Dynamic Semantics

Marcus Kracht *

II. Mathematisches Institut
Freie Universität Berlin
Arnimallee 3
D–14195 Berlin
`kracht@math.fu-berlin.de`

1 Introduction

Dynamic semantics is called 'dynamic' because it assumes that the meaning of a sentence is not its truth condition but rather its impact on the hearer. In contrast to standard semantics in terms of predicate logic (from now on also called *static semantics*), where formulae are interpreted as conditions on models, dynamic semantics interprets formulae as update functions on databases. The change from the static to the dynamic view was necessitated by problems concerning extrasentential anaphors, but nowadays many more applications of this new semantics have been found. We will begin however with the classical problem. Consider the following examples.

(1) There is a fat man at the metro entrance. He is selling souvenirs.

(2) If Alfred has a car, he washes it every weekend.

In Montague semantics, following the philosophical tradition, a sentence expresses a proposition. A proposition corresponds to a closed formula in predicate logic. Using some self–explanatory abbreviations, the above sentences may be rendered as follows.

*I have benefitted from enlightening discussions with Fritz Hamm, Peter Staudacher, Kees Vermeulen and Albert Visser.

(1a) $(\exists x)(\mathsf{fat–man}(x) \wedge \mathsf{at–metro–entrance}(x) \wedge \mathsf{sell–souvenirs}(x))$

(2a) $(\forall x)(\mathsf{car}(x) \wedge \mathsf{own}(a, x) \to \mathsf{wash–every–week}(a, x))$

In the translation we have ignored certain details such as tense. The problem is to arrive at the given translations in a systematic way, that is, using λ–expressions as in Montague–semantics. Let us illustrate this with (1). (1) is composed from two sentences; each of them is translated by a closed formula. If we assume that the meaning of two sentences in sequence is simply the conjunction of the meanings of the individual sentences we get the following translation.

(1b.1) $(\exists x)(\mathsf{fat–man}(x) \wedge \mathsf{at–metro–entrance}(x))$

(1b.2) $(\exists x)\mathsf{sell–souvenirs}(x)$

(1b) $(\exists x)(\mathsf{fat–man}(x) \wedge \mathsf{at–metro–entrance}(x)) \wedge (\exists x)\mathsf{sell–souvenirs}(x)$

The transliteration of (1b) is *there is a fat man at the metro entrance and there is someone selling souvenirs* — which is not the meaning of (1). The problem is that (1b) can be satisfied when there is a fat man at the metro entrance and someone different, who is not fat but sells souvenirs. Notice that the choice of the variable in (1b.1) and (1b.2) is completely immaterial. We can replace x by any other variable.

Obviously, the problem lies in the translation of the pronoun. In our previous attempt we have tacitly assumed that a pronoun is to be translated by a variable. This strategy fails. Similar problems arise with sentence (2) above. Now what can be done? One solution is to interpret a pronoun as a covert definite description. Let us consider again example (1). After hearing the second sentence we may ask ourselves who is meant by *he*. The obvious answer is: *the fat man at the metro entrance*. So, rather than picking up a referent by means of a syntactic variable we may pick it up by a suitable definite description. Without going into details we may note that under this strategy (1) turns out to be synonymous with (3).

(3) There is a fat man at the metro entrance. The fat man at the metro entrance is selling souvenirs.

This, however, is not without problems. For suppose there are two fat men at the metro entrance and only one is selling souvenirs. Then (1) is still true, while (3) is false since the definite description *the fat man at the metro entrance* fails to refer.

Another possibility is to use open formulae as translations. Rather than translating (1) as (1a) we may simply translate it as

(1c) $\mathsf{fat\text{–}man}(x) \wedge \mathsf{at\text{–}metro\text{–}entrance}(x) \wedge \mathsf{sell\text{–}souvenirs}(x)$

This allows to derive (1c) in a systematic way from the meaning of the two sentences, which we give as (1c.1) and (1c.2)

(1c.1) $\mathsf{fat\text{–}man}(x) \wedge \mathsf{at\text{–}metro\text{–}entrance}(x)$

(1c.2) $y = x \wedge \mathsf{sell\text{–}souvenirs}(y)$

Here, *selling souvenirs* is rendered as $\mathsf{sell\text{–}souvernirs}(y)$, where y is a fresh variable, and the pronoun *he* is rendered $y = x$. Again we will not go into details here. The truth conditions for (1c) are different from the standard conditions in predicate logic, where a free variable is treated as if universally quantified. Rather, a free variable is treated as if *existentially quantified.* Let us say that (1c) is true in a model $\mathfrak{M}$ under a valuation g if there exists a valuation h differing from g in at most x such that (1c) is true in $\mathfrak{M}$ under the valuation h. This is, modulo some minor variation, the solution presented in Discourse Representation Theory (DRT) (see [4]). Note especially how it assigns truth conditions to an implication such as (2). Namely, an implication $\phi \Rightarrow \psi$ [1] is true in a model $\mathfrak{M}$ under a valuation g if for every valuation h differing from g in at most the free variables of ϕ that makes ϕ true there is a valuation k differing in at most the free variables of ψ that are not free in ϕ such that that k makes ψ true. Let us apply this to the translation of (2), (2c).

(2c) $\mathsf{car}(x); \mathsf{own}(a, x) \Rightarrow y = x; \mathsf{wash\text{–}every\text{–}week}(a, y)$

(2c) is true in $\langle \mathfrak{M}, g\rangle$ if for every h differing from g in at most x, such that h assigns for example j to x and j is a car and Alfred owns it in $\mathfrak{M}$, then there is a valuation k differing from h in at most y such that $k(y) = k(x)(= j)$ and Alfred washes $k(y)$ every day. So, (2c) is true iff Alfred washes every car that he owns every week. This is exactly as it should be.

This definition of truth anticipates certain features of dynamic semantics. Although it still employs the static notion of satisfaction (or truth) in a model it already works with dynamically changing assignments. In predicate logic, only the truth conditions for quantifiers allow for a change in assignments, while in DRT the standard logical connectives may also change them. Compare, for example, the truth condition for an implication in predicate logic with that of DRT. In the former, $\phi \rightarrow \psi$ is true in $\langle \mathfrak{M}, g\rangle$ if either ϕ is false or ψ is true in it. The dynamic character of the simple connectives also allows DRT to dispense with

[1] We write $\Rightarrow$ for the implication and ; for the conjunction of DRT in order not to get confused with the standard symbols of predicate logic.

the usual quantifiers. For notice that $(\exists x)\phi(x)$ is equivalent to $\phi(x)$ and $(\forall x)\phi(x)$ is equivalent to $x = x \Rightarrow \phi(x)$. Consequently, DRT dispenses with quantifiers and only introduces a separate head section in a DRS to annotate for which variables the valuation may be changed. So, rather than $\phi(x)$ we write $[x : \phi(x)]$. In this way we can distinguish between a contextually unbound variable (corresponding to an indefinite description) and a contextually bound variable (for example in translating a pronoun by an expression $y = x$, where x has appeared already).

2 Dynamic Predicate Logic

In DRT, we have no quantifiers, only conditions on assignments. This may solve the problem for the indefinite expressions and the existential quantifiers. However, DRT has no analogue for the universal quantifier. The reduction of the universal quantifier to an implication is merely a formal trick and can in fact not be used for other quantifiers. Several people have noticed independently that the problems of anaphoric reference can only be solved if we allow to memorize the value given to a certain variable. In DRT this is achieved by simply removing the quantifier and readjusting the satisfaction clauses for free formulae. Yet another path was followed by Peter Staudacher and, somewhat later but independently, Jeroen Groenendijk and Martin Stokhof. (See [9] and [1]. A comparison of the two approaches can be found in [10].) Hence our basic vocabulary consists of

1. a set *Var* of variables over individuals,
2. a set *Con* of constants for individuals,
3. some atomic relation symbols,
4. the boolean connectives $\top$, $\bot$, $\neg$, $\wedge$, $\vee$, $\rightarrow$,
5. the quantifiers $\exists$ and $\forall$.

(That we have no function symbols is just a question of simplicity. In the formal definitions we will also often ignore the constants to keep the notation simple.) As usual, a well–formed formula (simply called a *formula*) is defined by induction.

1. If R is an n–ary relation symbol and $u_i \in Var \cup Con$ for all $1 \leq i \leq n$, then $R(u_1, \ldots, u_n)$ is a formula.
2. $\top$ and $\bot$ are formulae.
3. If ϕ is a formula, so is $\neg\phi$.

4. If ϕ and ψ are formulae then so are $\phi \wedge \psi$, $\phi \vee \psi$ and $\phi \to \psi$.

5. If x is a variable and ϕ a formula then $(\exists x)\phi$ and $(\forall x)\phi$ are formulae.

Definition 1 *A* ***model*** *is a pair $\langle D, I\rangle$, where D is a set, called the* ***domain****, and I a function, the* ***interpretation function****, assigning to each constant an element of D and to each n–ary relation R of the language a subset of D^n. An* ***assignment*** *or* ***valuation*** *is a function $g : Var \to D$. The set of all assignments into D is denoted by $\mathbb{V}(D)$.*

We write $\langle \mathfrak{M}, g\rangle \models \phi$ if ϕ is true in $\mathfrak{M}$. This is defined by induction. On the basis of that we define the **static meaning** of ϕ, $[\phi]_{\mathfrak{M}}$ or simply $[\phi]$, to be

$$[\phi] := \{g : \langle \mathfrak{M}, g\rangle \models \phi\}$$

Let us write $g \sim_x h$ if $g(y) = h(y)$ for all $y \neq x$.

Definition 2 (Static Meaning) *Given a model $\mathfrak{M}$, the static meaning of a formula ϕ is computed as follows.*

$$\begin{array}{lcl}
[R(x_1, \ldots, x_n)] & := & \{g : \langle g(x_1), \ldots, g(x_n)\rangle \in I(R)\} \\
[\top] & := & \mathbb{V}(D) \\
[\bot] & := & \varnothing \\
[\neg\phi] & := & \mathbb{V}(D) - [\phi] \\
[\phi \vee \psi] & := & [\phi] \cup [\psi] \\
[\phi \wedge \psi] & := & [\phi] \cap [\psi] \\
[\phi \to \psi] & := & (\mathbb{V}(D) - [\phi]) \cup [\psi] \\
[(\exists x)\phi] & := & \{g : \textit{exists } h \sim_x g : h \in [\phi]\} \\
[(\forall x)\phi] & := & \{g : \textit{for all } h \sim_x g : h \in [\phi]\}
\end{array}$$

The idea of the dynamic interpretation is to keep the full syntax of predicate logic and instead change the underlying notion of meaning. Rather than talking of an assignment g in a model $\mathfrak{M}$ making a formula true we will now talk of ϕ being processable or unprocessable under the assignment g. If ϕ is processable we may further speak of ϕ taking us from the assignment g to an assignment h. We will assume that the meaning of a formula ϕ of predicate logic is a *binary relation* on the set of assignments of variables. We discuss this with our examples (1) and (2). We translate (1) and (2) now as follows.

(1d) $(\exists x)(\mathsf{fat–man}(x) \wedge \mathsf{at–metro–entrance}(x)) \wedge \mathsf{sell–souvenirs}(x)$

(2d) $(\exists x)(\mathsf{car}(x) \wedge \mathsf{own}(a, x)) \to \mathsf{wash–every–week}(a, x)$

Notice that the last occurrence of x in (1d) is outside the scope of the quantifier. Likewise the last occurrence of x in (2d). Notice furthermore that in (2d) the indefinite expression is translated by an existential quantifier; we will see that this nevertheless gives the right analysis. All three facts are vital for the possibility to arrive at the translation in a compositional manner, but we will defer the details for later. We shall assume now the following: the denotation of a formula ϕ in a model $\mathfrak{M}$ is a relation between assignments. We denote it by $[\![\phi]\!]_{\mathfrak{M}}$ or usually by $[\![\phi]\!]$. We usually write $g \xrightarrow{\phi} h$ rather than $\langle g, h\rangle \in [\![\phi]\!]$. (Actually, to make the dependency on the model $\mathfrak{M}$ explicit we would have to write $g \xrightarrow{\phi}_{\mathfrak{M}} h$, but we refrain from overly pedantic notation.) $g \xrightarrow{\phi} \surd$ means that there exists a k such that $g \xrightarrow{\phi} k$. If $g \xrightarrow{\phi} \surd$ we say that ϕ is *processable* in g.

Definition 3 (Dynamic Meaning) *Given a model $\mathfrak{M}$, the dynamic meaning of ϕ in $\mathfrak{M}$ is computed as follows.*

$$
\begin{array}{lcl}
[\![R(x_1, \dots, x_n)]\!] & := & \{\langle g, g\rangle : \langle \mathfrak{M}, g\rangle \models R(x_1, \dots, x_n)\} \\
[\![\top]\!] & := & \{\langle g, g\rangle : g \in \mathbb{V}(D)\} \\
[\![\bot]\!] & := & \varnothing \\
[\![\phi \wedge \psi]\!] & := & \{\langle g, h\rangle : \textit{for some } k : g \xrightarrow{\phi} k \xrightarrow{\psi} h\} \\
[\![\phi \vee \psi]\!] & := & \{\langle g, g\rangle : g \xrightarrow{\phi} \surd \textit{ or } g \xrightarrow{\psi} \surd\} \\
[\![\phi \to \psi]\!] & := & \{\langle g, g\rangle : \textit{for all } h : \textit{if } g \xrightarrow{\phi} h \textit{ then } h \xrightarrow{\psi} \surd\} \\
[\![\neg\phi]\!] & := & \{\langle g, g\rangle : \textit{not} : g \xrightarrow{\phi} \surd\} \\
[\![(\exists x)\phi]\!] & := & \{\langle g, h\rangle : \textit{exists } k \sim_x g : k \xrightarrow{\phi} h\} \\
[\![(\forall x)\phi]\!] & := & \{\langle g, g\rangle : \textit{for all } h \sim_x g : h \xrightarrow{\phi} \surd\}
\end{array}
$$

In Figure 1 further below we show the dynamic meaning of two formulae, namely $Q(x, y) \wedge (\exists x)(\exists y)(x = y)$ and $Q(x, y)$. We assume $Var = \{x, y\}$, $D = \{a, b\}$ and $I(Q) = \{\langle a, a\rangle, \langle a, b\rangle, \langle b, b\rangle\}$. The assignments are listed as pairs $\langle g(x), g(y)\rangle$.

Definition 4 *Let ϕ and ψ be two formulae. ϕ is called a* ***tautology*** *if $[\![\phi]\!] = [\![\top]\!]$, and a* ***contradiction*** *if $[\![\phi]\!] = \bot$. We say that ϕ and ψ are* ***equivalent*** *and write $\phi \approx \psi$ if $[\![\phi]\!] = [\![\psi]\!]$.*

Notice that $[\![\phi]\!] = [\![\psi]\!]$ means that for all models $\mathfrak{M}$, $[\![\phi]\!]_{\mathfrak{M}} = [\![\psi]\!]_{\mathfrak{M}}$, which in turn means that for all $\mathfrak{M}$ and all assignments g and h, $g \xrightarrow{\phi} h$ iff $g \xrightarrow{\psi} h$.

Definition 5 *Given a model $\mathfrak{M}$ and a formula ϕ, ϕ is* ***true*** *under the assignment g iff ϕ is processable in g iff there is a k such that $\langle g, k\rangle \in [\![\phi]\!]_{\mathfrak{M}}$. The set of all g such that ϕ is true under g is denoted by $\backslash\phi\backslash_{\mathfrak{M}}$. Dually,*

$$/\phi/_{\mathfrak{M}} := \{g : \textit{exists } k : k \xrightarrow{\phi} g\}$$

$\backslash\phi\backslash_{\mathfrak{M}}$ is also called the *satisfaction set* in [1]. We skip the motivation for these definitions and turn directly to our examples. (1d) is true in a model under an assignment g iff there is are assignments h and k such that $g \overset{(1d.1)}{\to} h \overset{(1d.2)}{\to} k$.

(1d.1) $(\exists x)(\mathsf{fat\text{–}man}(x) \wedge \mathsf{at\text{–}metro\text{–}entrance}(x))$

(1d.2) $\mathsf{sell\text{–}souvenirs}(x)$

Now $g \overset{(1d.1)}{\to} h$ iff for some m differing in at most x, the pair $\langle m, m\rangle$ is in the interpretation of $\mathsf{fat\text{–}man}(x) \wedge \mathsf{at\text{–}metro\text{–}entrance}(x)$ and $m = h$. This is the case simply when $h = m$ and $m(x)$ is a fat man in the metro entrance in $\mathfrak{M}$. So, $g \overset{(1d.1)}{\to} h$ iff h differs from g in at most x and $h(x)$ is a fat man in the metro entrance. $h \overset{(1d.2)}{\to} k$ iff $k = h$ and $h(x)$ is selling souvenirs. This is as desired.

Now take (2d). The relation $[\![(2d)]\!]$ is exactly the set of all $\langle g, g\rangle$ such that (2d) is true under g. (2d) is true under the assignment g if for every h such that $g \overset{(2d.1)}{\to} h$ there is a k such that $h \overset{(2d.2)}{\to} k$.

(2d.1) $(\exists x)(\mathsf{car}(x) \wedge \mathsf{own}(a, x))$

(2d.2) $\mathsf{wash\text{–}every\text{–}week}(a, x)$

Now, $g \overset{(2d.1)}{\to} h$ iff h differs from g at most in x and $h(x)$ is a car that Alfred owns. $h \overset{(2d.2)}{\to} k$ iff $k = h$ and Alfred washes $h(x)$ every week. Thus, $g \overset{(2d)}{\to} g$ iff for every x which is a car that Alfred owns, Alfred washes x every week. This is as it should be.

So, the translations which we have given for the sentences turn out to be correct. Let us see now that we can assign these translations to sentences in a compositional way. We will highlight only the relevant details here. The assumption is that a phrase of the form *some NP* or *a(n) NP* is translated by $(\exists x)\phi$ where (a) x is a fresh variable and (b) ϕ is the translation of *NP*. The condition on freshness of the variable is problematic (see Section 5), so we assume that the input for the translation algorithm is a sentence enriched with indices which tell us which variable to use. So the input for the translation are the sentences

(1′) *There is [a fat man]^1 at the metro entrance. He$^2_1$ is selling souvenirs.*

(2′) *If Alfred^1 owns [a car]^2 then he$^3_1$ washes it$^4_2$ every day.*

Here the superscripts are used for newly introduced referents and the subscripts for already existing referents. Pronouns have both a superscript and a subscript, so they pick up a previously introduced referent and introduce a new one. The translation now works exactly as in Montague Semantics. Let $(-)^\dagger$ be the translation from natural language into λ–expressions. Then the pronoun he^i_j is assigned the meaning $x_i = x_j$; likewise for the other pronouns *she*, *it*. Case endings are ignored. We give some sample translations for verbs, adjectives and nouns. (Here, $\mathcal{P}$ is a variable of type $\langle e, t\rangle$.)

$$\begin{array}{lcl}
(he^i_j)^\dagger & := & \lambda\mathcal{P}.\exists x_i.x_i = x_j \wedge \mathcal{P}(x_j) \\
(a(n)\ NP^i)^\dagger & := & \lambda\mathcal{P}.\exists x_i.NP^\dagger(x_i) \wedge \mathcal{P}(x_i) \\
(every\ NP^i)^\dagger & := & \lambda\mathcal{P}.\forall x_i.NP^\dagger(x_i) \to \mathcal{P}(x_i) \\
own^\dagger & := & \lambda x.\lambda y.\mathsf{own}(y, x) \\
sell\ souvenirs^\dagger & := & \lambda x.\mathsf{sell\text{–}souvenirs}(x) \\
fat^\dagger & := & \lambda\mathcal{P}.\lambda x.\mathsf{fat}(x) \wedge \mathcal{P}(x) \\
man^\dagger & := & \lambda x.\mathsf{man}(x) \\
(if\ S\ then\ T)^\dagger & := & S^\dagger \to T^\dagger \\
(S.\ T)^\dagger & := & S^\dagger \wedge T^\dagger
\end{array}$$

From these translations we derive the following formulae for $(1')$ and $(2')$:

(1e) $(\exists x_1)(\mathsf{fat}(x_1) \wedge \mathsf{man}(x_1) \wedge \mathsf{at\text{–}metro\text{–}entrance}(x_1))\wedge$
$(\exists x_2)(x_1 = x_2 \wedge \mathsf{sell\text{–}souvenirs}(x_2))$

(2e) $(\exists x_2)(\mathsf{car}(x_2) \wedge \mathsf{own}(a, x_2)) \to$
$(\exists x_3)(x_3 = a \wedge (\exists x_4)(x_4 = x_2 \wedge \mathsf{wash\text{–}every\text{–}week}(a, x_4)))$

This translation is somewhat more detailed than the one we gave earlier. It is easily checked that the satisfaction sets of the translations are the same as before. Therefore, with respect to the truth conditions we have succeeded in giving a compositional semantics. However, the situation is nevertheless not entirely ideal. We repeat (1d) with x_1 replacing x.

(1d) $(\exists x_1)(\mathsf{fat\text{–}man}(x_1) \wedge \mathsf{at\text{–}metro\text{–}entrance}(x_1)) \wedge \mathsf{sell\text{–}souvenirs}(x_1)$

Although (1d) and (1e) are truth requivalent, they do not have the same meaning in terms of the relation. That is to say, we have $\backslash(1d)\backslash = \backslash(1e)\backslash$ but we do not have $[\![(1d)]\!] = [\![(1e)]\!]$. Likewise for (2d) and (2e). Suppose for the sake of the argument that the variable x in (1d) is the same variable as x_1 in (1e). [2] Suppose we have the assignment $g : x_1 \mapsto \text{John}, x_2 \mapsto \text{Paul}$. Suppose further that John and Paul are different and that John is fat and standing at the metro entrance selling souvenirs. Let $h : x_1 \mapsto \text{John}, x_2 \mapsto \text{John}$. Then it turns out that $g \overset{(1e)}{\to} h$ but that $g \overset{(1d)}{\not\to} h$. The reason is easily identified: (1e) allows to change the value of x_1 and x_2 while (1d) allows to change only x_1. It may be thought that this is an effect of the translation of variables as existentially quantified expressions

[2] In fact, if they were not identical, matters would be quite the same.

rather than simply variables. However, this is not so. Consider for simplicity the two formulae $(\exists x_1)\phi(x_1)$ and $(\exists x_2)\phi(x_2)$. (Think of ϕ as the translation of (1d).)

Proposition 6 *Let ϕ be an expression with only one variable. Then for all models $\mathfrak{M}$:*

$$\backslash(\exists x_1)\phi(x_1)\backslash_{\mathfrak{M}} = \backslash(\exists x_2)\phi(x_2)\backslash_{\mathfrak{M}}$$

However in general

$$[\![(\exists x_1)\phi(x_1)]\!]_{\mathfrak{M}} \neq [\![(\exists x_2)\phi(x_2)]\!]_{\mathfrak{M}}$$

Proof. Put $\chi_1 := (\exists x_1)\phi(x_1)$ and $\chi_2 := (\exists x_2)\phi(x_2)$. Assume $g \xrightarrow{\chi_1} \surd$. Then there exists a h such that $h \sim_{x_1} g$ and $h(x_1)$ satisfies ϕ in $\mathfrak{M}$. Put $k(x_2) := h(x_1)$. Then $k \sim_{x_2} g$ and $k(x_2)$ satisfies ϕ in $\mathfrak{M}$. So, $g \xrightarrow{\chi_2} k$. This shows one inclusion. The other is similar. For the other assertion, let $\mathfrak{M}$ have at least two elements, a and b. Let $g : x_1 \mapsto a, x_2 \mapsto b$. Assume that a satisfies ϕ but not b. Then $g \xrightarrow{\chi_2} g$ but $g \overset{\chi_1}{\nrightarrow} g$. Q. E. D.

This is a pervasive feature of the dynamic interpretation and is a direct consequence of what the dynamic semantics sought to achieve. The meaning of an expression not only encodes its truth conditions but also its context behaviour. The previous theorem is a direct consequence of this fact: while the two formulae are true in the same models, they are nevertheless not substitutable in all contexts without changing the truth. Here is a concrete example. Suppose that Peter is watching John but not Albert. John is on the balcony. The valuation g is such that $g(x_2) =$ Albert. Now under g (4) turns out to be false, while (5) is true.

(4) *Someone^1 is standing on the balcony. Peter watches him$_2$.*

(5) *Someone^2 is standing on the balcony. Peter watches him$_2$.*

Here, the first sentence of these examples is translated by

$$(\exists x_{1/2})(\mathsf{stand\text{–}on\text{–}balcony}(x_{1/2}))$$

while the second is translated by

$$\mathsf{watch}(p, x_2)$$

Indeed, in (4) the existential quantifier sets x_1 to John. Everything else remains the same. Therefore, under this new valuation x_2 is still set to Albert. But Peter does not watch Albert. Therefore, (4) is false. In (5), however, the existential quantifier allows to change g to h, where $h(x_2) =$ John. Peter watches $h(x_2)$, which is John. So, (5) is true.

We still owe the reader a definition of the *logic* corresponding to this new interpretation. [3] Matters are a little bit difficult here. First of all, we will not define a relation between sets of formulae and a formula but between sequences of formulae and a single formula.

Definition 7 (Dynamic Consequence) *Let ϕ_i, $1 \leq i \leq n$, and ψ be formulae. Then $\phi_1; \phi_2; \ldots; \phi_n \models^d \psi$ if for all $\langle \mathfrak{M}, g \rangle$:*

$$g \xrightarrow{\phi_1} g_1 \xrightarrow{\phi_2} g_2 \ldots \xrightarrow{\phi_n} g_n \text{ implies } g_n \xrightarrow{\psi} \surd$$

This relation is called the ***dynamic consequence relation****.*

The reader may verify that the dynamic consequence relation satisfies very few of the postulates for ordinary consequence relations. For example, if Φ is a sequence and ϕ occurs in Φ then we ordinarily have $\Phi \models \phi$; but this fails for the dynamic consequence relation. Also, the sequence $\phi_1; \phi_2$ and the sequence $\phi_2; \phi_1$ have different dynamic consequences. These facts easily fall out of the results of the next section once some elementary facts are noted.

Proposition 8 (Deduction Theorem)

$$\phi_1; \phi_2; \ldots; \phi_n \models^d \psi \qquad \textit{iff} \qquad \phi_1; \phi_2; \ldots; \phi_{n-1} \models^d \phi_n \to \psi$$

Proposition 9

$$\phi_1; \ldots; \phi_i; \phi_{i+1}; \ldots; \phi_n \models^d \psi \qquad \textit{iff} \qquad \phi_1; \ldots; \phi_i \wedge \phi_{i+1}; \ldots; \phi_n \models^d \psi$$

The last theorem allows to reduce the dynamic consequence relation to a relation between formulae. (In view of the deduction theorem we can even reduce this to a set of theorems.) Two other relations between formulae come to mind, namely the static entailment and the meaning inclusion.

Definition 10 *Let ϕ and ψ be two formulae. ϕ* ***statically implies*** *ψ if $\backslash\phi\backslash \subseteq \backslash\psi\backslash$. ψ is* ***meaning included*** *in ϕ if $[\![\phi]\!] \subseteq [\![\psi]\!]$.*

It follows that $\phi \approx \psi$ iff ϕ is meaning included in ψ and ψ is meaning included in ϕ. Notice that a dynamic tautology is a formula ϕ such that $\models^d \phi$. In other words, ϕ is a tautology iff it is processable in every assignment iff $\backslash\phi\backslash = \mathbb{V}(D)$ iff ϕ is statically implied by $\top$. However, this is *not* equivalent with $[\![\phi]\!] \supseteq [\![\top]\!]$. Here we see once again that the dynamic notion of truth is quite counterintuitive and should not be seen as superseding the static notion of truth.

[3] This section is not essential for understanding of this paper and may be skipped.

3 Taking A Closer Look

In order to have a better grip on the mechanism of DPL we will prove some theorems about it and illustrate its relationship with static predicate logic. First, however, we will introduce some simplification according to Albert Visser. Namely, we will change the syntax of the existential quantifier in the following way. If x is a variable then $\exists x$ is an expression. Furthermore, we let

$$[\![\exists x]\!] := \{\langle g, h\rangle : g \sim_x h\}$$

Then note the following:

$$\begin{array}{rcl} [\![\exists x \wedge \phi]\!] & = & \{\langle g, h\rangle : \textit{exists } k : g \stackrel{\exists x}{\to} k \stackrel{\phi}{\to} h\} \\ & = & \{\langle g, h\rangle : \textit{exists } k : g \sim_x k \textit{ and } k \stackrel{\phi}{\to} h\} \end{array}$$

This is exactly the semantics of $(\exists x)\phi(x)$. This rather strange change in the syntax can actually be rather nicely motivated from the ideology of dynamic semantics. In the dynamic setting a formula can also be seen as a program to change the state of the hearer. The statement $g \stackrel{\phi}{\to} h$ means that the hearer may change from g into h upon being told that ϕ. We may also think of a formula as denoting a nondeterministic program, exactly as in Dynamic Logic. [4] So, in this view, the formula $\phi \wedge \psi$ is the consecutive execution of the programs ϕ and ψ; it allows to change first via ϕ and next via ψ. Consequently, we may interpret $\exists x$ as a *random reset* of x. Under this interpretation, $\exists x \wedge \phi$ is the instruction to first reset x randomly and then to execute ϕ.

Definition 11 (Visser Style Syntax) *Let ϕ be a formula of dynamic predicate logic. Then its translation, $\phi^\S$, is defined by*

$$\begin{array}{lcl} R(x_1, \ldots, x_n)^\S & := & R(x_1, \ldots, x_n) \\ \top^\S & := & \top \\ \bot^\S & := & \bot \\ (\neg\phi)^\S & := & \neg\phi^\S \\ (\phi \wedge \chi)^\S & := & \phi^\S \wedge \chi^\S \\ (\phi \to \chi)^\S & := & \phi^\S \to \chi^\S \\ (\phi \vee \chi)^\S & := & \chi^\S \vee \chi^\S \\ ((\exists x)\phi)^\S & := & \exists x \wedge \phi^\S \\ ((\forall x)\phi)^\S & := & \exists x \to \phi^\S \end{array}$$

We will in sequel prefer the Visser style syntax. Furthermore, we note that

$$\phi \wedge (\psi \wedge \chi) \approx (\phi \wedge \psi) \wedge \chi$$

[4] This connection has already been noted in [1].

Therefore — to save space — we will write '.' instead of $\wedge$ and drop brackets. So, $\exists x.\phi.\psi$ denotes either of $((\exists x)\phi) \wedge \psi$ or $(\exists x)(\phi \wedge \psi)$. Let us also note the following equivalences, which allow us to reduce the set of basic logical symbols rather drastically.

$$\begin{array}{lcl} (\forall x)\phi & \approx & \exists x \to \phi \\ \neg\phi & \approx & \phi \to \bot \\ \phi \vee \psi & \approx & \neg\phi \to \psi \end{array}$$

The first equivalence has already been used implicitly to define the translation of the universal quantifier. So, all connectives can be defined from $\exists$, $\bot$, $\to$ and $\wedge$. The logic of DPL is rather unusual otherwise. Various theorems of static logic fail to hold. For example, $\wedge$ is not commutative and not idempotent. That is to say, we neither have $\phi.\psi \approx \psi.\phi$ nor $\phi.\phi \approx \phi$ for all ϕ and ψ. Here are some counterexamples.

$$\begin{array}{lcl} \exists x.P(x).\exists x.\neg P(x) & \not\approx & \exists x.\neg P(x).\exists x.P(x) \\ P(x).\exists x.\neg P(x).P(x).\exists x.\neg P(x) & \not\approx & P(x).\exists x.\neg P(x) \end{array}$$

In fact, consider the set $D = \{a, b\}$ and let P be true of a but not of b. Put $g(x) := a$ and $h(x) := b$, $g(y) = h(y)$ for all $y \neq x$. We have

$$h \stackrel{\exists x}{\to} g \stackrel{P(x)}{\to} g \stackrel{\exists x}{\to} h \stackrel{\neg P(x)}{\to} h$$

and

$$g \stackrel{\exists x}{\to} h \stackrel{\neg P(x)}{\to} h \stackrel{\exists x}{\to} g \stackrel{P(x)}{\to} g$$

On the other hand, $\langle g, g\rangle \notin [\![\exists x.P(x).\exists x.\neg P(x)]\!]$ and $\langle h, h\rangle \notin [\![\exists x.\neg P(x).\exists x.P(x)]\!]$. The reason is simply that otherwise in the first case we must have $\langle g, k\rangle \in [\![\exists x.P(x).\exists x]\!]$ for some k and $\langle k, g\rangle \in [\![\neg P(x)]\!]$. But since $g(x) = a$, which does not satisfy P, the latter cannot hold. Turning now to the second example notice that the lower left hand side is a contradiction. For $P(x).\neg P(x)$ is a contradiction and — as the reader may check — if ϕ is a contradiction, so is $\chi.\phi$ and $\phi.\chi$. But we have

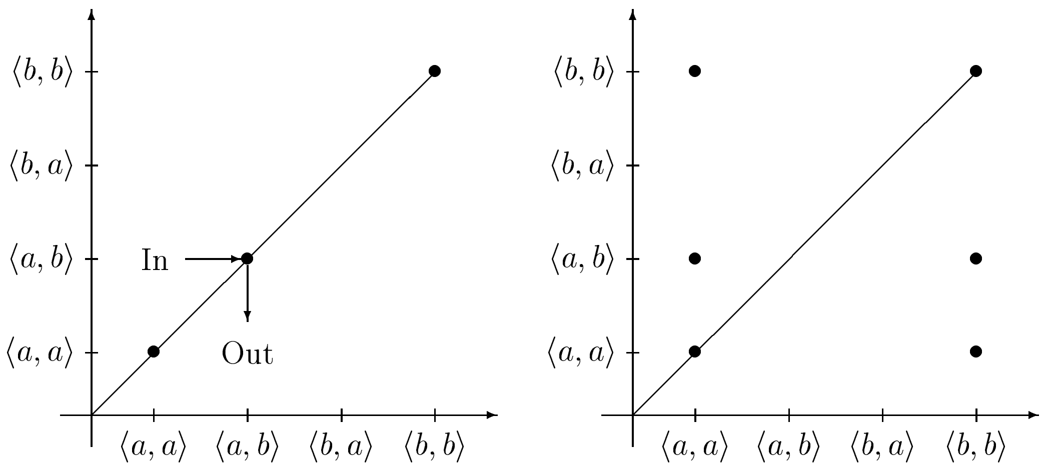
$$g \stackrel{P(x)}{\to} g \stackrel{\exists x}{\to} h \stackrel{\neg P(x)}{\to} h$$

An important characteristic of some connectives is the ability to change the valuation. For example, we can have $g \stackrel{\phi}{\to} h$ for some $h \neq g$; take $\phi = \exists x_1.\mathsf{car}(x_1)$. Think of the formula ϕ in this context as picking up g and returning h. (Note that due to the relational character, a formula may return many valuations.) If ϕ is a complex formula, then this scheduling of picking up valuations and returning other valuations can in fact become rather complex. For notice that not all formulae return a different valuation; one example is $\neg\phi$. Therefore, the following definition is introduced.

Definition 12 *A formula ϕ is a* ***test*** *if for all models $\mathfrak{M}$, $g \stackrel{\phi}{\to} h$ implies $g = h$.*

Figure 1: Dynamic Meaning: $D = \{a, b\}$, $I(Q) = \{\langle a, a\rangle, \langle a, b\rangle, \langle b, b\rangle\}$ and $Var = \{x, y\}$.

$[\![Q(x, y)]\!]$ $\qquad\qquad$ $[\![Q(x, y).\exists x.\exists y.x = y]\!]$



The *diagonal* is the set $\{\langle g, g\rangle : g \in \mathbb{V}(D)\} = [\![\top]\!]$. A formula is a test iff its meaning is a subset of the diagonal. The diagonal is inserted in both pictures of Figure 1. Tests have a blob only along the diagonal. So, $Q(x, y)$ is a test but $Q(x, y).\exists x.\exists y.x = y$ is not. Tests behave in much the same way as their static companion. To see this, note first of all the following.

Proposition 13 *If $\phi \approx \psi$ then $\backslash\phi\backslash = \backslash\psi\backslash$ and $/\phi/ = /\psi/$. However, $\backslash\top\backslash = \backslash\exists x\backslash$ and $/\top/ = /\exists x/$ but $[\![\top]\!] \neq [\![\exists x]\!]$.*

(For those who like to see a proper DPL example, the formula $(\exists x)\top$ does the trick instead of $\exists x$.)

Proposition 14 *For tests ϕ and ψ the following holds:*

$$\backslash\phi\backslash = \backslash\psi\backslash \quad \textit{iff} \quad /\phi/ = /\psi/ \quad \textit{iff} \quad [\![\phi]\!] = [\![\psi]\!]$$

For a proof note that

$$[\![\phi]\!] = \{\langle g, g\rangle : g \in \backslash\phi\backslash\} = \{\langle g, g\rangle : g \in /\phi/\}$$

All formulae of the following kind are tests:

1. Atomic formulae,

2. $\neg\phi$, $\phi \to \psi$ and $\phi \vee \psi$,

Table 1: Dynamic Properties of Connectives

	External	Internal
$\wedge$	$+$	$+$
$\vee$	$-$	$-$
$\rightarrow$	$-$	$+$
$\neg$	$-$	
$\exists$	$+$	
$\forall$	$-$	

3. $\phi.\psi$, given that ϕ and ψ are tests.

It follows that $(\forall x)\phi$ is a test, since it is of the form $\exists x \rightarrow \phi$. We may now note the following.

$$\begin{array}{lcl}
\backslash\neg\phi\backslash & = & \mathbb{V}(D) - \backslash\phi\backslash \\
\backslash\phi \vee \psi\backslash & = & \backslash\phi\backslash \cup \backslash\psi\backslash \\
\backslash\phi \rightarrow \psi\backslash & = & (\mathbb{V}(D) - \backslash\phi\backslash) \cup \backslash\psi\backslash \\
\backslash(\forall x)\phi\backslash & = & \{g : \mathit{for\ all}\ h \sim_x g : h \in \backslash\phi\backslash\}
\end{array}$$

These are exactly the clauses of static predicate logic if we read $\backslash\phi\backslash$ simply as $[\phi]$. Furthermore, if ϕ and ψ are tests then

$$\backslash\phi.\psi\backslash = \backslash\phi\backslash \cap \backslash\psi\backslash$$

This notion of a test can be further refined. A change in valuation can be either internal or external to the formula. In the formula $\phi.\psi$, ϕ can reset the valuation, for example if $\phi = \exists x_1.\chi$. The new value is then passed on to ψ. Because of this behaviour we call $\wedge$ *internally dynamic*. Furthermore, ψ may also change the valuation, and this latter change persists. That is to say, a formula to the right of $\phi.\psi$ picks up the valuation from ψ, so to speak. Therefore we call $\phi \wedge \psi$ *externally dynamic*. By contrast, $\phi \vee \psi$ is not internally dynamic (and so we say it is *internally static*). For whatever change ϕ may produce, ψ cannot pick up that new valuation; rather, it is evaluated against the same valuation as is ϕ. $\phi \vee \psi$ is also not externally dynamic: a formula at the right end of $\phi \vee \psi$ also starts at the same valuation as did $\phi \vee \psi$. Hence, $\phi \vee \psi$ is externally static. Externally static formulae are exactly the tests. We can summarize the behaviour in Table 1. Of course, the concept of internal dynamicity does not apply to unary connectives. If ϕ is a formula, then $\neg\neg\phi$ is a test. Moreover, we calculate that

$$[\![\neg\neg\phi]\!] = \{\langle g, g\rangle : g \xrightarrow{\phi} \surd\} = \{\langle g, g\rangle : g \in \backslash\phi\backslash\}$$

Namely, $[\![\neg\phi]\!] = \{\langle g, g\rangle : g \overset{\phi}{\nrightarrow} \surd\}$, that is, the set of pairs $\langle g, g\rangle$ such that ϕ is not processable in g. Hence, $[\![\neg\neg\phi]\!]$ is the set of all pairs such that $\neg\phi$ is not processable in g, which is the set of all $\langle g, g\rangle$ such that ϕ is processable in g.

Proposition 15 *Let ϕ be a formula. Then $\neg\neg\phi$ is a test and $\backslash\neg\neg\phi\backslash = \backslash\phi\backslash$.*

We call $\neg\neg\phi$ the *static counterpart* of ϕ. In this way we can define new connectives which are externally static, for example an externally static conjunction ($\neg\neg(\phi.\psi)$) and an externally static existential quantifier ($\neg\neg(\exists x.\phi)$). Likewise, we can remove the internal dynamicity of a connective. For example, the following connectives are internally static.

$$\begin{array}{lcl} \phi \mathbin{\dot\wedge} \phi & := & \neg\neg\phi.\psi \\ \phi \mathbin{\dot\to} \phi & := & \neg\neg\phi \to \psi \end{array}$$

And thirdly, the internal and external dynamicity can be cancelled together. In this way, we get totally static connectives. The static conjunction and the static implication are as follows.

$$\begin{array}{lcl} \phi \wedge^s \psi & := & \neg(\neg\phi \vee \neg\psi) \\ \phi \to^s \psi & := & \neg\phi \vee \psi \end{array}$$

It is however impossible to introduce a dynamicity into a connective. For example, the connective $\neg\phi \to \psi$ is an internally and externally static disjunction (equivalent to $\phi \vee \psi$). So there is no way to dynamify essentially static connectives. In a sense, this has to be expected. It is not clear a priori what for example an externally dynamic implication should be like. In fact, implication, disjunction and universal quantifiers are not externally dynamic. We give examples.

(6) *If someone^1 is watching you, then you must be careful. *He$_1$ is from the mafia.*

(7) *Either I am stupid or someone^1 is watching me. *He$_1$ is from the mafia.*

(8) *Be careful with [everyone who watches you]^1. *He$_1$ is from the mafia.*

Notice that if the second sentence is in subjunctive mood (e. g. *He might be from the mafia.*) then the continuation is generally acceptable (see [8]). This means that the second sentence is attached not on the main level of discourse but rather added into (generally) the second subformula. This phenomenon is called *subordination*. However, the subordinated material is not added as a mere conjunction, so subordination cannot be incorporated into the present semantics. Here are the (somewhat liberal) renderings of the sentences into predicate logic, showing the unacceptability of the pronoun in the second sentence.

(6a) $(\exists x_1.\mathsf{watch}(x_1, \mathsf{you}) \to \Box\mathsf{beware\text{–}of}(\mathsf{you}, x_1)).\mathsf{mafioso}(x_1)$

(7a) $(\mathsf{stupid}(\mathsf{me}) \vee \exists x_1.\mathsf{watch}(x_1, \mathsf{me})).\mathsf{mafioso}(x_1)$

(8a) $(\forall x_1)(\mathsf{watch}(x_1, \mathsf{you}) \to \Box\mathsf{beware\text{–}of}(\mathsf{you}, x_1)).\mathsf{mafioso}(x_1)$

It has however been noted that disjunction can behave internally dynamic.

(9) *Either Albert has not written any letter*[1] *or it*$_1$ *has been delayed.*

(9a) $\neg(\exists x_1.\mathsf{wrote}(a, x_1)) \vee \mathsf{delayed}(x_1)$

Formally, the translation, being of the form $\neg\phi \vee \psi$ does not allow to export the value of ϕ to ψ. Hence, in dynamic logic the anaphoric reference is blocked. Notice that by the laws of classical logic $\neg\phi \vee \psi$ is equivalent to $\phi \to \psi$, so in this special circumstance we may resort to the translation $\phi \to \psi$ in place of $\neg\phi \vee \psi$. In the latter the sharing of a referent between ϕ and ψ is legitimate and also possible in dynamic logic. There are however problems with compositionality. We may alternatively define a dynamic disjunction as follows.

$$[\![\phi \vee^d \psi]\!] := \{\langle g, g\rangle : g \xrightarrow{\phi} \surd \ \mathit{or}\ g \overset{\phi}{\not\to} h \xrightarrow{\psi} \surd\}$$

Notice that $\vee^d$ is a new connective, not definable from the previous ones.

4 Dynamic Binding and Scope

Recall from predicate logic the notion of the *scope* of a quantifier. Scope is a structural notion designed to capture the domain within which occurrences of the same variable invariably are interpreted as the same object in the model. We will define again the scope of a quantifier and then proceed to the extended binding domains of the dynamic quantifiers. We define the notion of a subformula in the usual way. ψ is a subformula of ϕ if $\phi = \neg\chi$ and ψ is a subformula of χ, or $\phi = \chi_1 \vee \chi_2$ or $\phi = \chi_1 \to \chi_2$ or $\phi = \chi_1 \wedge \chi_2$ and ψ is a subformula of χ_1 or of χ_2, or $\phi = (\exists x)\chi$ or $\phi = (\forall x)\chi$ and ψ is a subformula of χ. In Visser style syntax, $\exists x$ is a formula and so the clauses for the quantifiers can be dropped.

Definition 16 *Let $Q \in \{\forall, \exists\}$. Let ϕ be a formula and $(Qx)\chi$, ζ be subformulae of ϕ. ζ is said to occur in the* ***scope*** *of Qx in ϕ iff it occurs as a subformula of χ. Qx* ***binds*** *a variable x iff it is the quantifier with smallest scope containing x. A* ***static binding pair*** *of ϕ is a pair of occurrences of Qx and x, where Qx binds x.*

The dynamic notions are somewhat more roundabout. The definition of scope remains the same, except for the Visser style syntax, in which it is obsolete.

Definition 17 *Let ϕ be a formula. The dynamic accessibility relation of ϕ is a relation between occurrences of subformulae of ϕ, and it is defined as follows. χ is **dynamically accessible for** ξ (in ϕ) if*

1. *$\chi \to \xi$ is a subformula of ϕ or*
2. *$\chi.\xi$ is a subformula of ϕ or*
3. *ξ occurs in μ and χ is dynamically accessible for μ or*
4. *$\chi.\chi'$ is a subformula of ϕ and $\chi.\chi'$ is dynamically accessible for ξ or*
5. *$\chi'.\chi$ is a subformula of ϕ and $\chi'.\chi$ is dynamically accessible for ξ.*

*If χ is dynamically accessible for ξ we also say that ξ is **dynamically accessible from** χ.*

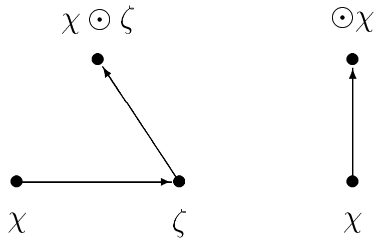
The dynamic accessibility relation is used mainly with respect to atomic subformulae so that the bracketing of a conjunction is mostly irrelevant. In a sequence $\phi_1.\phi_2.\ldots.\phi_n$, a formula ϕ_i is accessible for all subsequent ones, that is, for all ϕ_j with $j > i$. Further, in $\phi \to (\chi \to \psi)$, ϕ and χ are accessible for ψ, and ϕ is accessible for χ. The same applies to $(\phi.\chi) \to \psi$. By contrast, look at $(\phi \to \chi) \to \psi$. Here, ϕ is dynamically accessible for χ but not for ψ. Roughly speaking, the connective $\to$ in the antecedent $\phi \to \chi$ destroys the accessibility of ϕ and χ for other formulae.

Definition 18 *Let ϕ be a formula. A **dynamic binding pair** of ϕ is a pair of occurrences of Qx and x such that either (A) $Q = \forall$ and the pair is a static binding pair or (B) $Q = \exists$ and either (i) the least formula containing $\exists x$ also contains x or (ii) there is a ξ such that x occurs in ξ and ξ is dynamically accessible from $\exists x$. If $\exists x$ and x form a dynamic binding pair of ϕ we say that that $\exists x$ **dynamically binds** x.*

The following is an immediate consequence.

Proposition 19 *Let ϕ be a formula. Then if $\exists x$ and x form a static binding pair they also form a dynamic binding pair.*

Figure 2: Immediate accessibility



The reader may check the following fact. Suppose ϕ is a formula and $\phi^{\S}$ its translation into Visser style syntax. $\exists x$ and x form a dynamic binding pair of ϕ iff they form a dynamic binding pair of $\phi^{\S}$. This allows us to use both notations interchangeably when talking about binding. The dynamic accessibility relation can be defined purely in terms of the internal and external dynamicity of the connectives. This is what we will do now; it gives us a deeper understanding of these definitions and allows us to generalize them to formulae with other connectives. Let $\odot$ be a binary connective and $\chi \odot \zeta$ a formula. If $\odot$ is internally dynamic, we say that ζ is *immediately accessible* from χ. If $\odot$ is not internally dynamic, we say that ζ is *inaccessible* from χ. [5] If $\odot$ is externally dynamic we say that $\chi \odot \zeta$ is *immediately accessible* from ζ. If $\odot$ is not externally dynamic, $\chi \odot \zeta$ is *inaccessible* from ζ. Now let $\odot$ be a unary connective and χ a formula. Then $\odot\chi$ is a formula. If $\odot$ is externally dynamic, $\odot\chi$ is *immediately accessible* from χ; if $\odot$ is not externally dynamic, $\odot\chi$ is *inaccesible* from χ. The immediate accessibility relation might be pictured as in Figure 2 and Table 2. The internal dynamicity is 'horizontal', going in the direction of the time–arrow, the external dynamicity is 'vertical', going in the direction of the architecture of the formula. Now, say that ζ is accessible^1 from χ in ϕ if there exists a chain of (occurrences of) subformulae $\xi_1, \xi_2, \ldots, \xi_n$ such that $\xi_1 = \chi$ and $\xi_n = \zeta$, and ξ_{i+1} is immediately accessible from ξ_i for $1 \leq i < n$. It may happen that $n = 1$, in which case $\zeta = \chi$. Hence, accessibility^1 is the reflexive and transitive closure of immediate accessibility. Finally, we can give the following characterization.

Proposition 20 *Let ϕ be a formula, and ζ, χ subformulae of ϕ. ζ is dynamically accessible from χ if there exist a subformula $\zeta_1 \odot \chi_1$ of ϕ such that ζ is a subformula*

[5] To be accurate, we would have say that ζ is not immediately accessible from χ. However, it will turn out that under the definition of dynamic accessibility these two will turn to be same for the formula occurrences in question.

Table 2:

$\odot$	Immediate Accessibility
$\wedge$	$\{\langle \chi, \zeta \rangle, \langle \zeta, \chi \wedge \zeta \rangle\}$
$\to$	$\{\langle \chi, \zeta \rangle\}$
$\vee$	$\varnothing$
$\neg$	$\varnothing$
$\exists$	$\{\langle \zeta, (\exists x)\zeta\}$
$\forall$	$\varnothing$

of ζ_1, χ a subformula of χ_1, ζ_1 accessible[1] *from ζ, and $\odot$ is internally dynamic.*

This is again somewhat lengthy but quite a practical definition. Notice first that if ζ is accessible[1] from χ and does not contain χ (as a subformula) then it is accessible from χ. Hence another characterization is as follows. ζ is dynamically accessible from χ if (i) ζ does not contain χ, and (ii) ζ is a subformula of ζ_1 which is accessible[1] from χ.

We give an example. Let α be the formula

$$((\zeta \vee \eta).\neg(\theta.\phi)) \to (\chi \vee \psi)$$

The accessibility relation for α is as follows.

$$\begin{array}{lll}
\{\langle \zeta \vee \eta, \neg(\theta.\phi)\rangle, & \langle \zeta \vee \eta, \theta\rangle, & \langle \zeta \vee \eta, \phi\rangle, \\
\langle \zeta \vee \eta, \chi \vee \psi\rangle, & \langle \zeta \vee \eta, \chi\rangle, & \langle \zeta \vee \eta, \psi\rangle, \\
\langle \neg(\theta.\phi), \chi \vee \psi\rangle, & \langle \neg(\theta.\phi), \chi\rangle, & \langle \neg(\theta.\phi), \psi\rangle, \\
\langle (\zeta \vee \eta).\neg(\theta.\phi), \chi \vee \psi\rangle, & \langle (\zeta \vee \eta).\neg(\theta.\phi), \chi\rangle, & \langle (\zeta \vee \eta).\neg(\theta.\phi), \psi\rangle\}
\end{array}$$

5 Referent Systems

The advantage of DPL over static predicate logic is that it can handle the transsentential anaphors of the type exemplified in (1) and (2). However, as we have noted, one has to annotate the words in the sentences with indices in order to get a systematic (= compositional) translation from surface structure into DPL. The problem arises exactly as in Montague semantics with the pronouns and the quantifiers. Rather than having only one quantifier and only one pronoun we have infinitely many of them and we must be told beforehand (by means of annotation) which one to choose. This state of affairs is rather unsatisfactory because Montague semantics is otherwise successful in exploiting λ–calculus to get the variable management right. To see the effect of the λ–calculus, suppose we would say that the meaning of *man* is $\mathsf{man}(x)$ rather than $\lambda x.\mathsf{man}(x)$. Then every time we calculate the meaning of an expression containing the word *man* we have to check which variable we have to use in place of x. So we would have to decide whether to put in $\mathsf{man}(x)$ or $\mathsf{man}(y)$, for example. Likewise if we choose to translate *tall* by $\mathsf{tall}(x)$ rather than $\lambda \mathcal{P}.\lambda x.(\mathsf{tall}(x) \wedge \mathcal{P}(x))$. To take an easy example, the expression *tall man* would in principle be translated as $\mathsf{tall}(y) \wedge \mathsf{man}(x)$ rather than $\mathsf{tall}(x) \wedge \mathsf{man}(x)$ or $\mathsf{tall}(y) \wedge \mathsf{man}(y)$. To prevent this, we have to see to it that whatever variable we use to translate *tall* that same variable is used to translate *man*. In Montague semantics this problem does not arise by choice of the translation (and the λ–calculus, which renames variables automatically for us when needed). But while Montague semantics solves this problem elegantly, it nevertheless cannot solve the problem of quantifiers and pronouns as we have seen. DPL actually is a step back from Montague semantics insofar as it allows accidental capture of free variables and therefore cannot rely entirely on λ–calculus. (Notice, however, that this effect was intended, though not in all of its consequences, as Peter Staudacher has brought to my attention.) There is another problem of DPL, namely the accidental loss of variables. Suppose we have the following text (10a) and we translate accidentally by (10b) rather than by (10c).

(10a) There is a dog in the garden. There is a cat in the garden.

(10b) $\exists x.\mathsf{dog}(x).\mathsf{in–garden}(x).\exists x.\mathsf{cat}(x).\mathsf{in–garden}(x)$

(10c) $\exists x.\mathsf{dog}(x).\mathsf{in–garden}(x).\exists y.\mathsf{cat}(y).\mathsf{in–garden}(y)$

The truth conditions of (10b) and (10c) are in fact the same. However, in (10b) we have lost the possibility to refer back to the dog. Hence, the dynamic meanings of the two formulae are not identical. This is rather unfortunate. What can be done?

A solution to this circle of problems was outlined by Kees Vermeulen and Albert Visser in [13] and [14]. Since the second paper is rather advanced and technical we will concentrate on the first one, which introduces the so–called *referent systems*. Referent systems will solve the problem only partly but that will be enough for our purposes. Our solution is clearly intended by [13] and [14] though the actual details might differ. First of all, referent systems take a step back from Montague semantics in using no λ–expressions. The variable management that was left implicit in Montague semantics is now made fully explicit. So we will actually translate *tall* by $\mathsf{tall}(x)$ and *man* by $\mathsf{man}(x)$. The expression *tall man* will be translated by the merge of the two translations. The secret lies in the definition of the merge of representations. Basically, merge should be seen as conjunction; each lexical entry provides some information and these pieces of information are piled up. However, lexical items do also provide information about the syntactic structure, and this information is ancillary in finding the meaning of the sentence. For example, Montague semantics uses the syntactic structure to steer the semantic translation. By virtue of the fact that both words form a constituent and the adjective precedes the noun, the expression *tall man* is translated as

$$(\lambda \mathcal{P}.\lambda x.\mathsf{tall}(x) \wedge \mathcal{P}(x))(\lambda x.\mathsf{man}(x)) \rightsquigarrow \lambda x.\mathsf{tall}(x) \wedge \mathsf{man}(x)$$

The linear order is directly visible whereas the syntactic structure is not. While this is not such an apparent problem for English since constituents are as a rule continuous segments of speech (text), in other languages the situation is not so favourable. Take Latin. The following are acceptable sentences.

(11) ***Illustrem*** *habet Cicero* ***servum.***

(11′) *Cicero has* ***a famous slave.***

(12) ***Magno*** *fuerunt in* ***horto.***

(12′) *They were in* ***a big garden.***

In both cases, the Latin constituents highlighted by boldface type are not continuous segments in contrast to their English counterparts. This shows that constituency is defined by other criteria than simply contiguity. [6] To simplify the matter rather greatly, we may say that in Latin the *agreement suffixes* define the constituency. To implement this, we introduce the notion of a *referent system*.

Definition 21 *Let N be a set. A referent system over N is a triple $\mathfrak{R} = \langle I, R, E\rangle$, where R is a set, called the set of* ***referents****, I is a partial injective function from N to R, called the* ***import function****, and E a partial injective function from R to N, called the* ***export function****. N is the set of* ***names****.*

When $E(r) = A$ we say that $\mathfrak{R}$ *exports r under the name A*, and when $I(B) = r$ we say that $\mathfrak{R}$ *imports r under the name B*. It is not required that $A = B$! Meaning units are pairs $\mathfrak{E} = \langle \mathfrak{R}, \Phi\rangle$ where $\mathfrak{R} = \langle I, R, E\rangle$ is a referent system over some appropriate set of names and Φ a set of formulae using only the referents from R. Let $\langle D, I\rangle$ be a model. An *assignment* is a map from a set of referents

[6] Even English has discontinous constituents. For example in *He rang me up.* or in *A man talked to me who had an extraordinarily long beard.*

into D. $\langle \mathfrak{R}, \Phi \rangle$ is *satisfied* in a model under the assignment h if h assigns a value to each referent from $\mathfrak{R}$ and all formulae from Φ are true under h. Hence, the satisfaction clauses are pretty much those of DRT. The consequence is that the renaming of referents does not change satisfiability in a model.

Let $\mathfrak{E}_i = \langle \mathfrak{R}_i, \Phi_i \rangle$, $i \in \{1, 2\}$, be meaning units. The merge $\mathfrak{E}_1 \circ \mathfrak{E}_2$ is defined as follows. We define the merge $\mathfrak{R}_3 := \mathfrak{R}_1 \bullet \mathfrak{R}_2$ of referent systems plus injective functions $\iota_1 : R_1 \to R_3$, $\iota_2 : R_1 \to R_3$, and then put

$$\mathfrak{E}_1 \circ \mathfrak{E}_2 := \langle \mathfrak{R}_3, \iota_1[\Phi_1] \cup \iota_2[\Phi_2] \rangle$$

We are left with a definition of the merge. We say first of all given two referents $r \in R_1$ and $s \in R_2$ that r *supervenes* s if $I_2(E_1(r)) = s$. Supervenience is a relation $\subseteq R_1 \times R_2$. Let U be the set of supervened elements of R_2. Then we put

$$\begin{array}{lcl} R_3 & := & (R_1 \times \{1\} \cup R_2 \times \{2\}) - U \times \{2\} \\ \iota_1(r) & := & \langle r, 1 \rangle \\ \iota_2(s) & := & \left\{ \begin{array}{ll} \langle r, 1 \rangle & \text{if } s \text{ is supervened by } r \\ \langle s, 2 \rangle & \text{else} \end{array} \right. \end{array}$$

Say that r *I–preempts* s if there is an $A \in N$ such that $I_1(A) = r$ and $I_2(A) = s$; and that s *E–preempts* r if $E_2(s) = E_1(r)$. Notice that r can both I–preempt and supervene s. Finally, for a partial function f we write $f(x) = \uparrow$ if f is undefined on x, and $f(x) = \downarrow$ if f is defined. The import and export functions are now as follows.

$$I_3(A) \quad := \quad \left\{ \begin{array}{ll} \langle r, 1 \rangle & \text{if } I_1(A) = r \\ \langle s, 2 \rangle & \text{if } s \text{ is not I–preempted and } I_2(A) = s \\ \uparrow & \text{else} \end{array} \right.$$

$$E_3(u) \quad := \quad \left\{ \begin{array}{ll} E_2(s) & \text{if } u = \langle t, 1 \rangle \text{ and } t \text{ supervenes } s \\ E_2(s) & \text{if } u = \langle s, 2 \rangle \text{ and } s \text{ is not superverned} \\ E_1(r) & \text{if } u = \langle r, 1 \rangle, E_1(r) = \downarrow \text{ and is not E–preempted} \\ \uparrow & \text{else} \end{array} \right.$$

Some options can be summarized in Table 3. We write $[A : r : B]$ if r is a referent that is imported under A and exported under the name B. We write $[- : r : B]$ if r has no import name, and similarly $[A : r : -]$ and $[- : r : -]$. The table does not show the effect of the preemption. It can happen that two referents compete for the same import (export) name. In that case they must be from different referent systems (since both are functions and injective). Then the referent from the first system wins the import name; if they compete for the export name, the one from the second system wins the export name.

With the referent systems the Latin examples can be accounted for; as names we choose the cases, and the verbs and prepositions are simply referent systems

Table 3: The Merge of Referent Systems

$$\begin{array}{cccccc}
[A : r : B] & \bullet & [B : s : C] & = & [A : \langle r, 1\rangle : C] \\
[- : r : B] & \bullet & [B : s : C] & = & [- : \langle r, 1\rangle : C] \\
[A : r : B] & \bullet & [B : s : -] & = & [A : \langle r, 1\rangle : -] \\
[- : r : B] & \bullet & [B : s : -] & = & [- : \langle r, 1\rangle : -]
\end{array}$$

importing referents under certain names, while inflected nouns export referents under a given name. For example, Latin *horto* and *magno* are translated by

$$\begin{array}{|l|}\hline [- : r : \text{abl}] \\ \hline \mathsf{garden}(r) \\ \hline \end{array} \qquad \begin{array}{|l|}\hline [\text{abl} : r : \text{abl}] \\ \hline \mathsf{big}(r) \\ \hline \end{array}$$

Their merge is — according to the definition above —:

$$\begin{array}{|l|}\hline [- : \langle r, 1\rangle : \text{abl}] \\ \hline \mathsf{garden}(\langle r, 1\rangle) \\ \mathsf{big}(\langle r, 1\rangle) \\ \hline \end{array}$$

This is the translation of *horto magno*. Notice that *magno horto* would in this system not get the right translation. We need to assume in fact that lexical entries are associated with sets of referent systems, thereby allowing for different word order (and the fact that both *magno* and *horto* can also be dative). Obviously, this model is very simplistic, but it shows how one can incorporate morphological information about the syntactic structure into the semantics.

How would referent systems handle our examples (1) and (2)? We will present a solution, which is based on the following insight. Pronouns pick up their referent not by an index (such an index is simply not part of the language) but rather by the information that is resident in the semantics of the antecedent, the gender of the pronoun and more. To make matters simple, we assume that we only use gender information. Therefore the set of names consists of combinations of gender and case. However, one or both of gender and case can be absent, and this is represented by $\star$. Hence the set of names is as follows:

$$\{\text{M}, \text{F}, \text{N}, \star\} \times \{\text{N}, \text{A}, \star\}$$

(Here, N abbreviates *nominative*, A *accusative*, and F, M and N are the genders.) A particular pair is written like a vector, for example (M, A). Notice that $(\text{M}, \star)$ is a name in the technical sense, likewise $(\star, \star)$ and $(\star, \text{A})$. The cases will be used to steer the syntactic translation, and the gender is used to get at the binding. To make the whole thing work we have to play with the $\star$ to switch the assignment of gender and case on and off. For example, the pronouns he and him have the following semantics

$$\begin{array}{|l|}\hline [(\mathrm{M},\star) : x : (\mathrm{M},\mathrm{N})] \\ \hline \varnothing \\ \hline \end{array} \qquad \begin{array}{|l|}\hline [(\mathrm{M},\mathrm{A}) : x : (\mathrm{M},\star)] \\ \hline \varnothing \\ \hline \end{array}$$

The difference is that he has no case to the left and nominative to the right, while him has accusative to the left and no case to the right. The semantics of fat, man (nominative) and man (accusative) are as follows:

$$\begin{array}{|l|}\hline [(\alpha,\gamma) : x : (\alpha,\gamma)] \\ \hline \mathsf{fat}(x) \\ \hline \end{array} \qquad \begin{array}{|l|}\hline [(\mathrm{M},\star) : x : (\mathrm{M},\mathrm{N})] \\ \hline \mathsf{man}(x) \\ \hline \end{array} \qquad \begin{array}{|l|}\hline [(\mathrm{M},\mathrm{A}) : x : (\mathrm{M},\star)] \\ \hline \mathsf{man}(x) \\ \hline \end{array}$$

Here the variable is instantiated to any appropriate value (in this case, genders or $\star$ for α and case or $\star$ for γ). This is an extensions of the original proposal; what we argue is that the variables are part of the lexical representation and get instantiated after the representation has been inserted into the structure. The indefinite determiner a(n) wipes out the gender to the left. It has the semantics

$$\begin{array}{|l|}\hline [(\star,\gamma) : x : (\alpha,\gamma)] \\ \hline \varnothing \\ \hline \end{array}$$

Finally, the transitive verb see looks as follows

$$\begin{array}{|l|}\hline [(\alpha,\mathrm{N}) : x : (\alpha,\star)] \\ [(\beta,\star) : y : (\beta,\mathrm{A})] \\ \hline \mathsf{see}(x,y) \\ \hline \end{array}$$

We now take the sentence (1) in a slightly modified form. The bracketing (plus case assignment) is given by the syntax.

(13) Susan (sees (a fat man (at the metro entrance))). He (is selling souvenirs).

We continue our policy to leave the phrases ‘at the metro entrance’ and ‘is selling souvenirs’ unanalyzed. Putting together the object noun phrase and renaming the referents suitably gives

$$\begin{array}{|l|}\hline [(\star,\mathrm{A}) : x : (\mathrm{M},\star)] \\ \hline \mathsf{fat}(x), \mathsf{man}(x) \\ \mathsf{at\text{–}metro\text{–}entrance}(x) \\ \hline \end{array}$$

So the first sentence is translated thus:

$$\begin{array}{|l|}\hline [(\text{F},\star) : x : (\text{F},\text{N})] \\ \hline x = s \\ \hline \end{array} \circ \left(\begin{array}{|l|}\hline [(\alpha,\text{N}) : x : (\alpha,\star)] \\ [(\beta,\star) : y : (\beta,\text{A})] \\ \hline \mathsf{see}(x,y) \\ \hline \end{array} \circ \begin{array}{|l|}\hline [(\star,\text{A}) : x : (\text{M},\star)] \\ \hline \mathsf{fat}(x), \mathsf{man}(x) \\ \mathsf{at\text{–}metro\text{–}entrance}(x) \\ \hline \end{array} \right)$$

$$\simeq \begin{array}{|l|}\hline [(\text{F},\star) : x : (\text{F},\star)] \\ [(\text{M},\star) : y : (\text{M},\star)] \\ \hline x = s, \mathsf{fat}(y), \mathsf{man}(y) \\ \mathsf{at\text{–}metro\text{–}entrance}(y) \\ \mathsf{see}(x,y) \\ \hline \end{array}$$

Here, α is instantiated to F and β to M. $\simeq$ means that the structures are equal only after some renaming of referents. The point is now that the referents are open to anaphoric reference, but are syntactically inert since they carry the case description $\star$. Subsequently they can only be picked up by a pronoun. For example, we can merge with

$$\begin{array}{|l|}\hline [(\text{M},\star) : x : (\text{M},\star)] \\ \hline \mathsf{sell\text{–}souvenirs}(x) \\ \hline \end{array}$$

This gives

$$\begin{array}{|l|}\hline [(\text{F},\star) : x : (\text{F},\star)] \\ [(\text{M},\star) : y : (\text{M},\star)] \\ \hline x = s, \mathsf{fat}(y), \mathsf{man}(y) \\ \mathsf{at\text{–}metro\text{–}entrance}(y) \\ \mathsf{see}(x,y) \\ \mathsf{sell\text{–}souvenirs}(y) \\ \hline \end{array}$$

In order to be able to judge the success and failure of referent systems, compare the result with a slightly different sentence.

(14) Paul sees a fat man at the metro entrance. He is selling souvenirs.

What will happen is that the phrase 'a fat man at the metro entrance' will get a referent that is different from that for Paul. This is due to the fact that the determiner blocks the gender to the left. But the referent also E–preempts the referent for Paul since they export the same name before merge, $(\text{M}, \star)$. [7] The pronoun 'he' can therefore not refer back to Paul. Hence referent systems do not handle the facts correctly. In this case it is because the space of names is too small to make enough distinctions. However, the fact that anaphoric reference is blocked by antecedents which are less distant is not so far off the mark. This is a topic that deserves attention. The analogue of example (2) is less straightforward, since we have no means to represent the implication.

[7] Here is also a point where one has to be careful with the variables for names. We will not explore that theme further, though.

6 Outlook

This article is only an introduction into dynamic semantics. A survey of the various developments can be found in [7]. We will end with a few remarks about the relationship between dynamic predicate logic and DRT as well as other uses of DPL. First, with respect to DRT note that both DPL and DRT encode the linearity of the text into the notion of accessibility. In DRT as well as in DPL, in a formula $\phi \to \chi$, referents introduced in ϕ may be used in χ and not vice versa. In $\phi \vee \chi$, neither can χ access referents from ϕ (because $\vee$ is internally static) nor can ϕ access referents from χ (since connectives work from left to right). We have already mentioned the fact that disjunction may occasionally be internally dynamic (but left–to–right). We will comment on the left–to–right character of connectives below. DRT differs from DPL in the way in which conjunction and existential quantification is treated. The existential quantifier is not so much a problem. DRT employs an implicit quantifier, namely the head section of the box. Recall that a DRS looks like this

$x\ y$
$\phi(x)$ $\chi(x, y)$ $\zeta(y, z)$

Here, we may treat the variables of the upper section (x and y) as quantified existentially (by a dynamic existential quantifier). So, the DRS is translated by

$$\exists x.\exists y.\phi(x).\chi(x, y).\zeta(y, z)$$

Conversely, $\exists x$ of DPL is like putting x into the head section of the just created DRS. The biggest difference is however conjunction. DRT has no conjunction in the sense of the word, but we may for our purposes say that the joint occurrence of formulae in a DRS means in practice that they are occurring in a conjunction. Hence, we may read the DRS above also as

$x\ y$
$\phi(x) \wedge \chi(x, y) \wedge \zeta(y, z)$

If read in this way, DRT conjunction is fully commutative, in contrast to DPL conjunction. However, notice that in the present circumstances no difference arises. The reader may namely check the following.

Proposition 22 *Let χ_1 and χ_2 be atomic formulae. Then*

$$[\![\chi_1 . \chi_2]\!] = [\![\chi_2 . \chi_1]\!]$$

Therefore, when no quantifier intervenes in a conjunction, full commutativity holds in DPL as well, and so we may disregard the order between the conjuncts, as is done in DRT.

The left–to–right character of DRT and DPL is in many instances a problematic feature and is not observed as rigorously as one may think. Several cases may be noted. First, from a syntactic point of view, anaphors inside sentences disregard the order of elements. They are only sensitive to the syntactic structure. This at least is the claim in Government and Binding theory. We will not comment on the validity of the last claim (it is doubtfull as well) but simply note the following examples.

(15a) Albert^1 looks quite funny with his$_1$ hat.

(15b) With his$_1$ hat, Albert^1 looks quite funny.

(16a) Albert gave Pete^1 a photograph with his$_1$ family on it.

(16b) Albert gave a photograph with his$_1$ family on it to Pete^1.

(17a) Everybody^1 likes his$_1$ friends.

(17b) His$_1$ friends, everybody^1 likes.

In all examples, the pronoun can precede its referent. Moreover, it is known that texts and dialogues are structured and that pronouns may only refer to objects that are available at the right structural level. This structuring is not reflected in DPL. (See [2].)

Finally, it has often been noted that there is a close connection between anaphora and presupposition. Technically, the domains of accessibility turn out to be those that are used in the projection algorithm for presuppositions. For example, the (a) sentences are said to be free of presupposition because the sentence of the left implies the presupposition of the sentence to the right. By contrast, the (b) sentences contain a presupposition since the first sentence contains a presupposition.

(18a) The series $1 + 2^n$ is convergent. The limit of $1 + 2^n$ is 1.

(18b) The limit of the series $1 + 2^n$ is 1. The series $1 + 2^n$ is convergent.

(19a) If (a_n) is convergent then the limit of $(1 + a_n)$ is $1 + \lim(a_n)$.

(19b) If the limit of $(1 + a_n)$ is not $1 + \lim(a_n)$ then (a_n) is not convergent.

The first to notice this connection is Rob van der Sandt in [11]. He in facts uses DRT to give an integrated account for pressuposition and anaphora. Jan van Eijck (see among other [12]) actually tries to lift DPL to a three valued dynamic logic whereby replicating observations by Lauri Karttunen ([5]) and Irene Heim ([3]) in a dynamic setting. For a discussion about the use of three–valued logic and dynamics in this connection see [6].

References

[1] Jeroen Groenendijk and Martin Stokhof. Dynamic Predicate Logic. *Linguistics and Philosophy*, 14:39 – 100, 1991.

[2] B. Grosz and C. Sidner. Attention, intention and the structure of discourse. *Computational Linguistics*, 12:175 – 204, 1986.

[3] Irene Heim. On the projection problem for presuppositions. In M. Barlow and D. Flickinger, D. Westcoat, editors, *Proceedings of the 2nd West Coast Conference on Formal Linguistics*, pages 114 – 126, Stanford University, 1983.

[4] Hans Kamp and Uwe Reyle. *From Discourse to Logic, Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory.* Kluwer, Dordrecht, 1993.

[5] Lauri Karttunen. Presuppositions and linguistic context. *Theoretical Linguistics*, 1:181 – 194, 1974.

[6] Marcus Kracht. Logic and control: How they determine the behaviour of presuppositions. In Jan van Eijck and Albert Visser, editors, *Logic and Information Flow*, pages 88 – 111. MIT Press, Cambridge, Massachusetts, 1994.

[7] Reinhard Muyskens, Johan van Benthem, and Albert Visser. Dynamics. In Johan van Benthem and Alice ter Meulen, editors, *Handbook of Logic and Language*, pages 587 – 648. Elsevier, 1997.

[8] Craige Roberts. Modal Subordination and Pronominal Anaphora in Discourse. *Linguistics and Philosophy*, 12:683 – 723, 1989.

[9] Peter Staudacher. Zur Semantik indefiniter Nominalphrasen. In Brigitte Asbach-Schnitker and Johannes Roggenhofer, editors, *Neuere Forschungen zur Wortbildung und Historiographie der Linguistik. Festgabe für Herbert E. Brekle zum 50. Geburtstag*, pages 239 – 258. Gunter Narr Verlag, Tübingen, 1987.

[10] Peter Staudacher. PLA und Dynamic Predicate Logic (PDL). unpublished, 1996.

[11] Rob A. van der Sandt. Presupposition as anaphora resolution. *Journal of Semantics*, 9:333 – 377, 1992.

[12] Jan van Eijck. The dynamics of descriptions. *Journal of Semantics*, 10:239 – 267, 1993.

[13] Kees F. M. Vermeulen. Merging without Mystery or: Variables in Dynamic Semantics. *Journal of Philosophical Logic*, 24:405 – 450, 1995.

[14] Kees F. M. Vermeulen and Albert Visser. Dynamic bracketing and discourse representation. *Notre Dame Journal of Formal Logic*, 37:321 – 365, 1996.