a woman is married, her husband looks after her.} \tag{12}$$

Without entering into details of a translation procedure from natural language to DPL [7], we state that examples (11) and (12) have the following DPL translations (tense is disregarded).

$$\eta x; Mx; Fx; \iota y : Byx; Rxy. \tag{13}$$

$$(\eta x; Wx; Mx) \Rightarrow (\iota y : Hyx; Lyx). \tag{14}$$

The abort condition for DPL translation (13) of example (11) is calculated as follows.

$$
\begin{array}{l}
A(\eta x; Mx; Fx; \iota y : Byx; Rxy) \\
\leftrightarrow A(\eta x; Mx; Fx) \vee ([\boldsymbol{\eta x}; \boldsymbol{Mx}; \boldsymbol{Fx}][\boldsymbol{\iota y} : \boldsymbol{Byx}; \boldsymbol{Rxy}]\bot \\
\quad \wedge \langle \boldsymbol{\eta x}; \boldsymbol{Mx}; \boldsymbol{Fx}\rangle A(\iota y : Byx; Rxy)) \\
\leftrightarrow \forall x(Mx \rightarrow (Fx \rightarrow (\exists! y(Byx \rightarrow \forall y(Byx \rightarrow \neg Rxy) \\
\quad \wedge \exists x(Mx \wedge Fx \wedge A(\iota y : Byx; Rxy)) \\
\leftrightarrow \forall x(Mx \rightarrow (Fx \rightarrow (\exists! y(Byx \rightarrow \forall y(Byx \rightarrow \neg Rxy) \\
\quad \wedge \exists x(Mx \wedge Fx \wedge A(\iota y : Byx)) \\
\leftrightarrow \forall x(Mx \rightarrow (Fx \rightarrow (\exists! y(Byx \rightarrow \forall y(Byx \rightarrow \neg Rxy) \\
\quad \wedge \exists x(Mx \wedge Fx \wedge \neg \exists! y Byx)
\end{array}
$$

The presupposition of (11) is the negation of this formula:

$$\exists x(Mx \wedge Fx \wedge \exists! y Byx \wedge \exists y(Byx \wedge Rxy)) \vee \forall x(Mx \rightarrow (Fx \rightarrow \exists! y Byx)).$$

Note that this result is different from what one gets in a treatment following Heim [17], where it is assumed that one can only update with an assertion in contexts where the presupposition is satisfied. From this it would follow that the presupposition for the definite should hold in all contexts with a fat man in it, in other words, it predicts as presupposition for (11) that all fat men have a bicycle.

The abort condition for example procedure (14) is calculated as follows.

$$
\begin{array}{l}
A((\eta x; Wx; Mx) \Rightarrow (\iota y : Hyx; Lyx)) \\
\leftrightarrow A(\eta x; Wx; Mx) \\
\quad \vee([\boldsymbol{\eta x}; \boldsymbol{Wx}; \boldsymbol{Mx}](\neg A(\iota y : Hyx; Lyx) \rightarrow \langle \boldsymbol{\iota y} : \boldsymbol{Hyx}; \boldsymbol{Lyx}\rangle\top) \\
\quad \wedge \langle \boldsymbol{\eta x}; \boldsymbol{Wx}; \boldsymbol{Mx}\rangle A(\iota y : Hyx; Lyx)) \\
\leftrightarrow \forall x(Wx \rightarrow (Mx \rightarrow (\exists! y Hyx \rightarrow \exists y(Hyx \wedge Lyx)))) \\
\quad \wedge \exists x(Wx \wedge Mx \wedge \neg \exists! y Hyx).
\end{array}
$$

Again, the presupposition is expressed by the negation of this formula:

$$\exists x(Wx \wedge Mx \wedge \exists! yHyx \wedge \neg\exists y(Hyx \wedge Lyx)) \vee$$
$$\forall x(Wx \rightarrow (Mx \rightarrow \exists! yHyx).$$

Note that the second disjunct of this presupposition is vacuously true, for every married woman does have one and only one husband. Technically, this gets ensured by suitable meaning postulates for 'woman', 'married' and 'husband'. It follows from these meaning postulates that the presupposition of (14) reduces to $\top$.

10 Lexical Presuppositions

The calculus for DPL with error abortion can easily be extended to examples of lexical presupposition, such as the *bachelor* example of Section 4.

We assume two functions $\mathbf{la}, \mathbf{lp} : At \rightarrow QDLE$. **la** maps every atomic DPL relation test to its lexical assertion; **lp** maps every atomic DPL relation test to its lexical presupposition. We assume that the lexical assertion of every atom A logically implies its lexical presupposition. For example, (15) and (16) give the lexical presupposition and assertion for bachelorhood.

$$\mathbf{lp}(\mathit{bachelor}\ x) = (\mathit{male}\ x \wedge \mathit{adult}\ x). \tag{15}$$

$$\mathbf{la}(\mathit{bachelor}\ x) = (\mathit{male}\ x \wedge \mathit{adult}\ x \wedge \neg \mathit{married}\ x). \tag{16}$$

The semantic clause for atomic relation tests can now be modified to take the functions **la** and **lp** into account, as follows:

- $M, s, s' \models Rt_1 \cdots t_n$ iff $s =' s$ and $M \models_s \mathbf{la}(Rt_1 \cdots t_n)$.
- $M, s, s' \dashv Rt_1 \cdots t_n$ iff $s = s'$, $M \models_s \mathbf{lp}(Rt_1 \cdots t_n)$, and $M \not\models_s \mathbf{la}(Rt_1 \cdots t_n)$.

The axioms for atomic tests become:

A 10.1 $\langle \boldsymbol{Rt_1 \cdots t_n} \rangle \varphi \leftrightarrow (\mathbf{la}(Rt_1 \cdots t_n) \wedge \varphi)$.

A 10.2 $A(Rt_1 \cdots t_n) \leftrightarrow \neg \mathbf{lp}(Rt_1 \cdots t_n)$.

As an example application, we calculate the presupposition of (17).

A man placed a small add. It was a bachelor. (17)

The corresponding DPL procedure (M for being a man, i.e., being male, P for placing a small add, and B for being a bachelor):

$$\eta x; Mx; Px; Bx. \tag{18}$$

For the presupposition calculation, we assume that $\mathbf{lp}(Mx) = \top$, $\mathbf{la}(Mx) = Mx$, $\mathbf{lp}(Px) = \top$, $\mathbf{la}(Px) = Px$, $\mathbf{lp}(Bx) = Mx \wedge Ax$ (A for being adult), $\mathbf{la}(Bx) = Mx \wedge Ax \wedge \neg M'x$ (M' for being married). Then the abort condition is calculated like this:

$$\begin{aligned}
&A(\eta x; Mx; Px; Bx) \\
&\leftrightarrow A(\eta x) \vee ([\boldsymbol{\eta x}][\boldsymbol{Mx}; \boldsymbol{Px}; \boldsymbol{Bx}]\bot \wedge \langle \boldsymbol{\eta x}\rangle A(Mx; Px; Bx)) \\
&\leftrightarrow \neg\exists v\top \vee (\forall x(Mx \rightarrow (Px \rightarrow \neg(Mx \wedge Ax \wedge \neg M'x)))\wedge \\
&\quad \exists x A(Mx; Px; Bx)) \\
&\leftrightarrow \neg\exists v\top\vee \\
&\quad (\forall x(Mx \rightarrow (Px \rightarrow \neg(Mx \wedge Ax \wedge \neg M'x))) \wedge \exists x(Mx \wedge Px \wedge A(Bx)) \\
&\leftrightarrow \neg\exists v\top\vee \\
&\quad (\forall x(Mx \rightarrow (Px \rightarrow \neg(Mx \wedge Ax \wedge \neg M'x))) \wedge \exists x(Mx \wedge Px \wedge \neg Ax)).
\end{aligned}$$

Assuming that the domain is non-empty, we get as presupposition:

$$\exists x(Mx \wedge Px \wedge Ax \wedge \neg M'x) \vee \forall x((Mx \wedge Px) \rightarrow Ax).$$

Thus, according to this calculation, (17) presupposes that either some adult male unmarried individual placed a small add, or else every male individual in the domain who placed a small add is an adult.

11 Presuppositions of Performance Verbs

As an example of a presupposition of a 'performance' verb we discuss a sentence from the concluding note of Karttunen and Peters [20].

$$\textit{Someone managed to succeed George V.} \tag{19}$$

If we want to formalize this in DPL, we need a presupposition for $M(x, \pi)$ ('x manages to do π'). Intuitively, the presupposition should be: $D(x, \pi)$, for 'it is difficult for x to do π'. We can process this procedure further if we are prepared to decompose it into: 'if x does π then x has a difficulty'. This leads to the presupposition $\langle \boldsymbol{\pi}\rangle\top \rightarrow Dx$ for $M(x, \pi)$, which is close enough for our purposes. If we take $Dx \wedge \langle \boldsymbol{\pi}\rangle\top$ as assertion, then the assertion implies the presupposition, as it should. In the semantic clause we take the 'context change potential' of π into account as well. This gives:

- $M, s, s' \models M(x, \pi)$ iff $M \models_s Dx$ and $M, s, s' \models \pi$.

- $M, s, s' \dashv M(x, \pi)$ iff $s = s'$, there is an s'' with $M, s, s'' \models \pi$, and $M \neg \models_s Dx$.

In line with the axiom schemas for test relations with presuppositions, we get the following axioms.

A 11.1 $\langle \boldsymbol{M(x, \pi)} \rangle \varphi \leftrightarrow (Dx \wedge \langle \boldsymbol{\pi} \rangle \varphi)$.

A 11.2 $A(x, \pi) \leftrightarrow (\langle \boldsymbol{\pi} \rangle \top \wedge \neg Dx)$.

We can now calculate the abort condition of (19).

$$\begin{array}{l} A(\eta x; M(x, Sxg)) \\ \leftrightarrow A(\eta x) \vee ([\boldsymbol{\eta x}][\boldsymbol{M(x, Sxg)}]\bot \wedge \langle \boldsymbol{\eta x} \rangle A(M(x, Sxg))) \\ \leftrightarrow \neg \exists x \top \vee (\forall x(Dx \rightarrow \neg Sxg) \wedge \exists x(Sxg \wedge \neg Dx)) \end{array}$$

Assuming that the domain is non-empty, this is the presupposition of the example:

$$\exists x(Sxy \wedge Dx) \vee \forall x(Sxg \rightarrow Dx).$$

Thus, the sentence presupposes that either someone succeeded George V and had difficulty in doing so, or everyone who succeeded him did so with difficulty. This explains Karttunen and Peters' observation that the sentence sounds awkward, for it is actually false that the one who did succeed George V had difficulty in doing so. Karttunen and Peters remark that changing their rules to give the sentence the presupposition $\exists x(Sxg \wedge Dx)$ leads to problems with examples like (20).

If someone managed to succeed George V, (20)
then England is still a monarchy.

This example would then have as presupposition that someone did succeed to the throne, which is intuitively incorrect. Our finding that the presupposition of (19) is not $\exists x(Sxg \wedge Dx)$, but rather $\exists x(Sxg \wedge Dx) \vee \forall x(Sxg \rightarrow Dx)$ makes this problem vanish.

12 Conclusion

My specific contribution to the dynamics of presupposition is the demonstration of the use of assertion statements for presupposition projection. What I hope to have shown in this paper is that the tools of assertion logic and dynamic logic are eminently suited to analyse phenomena of context change in natural language. To be sure, the dynamic perspective on presupposition has to be worked into a full account, by a careful examination of further cases of aspectual presupposition, presuppositions of factive verbs, presupposition projection in the context of attitude verbs, presupposition and focus (presuppositions of *even* and *only*), and so on.

Application to natural language is not the only promise that reasoning about presuppositions via assertion logic has in store, however. Such reasoning can also be used for analysing imperative programming. Indeed, the dynamic error logic framework of this paper points the way to a modern alternative for the use of Hoare logic in Tucker and Zucker [30] for the analysis of error abortion in imperative programs.

Acknowledgements I wish to thank colleagues and students in Amsterdam and Utrecht for helpful comments. Colleague Albert Visser and student Yvon Wijnen deserve particular mention. Also thanks to the two anonymous referees of this Journal for their helpful feedback.

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] D.I. Beaver. The kinematics of presupposition. In P. Dekker and M. Stokhof, editors, *Proceedings of the Eighth Amsterdam Colloquium*, pages 17–36. ILLC, University of Amsterdam, 1992.

[3] D.I. Beaver. *What Comes First in Dynamic Semantics.* PhD thesis, University of Edinburgh, 1993.

[4] 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.

[5] J. van Benthem. General dynamics. *Theoretical Linguistics*, 17:159–201, 1991.

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

[7] O. Bouchez, J. van Eijck, and O. Istace. A strategy for dynamic interpretation: a fragment and an implementation. In S. Krauwer, M. Moortgat,

and Louis des Tombe, editors, *Sixth Conference of the European Chapter of the Association for Computational Linguistics — Proceedings of the Conference*, pages 61–70. ACL, 1993.

[8] J. van Eijck. The dynamics of description. *Journal of Semantics*, 10:239–267, 1993.

[9] J. van Eijck. Axiomatizing dynamic predicate logic with quantified dynamic logic. In J. van Eijck and A. Visser, editors, *Logic and Information Flow*, pages 30–48. MIT Press, Cambridge, Mass., 1994.

[10] J. van Eijck. Presuppositions and dynamic logic. Technical Report CSLI-94-186, CSLI, Stanford, February 1994. To appear in M. Kanazawa, C. Piñon, H. de Swart (eds.), Papers from the 2nd CSLI Workshop on Logic, Language and Computation, June 1993.

[11] J. van Eijck and G. Cepparello. Dynamic modal predicate logic. Technical Report OTS-WP-CL-93-005, OTS, Utrecht, October 1993. To appear in M. Kanazawa and C.J. Piñon (eds.), Dynamics, Polarity, and Quantification, CSLI, Stanford.

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

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

[14] G. Gazdar. A solution to the projection problem. In C.-K. Oh and D. Dinneen, editors, *Syntax and Semantics 11: Presupposition*, pages 57–89. Academic Press, 1979.

[15] 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.

[16] J. Groenendijk and M. Stokhof. Dynamic predicate logic. *Linguistics and Philosophy*, 14:39–100, 1991.

[17] I. Heim. On the projection problem for presuppositions. *Proceedings of the West Coast Conference on Formal Linguistics*, 2:114–126, 1983.

[18] L. Karttunen. Presuppositions of compound sentences. *Linguistic Inquiry*, 4:169–193, 1973.

[19] L. Karttunen. Presupposition and linguistic context. *Theoretical Linguistics*, pages 181–194, 1974.

[20] L. Karttunen and S. Peters. Conventional implicature. In C.-K. Oh and D. Dinneen, editors, *Syntax and Semantics 11: Presupposition*, pages 1–56. Academic Press, 1979.

[21] E. Krahmer. Partiality and dynamics; theory and application. In P. Dekker and M. Stokhof, editors, *Proceedings 9th Amsterdam Colloquium*, pages 391–410. ILLC, Amsterdam, 1994.

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

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

[24] S. Peters. A truth-conditional formulation of Karttunen's account of presupposition. *Texas Linguistic Forum*, pages 137–149, 1977.

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

[26] B. Russell. On denoting. *Mind*, 14:479–493, 1905.

[27] R.A. van der Sandt. *Context and Presupposition.* Croom Helm, London, 1988.

[28] P. Seuren. Präsuppositionen. In A. von Stechow and D. Wunderlich, editors, *Semantics, An International Handbook of Contemporary Research*, pages 286–318. De Gruyter, Berlin, 1991.

[29] S. Soames. Presupposition. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, pages 553–616. Reidel, 1984. Volume IV.

[30] J.V. Tucker and J.I. Zucker. *Program Correctness over Abstract Data Types, with Error State Semantics.* North Holland, 1988.

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

[32] A. Visser. Actions under presuppositions. In J. van Eijck and A. Visser, editors, *Logic and Information Flow*, pages 196–233. MIT Press, 1994.

[33] H. Zeevat. Presupposition and accommodation in update semantics. *Journal of Semantics*, 9(4):379–412, 1992. Special Issue: Presupposition, Part 2.