The title of the paper

Philippe de Groote
INRIA

Ekaterina Lebedeva
INRIA

October 3, 2012

Abstract

This is the paper's abstract ...

1 Introduction

This is the paper's introduction.

2 The Underlying Dynamic Logics

Philippe de Groote [1] showed that it is possible to handle dynamic phenomena of natural language by standard tools of mathematical logic, such as simply-typed lambda calculus, and, therefore, stay within Montague's program. This is accomplished in de Groote's framework, called below G_0, by providing Montague semantics with a notion of context in a systematic and precise way.

The meaning of a sentence is a function of the context. It can be expressed in lambda calculus by defining the term standing for the interpretation of a sentence as an abstraction over a variable standing for the context.

Definition 2.1. [Context, Environment] A **context** or **environment** is a term of type γ that stores the essential information from what has already been processed in the computation of the meaning of the whole discourse.

In order to make the framework flexible, the context type γ is a parameter, which can define any complex type. Therefore, there is no restriction on the representation of context. One can define it as a simple structure focusing on a particular phenomenon and elaborate it as more complex phenomena are considered.

A sentence can have a potential to change (or update) the context. The updated context has to be passed as an argument to the meaning of the subsequent sentence. In order to do so compositionally, de Groote used the notion of continuation: the meaning of a sentence not only is a function of a context, but also is a function of a continuation with respect to the computation of the meaning of the whole discourse. Within the body of the term standing for the meaning of a sentence, the continuation is given the possibly updated context and returns a proposition. Therefore, the continuation has type $(\gamma \to o)$.

Definition 2.2. [Continuation] A **continuation** is a term of type $(\gamma \to o)$ that denotes what is still to be processed in the computation of the meaning of the whole discourse.

Thus, a sentence is dynamically interpreted as a function that takes a context e of type γ and a continuation ϕ of type $(\gamma \to o)$ and returns a proposition:

$$[\![s]\!] = \underbrace{\gamma}_{\text{context}} \to \underbrace{(\gamma \to o)}_{\text{continuation}} \to \underbrace{o}_{\text{proposition}}$$

Type $(\gamma \to (\gamma \to o) \to o)$ is, therefore, defined to be the type of a dynamic proposition.

Example 2.3. The meaning of the sentence (1) is the λ-term (1):

(1) *John loves Mary.*

$$\underbrace{\overbrace{\lambda \underbrace{e^{\gamma}}_{\text{context}} \underbrace{\phi^{\gamma\to o}}_{\text{continuation}} . \overbrace{\overbrace{\overbrace{\mathbf{love}^{\iota\to\iota\to o}\ \mathbf{j}^{\iota}}^{\iota\to o}\ \mathbf{m}^{\iota}}^{o} \wedge \overbrace{\phi e^{*}}^{o}}^{o}}^{\gamma\to(\gamma\to o)\to o}}_{\text{dynamic proposition}} \tag{1}$$

where e^* is the context obtained by updating e. □

Note the presence of the conjunct ϕe^* in (1) that conveys that an updated context is passed as an argument to the continuation of a proposition, and is, therefore, accessible in the rest of the computation. This kind of conjunct is a subterm of every proposition in G_0.

In G_0 each object is interpreted as a variable (i.e. as a term of type ι). The framework is presented on the phenomena of cross-sentential and donkey anaphora and the type of context γ is defined as a list of individuals for the sake of simplicity:

$$\gamma \doteq \texttt{list of } [\ \iota\ ] \tag{2}$$

Thus, the context stores only interpretations of objects that previously occurred in the discourse. When a new object is interpreted as an individual x, the current context e is updated with x, resulting in $(x :: e)$, where $::$ is a list constructor of type $(\iota \to \gamma \to \gamma)$ (i.e. it is a function that takes an individual and a context and returns an (updated) context).[1]

Therefore, returning to Example 2.3, e^* is $(\mathbf{m} :: \mathbf{j} :: e)$. Hence, the interpretation of Sentence (1) is as follows:

$$\lambda e\phi.\mathbf{love}\ \mathbf{j}\ \mathbf{m} \wedge \phi(\mathbf{m} :: \mathbf{j} :: e) \tag{3}$$

Term (3) has to be computed compositionally from lexical meanings $[\![John]\!]$, $[\![Mary]\!]$ and $[\![loves]\!]$. Particularly, it has to be the result of normalizing $[\![loves]\!][\![Mary]\!][\![John]\!]$.

A noun phrase in Montague semantics is a term taking a property as an argument and returning a proposition. In framework G_0 there should be two additional arguments for a term to return a proposition. Therefore, everywhere where a term of type o occurs in Montague's interpretation, there has to be a term of type $(\gamma \to (\gamma \to o) \to o)$ in de Groote's interpretation, as can be easily seen comparing (4a) and (4b), where Ω is an abbreviation for $(\gamma \to (\gamma \to o) \to o)$. Thus, a noun phrase is interpreted as a function of three arguments (a property, a context and a continuation) that returns a proposition, as can be more easily seen in (4c), where no abbreviation is

[1] Operation $::$ is right associative. For example, $(x :: y :: e)$ is equivalent to $(x :: (y :: e))$.

used:

$$\llbracket np \rrbracket =_{Montague} \underbrace{(\iota \to o)}_{\substack{\text{static}\\ \text{property}}} \to \underbrace{o}_{\substack{\text{static}\\ \text{proposition}}} \tag{4a}$$

$$\llbracket np \rrbracket =_{de\ Groote} \underbrace{(\iota \to \Omega)}_{\substack{\text{dynamic}\\ \text{property}}} \to \underbrace{\Omega}_{\substack{\text{dynamic}\\ \text{proposition}}} \tag{4b}$$

$$\llbracket np \rrbracket =_{de\ Groote} \underbrace{(\iota \to \gamma \to (\gamma \to o) \to o)}_{\substack{\text{dynamic}\\ \text{property}}} \to \underbrace{\underbrace{\gamma}_{\text{context}} \to \underbrace{(\gamma \to o)}_{\text{continuation}} \to \underbrace{o}_{\text{proposition}}}_{\substack{\text{dynamic}\\ \text{proposition}}} \tag{4c}$$

The interpretation of *Mary*, for example, is as follows:

$$\llbracket Mary \rrbracket = \lambda \overbrace{\underbrace{\mathbf{P}^{\iota\to\gamma\to(\gamma\to o)\to o}}_{\substack{\text{dynamic}\\ \text{property}}}.\,\lambda \underbrace{\underbrace{e^{\gamma}}_{\text{context}}\ \underbrace{\phi^{\gamma\to o}}_{\text{continuation}}\ .\ \overbrace{\overbrace{\overbrace{\mathbf{Pm}^{\iota}}^{\gamma\to(\gamma\to o)\to o}\ e}^{(\gamma\to o)\to o}\ \overbrace{(\lambda e'^{\gamma}.\,\overbrace{\phi(\overbrace{\mathbf{m} :: e'}^{\gamma})}^{o})}^{\gamma\to o}}^{o}}_{\substack{\text{dynamic}\\ \text{proposition}}}}^{(\iota\to\gamma\to(\gamma\to o)\to o)\to\gamma\to(\gamma\to o)\to o} \tag{5}$$

The interpretation of *John* is analogous:

$$\llbracket John \rrbracket = \lambda \mathbf{P}.\lambda e\phi.\mathbf{Pj}e(\lambda e'.\phi(\mathbf{j} :: e')) \tag{6}$$

A transitive verb is interpreted in Montague semantics as a term taking two type-raised individuals and returning a proposition. Since in de Groote's framework there has to be an abstraction over a context and a continuation to get a proposition, everywhere where a term of type o occurs in Montague's interpretation, there has to be a term of type $(\gamma \to (\gamma \to o) \to o)$ in de Groote's interpretation. This can be seen comparing types in (7):

$$\llbracket tv \rrbracket =_{Montague} (\underbrace{(\iota \to o)}_{\text{property}} \to \underbrace{o}_{\text{proposition}}) \to (\underbrace{(\iota \to o)}_{\text{property}} \to \underbrace{o}_{\text{proposition}}) \to \underbrace{o}_{\text{proposition}} \tag{7a}$$

$$\llbracket tv \rrbracket =_{de\ Groote} (\underbrace{(\iota \to \Omega)}_{\text{property}} \to \underbrace{\Omega}_{\text{proposition}}) \to (\underbrace{(\iota \to \Omega)}_{\text{property}} \to \underbrace{\Omega}_{\text{proposition}}) \to \underbrace{\Omega}_{\text{proposition}} \tag{7b}$$

Then the interpretation of *loves* is as follows:

$$[\![loves]\!] = \overbrace{\lambda \mathbf{Y}^{(\iota\to\Omega)\to\Omega}\mathbf{X}^{(\iota\to\Omega)\to\Omega}.\, \overbrace{\mathbf{X}(\overbrace{\lambda x.\, \overbrace{\mathbf{Y}(\overbrace{\lambda y.(\overbrace{\lambda e'^{\gamma}\phi^{\gamma\to o}.\, \overbrace{\overbrace{\overbrace{\mathbf{love}^{\iota\to\iota\to o}x^{\iota}}^{\iota\to o}\, y^{\iota}}^{o} \wedge \overbrace{\phi e'}^{o}}^{o}}^{\Omega})}^{\iota\to\Omega})}^{\Omega}}^{\iota\to\Omega})}^{\Omega}}^{((\iota\to\Omega)\to\Omega)\to((\iota\to\Omega)\to\Omega)\to\Omega} \tag{8}$$

Example 2.4. [**D**, *John loves Mary*] Now, given lexical interpretations (8), (5) and (6) of *loves*, *Mary* and *John* respectively, the meaning (3) of Sentence (1) can be computed compositionally according to the parse tree in Figure 1:

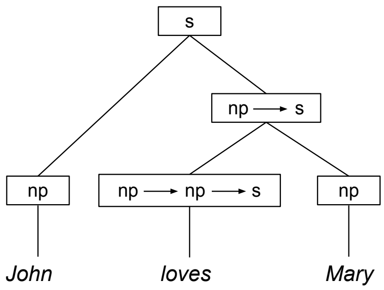

Figure 1: Syntactic parse tree of sentence *John loves Mary*.

$$
\begin{aligned}
\mathbf{D} &= [\![loves]\!][\![Mary]\!][\![John]\!] \\
&= (\lambda \mathbf{YX}.\mathbf{X}(\lambda x.\mathbf{Y}(\lambda y.(\lambda e'\phi.\mathbf{love}xy \wedge \phi e'))))[\![Mary]\!][\![John]\!] \\
&\to_\beta (\lambda \mathbf{X}.\mathbf{X}(\lambda x.[\![Mary]\!](\lambda y.(\lambda e'\phi.\mathbf{love}xy \wedge \phi e'))))[\![John]\!] \\
&\to_\beta [\![John]\!](\lambda x.[\![Mary]\!](\lambda y.(\lambda e'\phi.\mathbf{love}xy \wedge \phi e'))) \\
&= [\![John]\!](\lambda x.(\lambda \mathbf{P}.\lambda e\phi.\mathbf{Pm}e(\lambda e.\phi(\mathbf{m} :: e')))(\lambda y.(\lambda e'\phi.\mathbf{love}xy \wedge \phi e'))) \\
&\to_\beta [\![John]\!](\lambda x.\lambda e\phi.(\lambda y.(\lambda e'\phi.\mathbf{love}xy \wedge \phi e'))\mathbf{m}e(\lambda e'.\phi(\mathbf{m} :: e'))) \\
&\to_\beta [\![John]\!](\lambda x.\lambda e\phi.(\lambda e'\phi.\mathbf{love}x\mathbf{m} \wedge \phi e')e(\lambda e'.\phi(\mathbf{m} :: e'))) \\
&\to_\beta^* [\![John]\!](\lambda x.\lambda e\phi.\mathbf{love}x\mathbf{m} \wedge (\lambda e'.\phi(\mathbf{m} :: e')e)) \\
&\to_\beta [\![John]\!](\lambda x.\lambda e\phi.\mathbf{love}x\mathbf{m} \wedge \phi(\mathbf{m} :: e)) \\
&= (\lambda \mathbf{P}.\lambda e\phi.\mathbf{Pj}e(\lambda e'.\phi(\mathbf{j} :: e')))(\lambda x.\lambda e\phi.\mathbf{love}x\mathbf{m} \wedge \phi(\mathbf{m} :: e)) \\
&\to_\beta \lambda e\phi.(\lambda x.\lambda e\phi.\mathbf{love}x\mathbf{m} \wedge \phi(\mathbf{m} :: e))\mathbf{j}e(\lambda e'.\phi(\mathbf{j} :: e')) \\
&\to_\beta^* \lambda e\phi.\mathbf{lovejm} \wedge (\lambda e'.\phi(\mathbf{j} :: e'))(\mathbf{m} :: e) \\
&\to_\beta \lambda e\phi.\mathbf{lovejm} \wedge \phi(\mathbf{j} :: \mathbf{m} :: e)
\end{aligned} \tag{9}
$$

□

To cope with anaphora, the context has to be accessed. This is accomplished by a special function sel of type $(\gamma \to \iota)$ that takes a context and returns an individual. The function sel is assumed to implement an anaphora resolution algorithm and to work as an oracle always retrieving the correct antecedent. This allows to interpret pronouns as shown, for example, for *he* below:

$$
[\![he]\!] = \overbrace{\lambda \mathbf{P}^{\iota\to\gamma\to(\gamma\to o)\to o}.\lambda e^{\gamma}\phi^{\gamma\to o}.\ \overbrace{\overbrace{\overbrace{\mathbf{P}(\overbrace{\mathsf{sel}_{he} e}^{\iota})}^{\gamma\to(\gamma\to o)\to o}\ e}^{(\gamma\to o)\to o}\ \phi}^{o}}^{(\iota\to\gamma\to(\gamma\to o)\to o)\to\gamma\to(\gamma\to o)\to o} \tag{10}
$$

Example 2.5. [**S**, *He smiles at her*] The meaning of the sentence (2) computed in accordance with the parse-tree shown in Figure 2 is as follows:

(2) *He smiles at her.*

$$
\begin{aligned}
\mathbf{S} &= [\![smiles\_at]\!][\![her]\!][\![he]\!] \\
&= (\lambda \mathbf{YX}.\mathbf{X}(\lambda x.\mathbf{Y}(\lambda y.(\lambda e'\phi.\mathbf{smile}xy \wedge \phi e'))))[\![her]\!][\![he]\!] \\
&\to_\beta (\lambda \mathbf{X}.\mathbf{X}(\lambda x.[\![her]\!](\lambda y.(\lambda e'\phi.\mathbf{smile}xy \wedge \phi e'))))[\![he]\!] \\
&\to_\beta [\![he]\!](\lambda x.[\![her]\!](\lambda y.(\lambda e'\phi.\mathbf{smile}xy \wedge \phi e'))) \\
&= [\![he]\!](\lambda x.(\lambda \mathbf{P}.\lambda e\phi.\mathbf{P}(\mathsf{sel}_{her}e)e\phi)(\lambda y.(\lambda e'\phi.\mathbf{smile}xy \wedge \phi e'))) \\
&\to_\beta [\![he]\!](\lambda x.(\lambda e\phi.(\lambda y.(\lambda e'\phi.\mathbf{smile}xy \wedge \phi e'))(\mathsf{sel}_{her}e)e\phi)) \\
&\to_\beta [\![he]\!](\lambda x.(\lambda e\phi.(\lambda e'\phi.\mathbf{smile}x(\mathsf{sel}_{her}e) \wedge \phi e')e\phi)) \\
&\to_\beta^* [\![he]\!](\lambda x.(\lambda e\phi.\mathbf{smile}x(\mathsf{sel}_{her}e) \wedge \phi e)) \\
&= (\lambda \mathbf{P}.\lambda e\phi.\mathbf{P}(\mathsf{sel}_{he}e)e\phi)(\lambda x.(\lambda e\phi.\mathbf{smile}x(\mathsf{sel}_{her}e) \wedge \phi e)) \\
&\to_\beta \lambda e\phi.(\lambda x.(\lambda e\phi.\mathbf{smile}x(\mathsf{sel}_{her}e) \wedge \phi e))(\mathsf{sel}_{he}e)e\phi \\
&\to_\beta \lambda e\phi.(\lambda e\phi.\mathbf{smile}(\mathsf{sel}_{he}e)(\mathsf{sel}_{her}e) \wedge \phi e)e\phi \\
&\to_\beta^* \lambda e\phi.\mathbf{smile}(\mathsf{sel}_{he}e)(\mathsf{sel}_{her}e) \wedge \phi e
\end{aligned} \tag{11}
$$

□

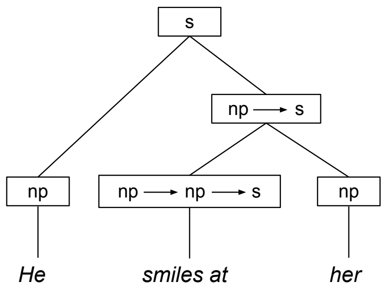

Figure 2: Syntactic parse tree of sentence *He smiles at her.*

As Example 2.5 shows, Sentence (2) is meaningful in de Groote's approach in the sense that it has an interpretation (11). However, the function sel can return individuals for *he* and *her* only when the sentence is evaluated over some context containing the corresponding antecedents. This can be done when the sentence is uttered in a discourse and the representation of this discourse is already computed. When the meaning of the discourse is updated with the meaning of the sentence, the pronominal anaphora is resolved.

Discourses in [1] are, like sentences, interpreted as terms of type $(\gamma \to (\gamma \to o) \to o)$. The update of a discourse interpreted as $\mathbf{D}$ with a sentence interpreted as $\mathbf{S}$ results in interpretation upd $\mathbf{D}$ $\mathbf{S}$ of a new discourse. This interpretation is defined by the following equation:

$$\mathsf{upd}\ \mathbf{D}\ \mathbf{S} \doteq \overbrace{\lambda e^{\gamma}\phi^{\gamma\to o}.\overbrace{\overbrace{\mathbf{D}^{\gamma\to(\gamma\to o)\to o}}^{(\gamma\to o)\to o}e\overbrace{(\lambda e'^{\gamma}.\overbrace{\overbrace{\mathbf{S}^{\gamma\to(\gamma\to o)\to o}}^{(\gamma\to o)\to o}e'\,\phi}^{o})}^{\gamma\to o}}^{o}}^{\gamma\to(\gamma\to o)\to o} \tag{12}$$

Example 2.6. [$\mathsf{upd}\ \mathbf{D}\ \mathbf{S}$] Now interpretations (9) and (11) can be composed through equation (12), regarding (1) as the discourse updated with the sentence (2). This leads to the interpretation of the piece of discourse (3):

(3) *John loves Mary. He smiles at her.*

$$\begin{aligned}
& \lambda e\phi.(\overbrace{\lambda e\phi.\mathbf{lovejm} \wedge \phi(\mathbf{j} :: \mathbf{m} :: e)}^{\mathbf{D}})e(\lambda e'.(\overbrace{\lambda e\phi.\mathbf{smile}(\mathsf{sel}_{he}e)(\mathsf{sel}_{her}e) \wedge \phi e}^{\mathbf{S}})e'\phi) \\
\to_{\beta}^{*}\ & \lambda e\phi.(\lambda e\phi.\mathbf{lovejm} \wedge \phi(\mathbf{j} :: \mathbf{m} :: e))e(\lambda e'.\mathbf{smile}(\mathsf{sel}_{he}e')(\mathsf{sel}_{her}e') \wedge \phi e') \\
\to_{\beta}^{*}\ & \lambda e\phi.\mathbf{lovejm} \wedge (\lambda e'.\mathbf{smile}(\mathsf{sel}_{he}e')(\mathsf{sel}_{her}e') \wedge \phi e')(\mathbf{j} :: \mathbf{m} :: e) \\
\to_{\beta}\ & \lambda e\phi.\mathbf{lovejm} \wedge \mathbf{smile}(\mathsf{sel}_{he}(\mathbf{j} :: \mathbf{m} :: e))(\mathsf{sel}_{her}(\mathbf{j} :: \mathbf{m} :: e)) \wedge \phi(\mathbf{j} :: \mathbf{m} :: e)
\end{aligned} \tag{13}$$

□

Interpretation (13) of the discourse consisting of the utterance of (3) is computed in a compositional manner. Note that the context of the interpretation of the first sentence is passed to sel operators of the interpretation of the second sentence. Assuming that an anaphora resolution mechanism is implemented in sel, the following semantic representation of (3) is obtained:

$$\lambda e\phi.\mathbf{love}\ \mathbf{j}\ \mathbf{m} \wedge \mathbf{smile}\ \mathbf{j}\ \mathbf{m} \wedge \phi(\mathbf{j} :: \mathbf{m} :: e) \tag{14}$$

The context $(\mathbf{j} :: \mathbf{m} :: e)$ in (13) (and hence in (14)) is accessible for future computation. This means that the individuals $\mathbf{j}$ and $\mathbf{m}$ can serve as ancestors for anaphoric pronouns in the following sentences. However, this is not always the case. For example, assuming accessibility constraint requirements of DRT, the individuals introduced by quantifiers in Sentence (4) should not be accessible for anaphoric triggers outside of the sentence. However, they clearly should be accessible for anaphoric pronouns within the sentence.

(4) *Every farmer who owns a donkey beats it.*

Lexical item	Syntactic category	Continuation-based interpretation in G_0
farmer	n	$\lambda xe\phi.\mathbf{f}x \wedge \phi e$
donkey	n	$\lambda xe\phi.\mathbf{d}x \wedge \phi e$
owns	np $\to$ np $\to$ s	$\lambda\mathbf{YX}.\mathbf{X}(\lambda x.\mathbf{Y}(\lambda y.(\lambda e'\phi.\mathbf{o}xy \wedge \phi e')))$
beats	np $\to$ np $\to$ s	$\lambda\mathbf{YX}.\mathbf{X}(\lambda x.\mathbf{Y}(\lambda y.(\lambda e'\phi.\mathbf{b}xy \wedge \phi e')))$
a	n $\to$ np	$\lambda\mathbf{PQ}.\lambda e\phi.\exists(\lambda x.\mathbf{P}xe(\lambda e'.\mathbf{Q}x(x :: e')\phi))$
every	n $\to$ np	$\lambda\mathbf{PQ}.\lambda e\phi.(\forall x.\neg(\mathbf{P}xe(\lambda e'.\neg(\mathbf{Q}x(x :: e')(\lambda e'''.\top))))) \wedge \phi e$
who	(np $\to$ s) $\to$ n $\to$ n	$\lambda\mathbf{RQ}x.\lambda e\phi.\mathbf{Q}xe(\lambda e'.\mathbf{R}(\lambda\mathbf{P}.\mathbf{P}x)e'\phi)$
it	np	$\lambda\mathbf{P}.\lambda e\phi.\mathbf{P}(\mathsf{sel}_{it}e)e\phi$

Table 1: Continuation-based interpretations of lexical items of the sentence *Every farmer who owns a donkey beats it* in framework G_0.

This accessibility constraint can also be implemented in de Groote's approach. For example, lexical items of (4) can be assigned meanings shown in Table 1 that lead to the desirable interpretation of the sentence, as demonstrated below. Since the lexical interpretations are dynamic, the resulting dynamic meaning of the donkey sentence does not suffer the drawbacks of the static meaning.

Example 2.7. [*Every farmer who owns a donkey beats it*] The meaning of the noun phrase *a donkey* is computed by reducing the term $[\![a]\!][\![donkey]\!]$:

$$\begin{aligned}
[\![a]\!][\![donkey]\!] &= (\lambda\mathbf{PQ}.\lambda e\phi.\exists(\lambda y.\mathbf{P}ye(\lambda e'.\mathbf{Q}y(y :: e')\phi)))[\![donkey]\!] \\
&\to_\beta \lambda\mathbf{Q}.\lambda e\phi.\exists(\lambda y.[\![donkey]\!]ye(\lambda e'.\mathbf{Q}y(y :: e')\phi)) \\
&= \lambda\mathbf{Q}.\lambda e\phi.\exists(\lambda y.(\lambda xe\phi.\mathbf{d}x \wedge \phi e)ye(\lambda e'.\mathbf{Q}y(y :: e')\phi)) \\
&\to_\beta \lambda\mathbf{Q}.\lambda e\phi.\exists(\lambda y.(\lambda e\phi.\mathbf{d}y \wedge \phi e)e(\lambda e'.\mathbf{Q}y(y :: e')\phi)) \\
&\to_\beta^* \lambda\mathbf{Q}.\lambda e\phi.\exists(\lambda y.\mathbf{d}y \wedge (\lambda e'.\mathbf{Q}y(y :: e')\phi)e) \\
&\to_\beta \lambda\mathbf{Q}.\lambda e\phi.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{Q}y(y :: e)\phi) \qquad (15)
\end{aligned}$$

Note that in Equation (15) the environment passed as an argument to $\mathbf{Q}$ contains the variable y introduced by the existential quantifier. This means that this variable is available to the formula $\mathbf{Q}$. Note also that the continuation ϕ of the term (15) is within the scope of the existential quantifier.

The meaning of the verb phrase *owns a donkey* is computed by β-reducing the term $[\![owns]\!]([\![a]\!][\![donkey]\!])$:

$$
\begin{aligned}
& [\![owns]\!]([\![a]\!][\![donkey]\!]) \\
= {} & (\lambda \mathbf{YX}.\mathbf{X}(\lambda x.\mathbf{Y}(\lambda y.(\lambda e'\phi.\mathbf{o}xy \wedge \phi e'))))([\![a]\!][\![donkey]\!]) \\
\rightarrow_\beta {} & \lambda \mathbf{X}.\mathbf{X}(\lambda x.([\![a]\!][\![donkey]\!])(\lambda \mathbf{y}.(\lambda e'\phi.\mathbf{o}xy \wedge \phi e'))) \\
= {} & \lambda \mathbf{X}.\mathbf{X}(\lambda x.(\lambda \mathbf{Q}.\lambda e\phi.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{Q}y(y :: e)\phi))(\lambda y.(\lambda e'\phi.\mathbf{o}xy \wedge \phi e'))) \\
\rightarrow_\beta {} & \lambda \mathbf{X}.\mathbf{X}(\lambda x.(\lambda e\phi.\exists(\lambda y.\mathbf{d}y \wedge (\lambda y.(\lambda e'\phi.\mathbf{o}xy \wedge \phi e'))y(y :: e)\phi))) \\
\rightarrow_\beta^* {} & \lambda \mathbf{X}.\mathbf{X}(\lambda x.(\lambda e\phi.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e))))
\end{aligned}
$$

The dynamic meaning of the relative clause *who owns a donkey* is computed as follows:

$$
\begin{aligned}
& [\![who]\!]([\![owns]\!]([\![a]\!][\![donkey]\!])) \\
= {} & (\lambda \mathbf{RQ}x.\lambda e\phi.\mathbf{Q}xe(\lambda e'.\mathbf{R}(\lambda \mathbf{P}.\mathbf{P}x)e'\phi))([\![owns]\!]([\![a]\!][\![donkey]\!])) \\
\rightarrow_\beta {} & \lambda \mathbf{Q}x.\lambda e\phi.\mathbf{Q}xe(\lambda e'.([\![owns]\!]([\![a]\!]\ [\![donkey]\!]))(\lambda \mathbf{P}.\mathbf{P}x)e'\phi) \\
= {} & \lambda \mathbf{Q}x.\lambda e\phi.\mathbf{Q}xe(\lambda e'.(\lambda \mathbf{X}.\mathbf{X}(\lambda x.(\lambda e\phi.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e)))))(\lambda \mathbf{P}.\mathbf{P}x)e'\phi) \\
\rightarrow_\beta {} & \lambda \mathbf{Q}x.\lambda e\phi.\mathbf{Q}xe(\lambda e'.(\lambda \mathbf{P}.\mathbf{P}x)(\lambda x.(\lambda e\phi.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e))))e'\phi) \\
\rightarrow_\beta {} & \lambda \mathbf{Q}x.\lambda e\phi.\mathbf{Q}xe(\lambda e'.(\lambda x.(\lambda e\phi.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e))))xe'\phi) \\
\rightarrow_\beta {} & \lambda \mathbf{Q}x.\lambda e\phi.\mathbf{Q}xe(\lambda e'.(\lambda e\phi.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e)))e'\phi) \\
\rightarrow_\beta^* {} & \lambda \mathbf{Q}x.\lambda e\phi.\mathbf{Q}xe(\lambda e'.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e'))) \qquad (16)
\end{aligned}
$$

The meaning of *farmer who owns a donkey* is computed by applying the λ-term (16) to the lexical interpretation of *farmer*:

$$
\begin{aligned}
& ([\![who]\!]([\![owns]\!]([\![a]\!][\![donkey]\!])))[\![farmer]\!] \\
= {} & (\lambda \mathbf{Q}x.\lambda e\phi.\mathbf{Q}xe(\lambda e'.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e'))))[\![farmer]\!] \\
\rightarrow_\beta {} & \lambda x.\lambda e\phi.[\![farmer]\!]xe(\lambda e'.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e'))) \\
= {} & \lambda x.\lambda e\phi.(\lambda xe\phi.\mathbf{f}x \wedge \phi e)xe(\lambda e'.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e'))) \\
\rightarrow_\beta {} & \lambda x.\lambda e\phi.(\lambda e\phi.\mathbf{f}x \wedge \phi e)e(\lambda e'.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e'))) \\
\rightarrow_\beta^* {} & \lambda x.\lambda e\phi.\mathbf{f}x \wedge (\lambda e'.\exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e')))e \\
\rightarrow_\beta {} & \lambda x.\lambda e\phi.\mathbf{f}x \wedge \exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e))
\end{aligned}
$$

The dynamic meaning of the noun phrase *every farmer who owns a donkey* is computed by applying the meaning of *every* to the meaning of *farmer who* *owns a donkey.*

$$
\begin{aligned}
& [\![every]\!](([\![who]\!]([\![owns]\!]([\![a]\!][\![donkey]\!])))[\![farmer]\!]) \\
&= (\lambda \mathbf{PQ}.\lambda e\phi.(\forall x.\neg(\mathbf{P}xe(\lambda e'.\neg(\mathbf{Q}x(x :: e')(\lambda e'''.\top))))) \wedge \phi e) \\
&\quad (([\![who]\!]([\![owns]\!]([\![a]\!]\ [\![donkey]\!])))[\![farmer]\!]) \\
&\to_\beta \lambda \mathbf{Q}.\lambda e\phi.(\forall x.\neg((([\![who]\!]([\![owns]\!]([\![a]\!]\ [\![donkey]\!])))[\![farmer]\!]) \\
&\qquad\qquad xe(\lambda e'.\neg(\mathbf{Q}x(x :: e')(\lambda e'''.\top))))) \wedge \phi e \\
&= \lambda \mathbf{Q}.\lambda e\phi.(\forall x.\neg((\lambda x.\lambda e\phi.\mathbf{f}x \wedge \exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e))) \\
&\qquad\qquad xe(\lambda e'.\neg(\mathbf{Q}x(x :: e')(\lambda e'''.\top))))) \wedge \phi e \\
&\to_\beta \lambda \mathbf{Q}.\lambda e\phi.(\forall x.\neg((\lambda e\phi.\mathbf{f}x \wedge \exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \phi(y :: e))) \\
&\qquad\qquad e(\lambda e'.\neg(\mathbf{Q}x(x :: e')(\lambda e'''.\top))))) \wedge \phi e \\
&\to_\beta^* \lambda \mathbf{Q}.\lambda e\phi.(\forall x.\neg(\mathbf{f}x \wedge \exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \\
&\qquad\qquad\qquad (\lambda e'.\neg(\mathbf{Q}x(x :: e')(\lambda e'''.\top)))(y :: e)))) \wedge \phi e \\
&\to_\beta \lambda \mathbf{Q}.\lambda e\phi.(\forall x.\neg(\mathbf{f}x \wedge \exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \neg(\mathbf{Q}x(x :: y :: e)(\lambda e'''.\top))))) \wedge \phi e
\end{aligned} \tag{17}
$$

Note that in Equation (17) the environment containing all the individuals with their properties collected during the computation is locally passed to the formula $\mathbf{Q}$. The continuation ϕ receives only the environment e that is passed to the term as an argument; therefore, all individuals collected during the computation of the meaning of *every farmer who owns a donkey* are not available outside the logical formula interpreting this phrase.

The meaning of the verb phrase *beats it* is computed as follows:

$$
\begin{aligned}
[\![beats]\!][\![it]\!] &= (\lambda \mathbf{YX}.\mathbf{X}(\lambda x.\mathbf{Y}(\lambda y.(\lambda e'\phi.\mathbf{b}xy \wedge \phi e'))))[\![it]\!] \\
&\to_\beta \lambda \mathbf{X}.\mathbf{X}(\lambda x.[\![it]\!](\lambda y.(\lambda e'\phi.\mathbf{b}xy \wedge \phi e'))) \\
&= \lambda \mathbf{X}.\mathbf{X}(\lambda x.(\lambda \mathbf{P}.\lambda e\phi.\mathbf{P}(\mathsf{sel}_{it}e)e\phi)(\lambda y.(\lambda e'\phi.\mathbf{b}xy \wedge \phi e'))) \\
&\to_\beta \lambda \mathbf{X}.\mathbf{X}(\lambda x.\lambda e\phi.(\lambda y.(\lambda e'\phi.\mathbf{b}xy \wedge \phi e'))(\mathsf{sel}_{it}e)e\phi) \\
&\to_\beta \lambda \mathbf{X}.\mathbf{X}(\lambda x.\lambda e\phi.(\lambda e'\phi.\mathbf{b}x(\mathsf{sel}_{it}e) \wedge \phi e')e\phi) \\
&\to_\beta^* \lambda \mathbf{X}.\mathbf{X}(\lambda x.\lambda e\phi.\mathbf{b}x(\mathsf{sel}_{it}e) \wedge \phi e)
\end{aligned} \tag{18}
$$

Finally, the dynamic meaning of the sentence is computed by applying the term (18) to the term (17):

$$
\begin{aligned}
& [\![beats]\!][\![it]\!]([\![every]\!](([\![who]\!]([\![owns]\!]([\![a]\!][\![donkey]\!])))[\![farmer]\!])) \\
= \; & (\lambda \mathbf{X}.\mathbf{X}(\lambda x.\lambda e\phi.\mathbf{b}x(\mathsf{sel}_{it}e) \wedge \phi e)) \\
& ([\![every]\!](([\![who]\!]([\![owns]\!]([\![a]\!][\![donkey]\!])))[\![farmer]\!])) \\
\to_\beta \; & ([\![every]\!](([\![who]\!]([\![owns]\!]([\![a]\!][\![donkey]\!])))[\![farmer]\!]))(\lambda x.\lambda e\phi.\mathbf{b}x(\mathsf{sel}_{it}e) \wedge \phi e) \\
= \; & (\lambda \mathbf{Q}.\lambda e\phi.(\forall x.\neg(\mathbf{f}x \wedge \exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \neg(\mathbf{Q}x(x :: y :: e)(\lambda e'''.\top))))) \wedge \phi e) \\
& (\lambda x.\lambda e\phi.\mathbf{b}x(\mathsf{sel}_{it}e) \wedge \phi e) \\
\to_\beta \; & \lambda e\phi.(\forall x.\neg(\mathbf{f}x \wedge \exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \\
& \qquad\qquad \neg((\lambda x.\lambda e\phi.\mathbf{b}x(\mathsf{sel}_{it}e) \wedge \phi e)x(x :: y :: e)(\lambda e'''.\top))))) \wedge \phi e \\
\to_\beta \; & \lambda e\phi.(\forall x.\neg(\mathbf{f}x \wedge \exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \\
& \qquad \neg((\lambda e\phi.\mathbf{b}x(\mathsf{sel}_{it}e) \wedge \phi e)(x :: y :: e)(\lambda e'''.\top))))) \wedge \phi e \\
\to_\beta^* \; & \lambda e\phi.(\forall x.\neg(\mathbf{f}x \wedge \exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \\
& \qquad \neg(\mathbf{b}x(\mathsf{sel}_{it}(x :: y :: e)) \wedge (\lambda e'''.\top)(x :: y :: e))))) \wedge \phi e \\
\to_\beta \; & \lambda e\phi.(\forall x.\neg(\mathbf{f}x \wedge \exists(\lambda y.\mathbf{d}y \wedge \mathbf{o}xy \wedge \neg(\mathbf{b}x(\mathsf{sel}_{it}(x :: y :: e)) \wedge \top)))) \wedge \phi e
\end{aligned}
\tag{19}
$$

The resulting dynamic interpretation (19) of the donkey sentence is logically equivalent to (20):

$$\lambda e\phi.\forall(\lambda x.\mathbf{f}x \to \forall(\lambda y.(\mathbf{d}y \wedge \mathbf{o}xy) \to \mathbf{b}x(\mathsf{sel}_{it}(y :: x :: e))))) \wedge \phi e \tag{20}$$

Note that, in accordance with DRT's accessibility constraint, the individuals bound by quantifiers are not accessible outside the sentence. □

First of all, the second argument of $\mathbf{b}$, standing for the anaphoric pronoun, is not a free dummy variable, but a term $(\mathsf{sel}_{it}(y :: x :: e))$. This term consists of the selection function sel that takes as argument a context containing the available individuals "collected" during the computation. Thus, in contrast to the static case, the second argument of $\mathbf{b}$ is self-sufficient: the function sel, which implements an anaphora resolution algorithm, selects a required individual from the context. In the current case, the selection function returns the individual y, leading to the final dynamic meaning (21) of Sentence (4):

$$\lambda e\phi.\forall(\lambda x.\mathbf{f}x \to \forall(\lambda y.(\mathbf{d}y \wedge \mathbf{o}xy) \to \mathbf{b}xy)) \wedge \phi e \tag{21}$$

Moreover, in the dynamic interpretation, unlike in the static one, the formula $\mathbf{b}xy$ is within the scope of the quantifier binding the variable y, exactly as desired. Furthermore, the quantifier binding y has been changed during the computation from existential to universal, which is also impossible in the static approach. These improvements are the consequences of employing a continuation-passing technique.

The list below summarizes the advantages that de Groote's approach brought to compositional semantics:

- The approach is independent of the intermediate language used to express meanings of the expressions. This allows to use mathematical and logical theories developed outside computational linguistics.[2] Therefore, natural language phenomena can be explained in terms of well-established and well-understood theories.

- Variables do not have any special status and are variables in the usual mathematical sense. Therefore, the notions of free and bound variables are standard.

- There is no imperative dynamic notions as assignment functions, therefore destructive assignment problem does not hold. Meanings assigned to expressions are closed λ-terms.

- There is no need for rules that artificially extend the scope of quantifiers.

- Context and content are regarded separately, but they do interact during the computation of the meaning of discourse.

- The approach does not depend on any specific structure given to the context. In contrast, context is defined as a term of type parameter γ and, therefore, its structure can be altered when necessary.

- The approach is truly compositional: the meaning of a complex expression is computed by β-reducing the composition of the meanings of its lexical items.

3 Dynamic Logic: First Order Case

Although compositional dynamic framework G_0, introduced in [1] and reviewed in the previous section, have shown itself to be promising by successfully handling donkey anaphora, its interpretations look complex and the computation of the meaning can be difficult to understand. In his later talks,

[2] An extension of first logic language with λ is used here because it is convenient and intuitive.

de Groote proposed an improvement of framework G_0, called here framework G, that represents his semantics in a more concise and systematic way. To do so, de Groote defined a continuation-based dynamic logic and this section presents this logic.

3.1 Formal Details

Terms and types are given by Definitions 3.1 and 3.3:

Definition 3.1. [λ-terms] The set of **λ-terms** Λ is constructed from an enumerable set of variables $V = \{v, v_1, v_2, \dots\}$, logical constants $\wedge, \exists, \neg$, two special constants :: and sel, an enumerable set of predicate symbols $R = \{R_1, R_2, \dots\}$ and an enumerable set of constants $K = \{c, c_1, c_2, \dots\}$ using application and (function) abstraction:

$$\begin{aligned}
x \in V &\implies x \in \Lambda \\
c \in K &\implies c \in \Lambda \\
M, N \in \Lambda &\implies (MN) \in \Lambda \\
x \in V, M \in \Lambda &\implies (\lambda x.M) \in \Lambda \\
M \in \Lambda &\implies (\exists M) \in \Lambda \\
M, N \in \Lambda &\implies (M \wedge N) \in \Lambda \\
M \in \Lambda &\implies (\neg M) \in \Lambda \\
M, x \in \Lambda &\implies (M :: x) \in \Lambda \\
M \in \Lambda &\implies (\mathsf{sel}(M)) \in \Lambda
\end{aligned}$$

Definition 3.2. [Free variables] The set of **free variables** of t, $FV(t)$, is defined inductively as follows:

$$\begin{aligned}
FV(x) &= \{x\} \\
FV(c) &= \{\} \\
FV(MN) &= FV(M) \cup FV(N) \\
FV(\lambda x.M) &= FV(M) - \{x\}
\end{aligned}$$

Definition 3.3. [Types] The set T of types is defined inductively as follows:

Atomic types:	$\iota \in \mathrm{T}$	(static individuals)
	$o \in \mathrm{T}$	(static propositions)
	$\gamma \in \mathrm{T}$	(context)
Complex types:	$\alpha, \beta \in \mathrm{T} \implies$	$(\alpha \to \beta) \in \mathrm{T}$

Typing rules define typing relations between terms and types:

Definition 3.4. [Typing rules] A statement $t : \delta$, meaning t has type δ, is **derivable** from the basis Δ, i.e. $\Delta \vdash t : \delta$, if $\Delta \vdash t : \delta$ can be produced using the following rules:

$$\frac{}{\Gamma, x : \alpha \vdash x : \alpha}\ \text{axiom}$$

$$\frac{}{\Gamma \vdash \top : o}\ \text{axiom}$$

$$\frac{}{\Gamma, M : o, N : o \vdash M \wedge N : o}$$

$$\frac{}{\Gamma, M : \iota \to o \vdash \exists M : o}$$

$$\frac{}{\Gamma, M : o \vdash \neg M : o}$$

$$\frac{}{\Gamma, c : \gamma \vdash \mathsf{sel}\ c : \iota}$$

$$\frac{}{\Gamma, i : \iota, c : \gamma \vdash (i :: c) : \gamma}$$

$$\frac{}{\Gamma \vdash c_{iv} : \iota \to o}\ \text{axiom}$$

$$\frac{}{\Gamma \vdash c_{tv} : \iota \to \iota \to o}\ \text{axiom}$$

$$\frac{}{\Gamma \vdash c_{n} : \iota \to o}\ \text{axiom}$$

$$\frac{}{\Gamma \vdash c_{np} : (\iota \to o) \to o}\ \text{axiom}$$

$$\frac{}{\Gamma \vdash c_{rp} : (((\iota \to o) \to o) \to o) \to (\iota \to o) \to (\iota \to o)}\ \text{axiom}$$

$$\frac{\Gamma \vdash v : \alpha \to \beta \qquad \Gamma \vdash u : \alpha}{\Gamma \vdash vu : \beta}\ \text{app}$$

$$\frac{\Gamma, x : \alpha \vdash v : \beta}{\Gamma \vdash \lambda x.v : \alpha \to \beta}\ \text{abs}$$

where c_{tv}, c_{iv}, c_n, c_{np} and c_{rp} are constants standing for transitive and intransitive verbs, nouns, noun phrases and relative pronouns respectively. Typing rules for other syntactic categories can be added analogously.

The first axiom and the two rules (application and abstraction) are standard typing relations in simply-typed lambda calculus. The second axiom determines the type of the logical $\top$ symbol. The constant sel has type $(\gamma \to \iota)$ and the constant $::$ has type $(\iota \to \gamma \to \gamma)$.

Each static type can be dynamized in the following way:

Definition 3.5. [Dynamization of types] Let ι and o be atomic types, γ be a type parameter, α and β be arbitrary types. Then the types are **dynamized** in the following way:

$$\overline{\iota} \doteq \iota \tag{22a}$$

$$\overline{o} \doteq \gamma \to (\gamma \to o) \to o \tag{22b}$$

$$\overline{\alpha \to \beta} \doteq \overline{\alpha} \to \overline{\beta} \tag{22c}$$

Note that the type o of a static proposition is transformed to type $(\gamma \to (\gamma \to o) \to o)$. Type $(\gamma \to (\gamma \to o) \to o)$ is thus the type of a dynamic proposition. Therefore, dynamic propositions are functions from γ (type of context) and $(\gamma \to o)$ (type of continuation) to o (type of static proposition). Static and dynamic individuals are defined to have the same type.

For each logical constant ($\neg$, $\wedge$ and $\exists$), its dynamic counterpart is specified by the following definition:

Definition 3.6. [Dynamic logical constants] Let $\mathbf{A}$ and $\mathbf{B}$ be terms of type $(\gamma \to (\gamma \to o) \to o)$, $\mathbf{P}$ be the term of type $(\iota \to \gamma \to (\gamma \to o) \to o)$, e and e' be terms of type γ, ϕ be a term of type $(\gamma \to o)$, x be the term of type ι. **Dynamic negation, conjunction and existential quantification** are defined respectively as follows:

$$\sim \mathbf{A} \doteq \lambda e\phi.\neg(\mathbf{A}e(\lambda e'.\top)) \wedge \phi e \tag{23a}$$

$$\mathbf{A} \curlywedge \mathbf{B} \doteq \lambda e\phi.\mathbf{A}e(\lambda e'.\mathbf{B}e'\phi) \tag{23b}$$

$$\Sigma(\lambda x.\mathbf{P}[x]) \doteq \lambda e\phi.\exists(\lambda x.\mathbf{P}[x]\ (x :: e)\ \phi) \tag{23c}$$

Dynamic negation of a dynamic proposition $\mathbf{A}$, $\sim \mathbf{A}$, is an abbreviation used for the term shown in (23a). Within the body of this term, the continuation and context of $\mathbf{A}$ are "erased" (by giving the term $(\lambda e'.\top)$ as the second argument of $\mathbf{A}$)[3], the resulting static proposition is negated, and the conjunct ϕe is added. Note that both ϕ and e are variables bound by λ. Therefore, the context of $\mathbf{A}$ is not available to the continuation of the resulting term $\sim \mathbf{A}$. Dynamic existentially quantified term $\Sigma(\lambda x.\mathbf{P}[x])$ is an

[3]This can be more clearly seen from Corollary 3.11.

abbreviation for the term shown in (23c). It has a λ-abstraction over variables e and ϕ and an existentially quantified variable x. Its body contains $\mathbf{P}[x]$, which is given e updated with x, $(x :: e)$, and ϕ as arguments. Note that $\sim$ and Σ are defined respectively via $\neg$ and $\exists$. Dynamic conjunction is defined as a composition of two dynamic terms. The logical conjunction of static propositions is provided by the fact that each dynamic proposition has a conjunct ϕe in its body, as defined in (24a) of the next definition.

Given dynamic logical connectives, all static terms can be dynamized:

Definition 3.7. [Dynamization of terms] Let P be a term of type $(\iota_1 \to \cdots \to \iota_n \to o)$, A and B be terms of type o, $t_1, \ldots, t_n$ and x be terms of type ι. Then propositions, negated propositions, conjunctions of two propositions and existentially quantified propositions are dynamized in the following way:

$$\overline{\mathrm{P}t_1 \ldots t_n} \doteq \lambda e\phi.\mathrm{P}t_1 \ldots t_n \wedge \phi e \tag{24a}$$

$$\overline{\neg \mathrm{A}} \doteq \ \sim \overline{\mathrm{A}} \tag{24b}$$

$$\overline{\mathrm{A} \wedge \mathrm{B}} \doteq \overline{\mathrm{A}} \curlywedge \overline{\mathrm{B}} \tag{24c}$$

$$\overline{\exists(\lambda x.\mathrm{P}[x])} \doteq \Sigma(\lambda x.\overline{\mathrm{P}[x]}) \tag{24d}$$

Equation (24a) defines the dynamization of a proposition $\mathrm{P}t_1 \ldots t_n$ of type o by adding a λ-abstraction with two arguments e and ϕ (of types γ and $(\gamma \to o)$ respectively) and a conjunct ϕe. Therefore, the resulting dynamic term is of type $(\gamma \to (\gamma \to o) \to o)$, the type of a dynamic proposition. Equations 24b, 24c and 24d extend dynamization to non-atomic formulas.

Note that Definitions 3.6 and 3.7 allow representing de Groote's [1] (Section **??**) dynamic terms in a compact way. While interpretations in [1] explicitly show extra parameters, i.e. contexts and continuations, these new definitions make it possible to hide these parameters. Moreover, the resulting compact dynamic terms are structurally analogous to their original static counterparts and, hence, are more intuitive.

Remark 3.8. In equations below, terms on the left side of $\doteq$ abbreviate respective terms on the right side:

$$\mathbf{A} \Rightarrow \mathbf{B} \doteq \ \sim (\mathbf{A} \curlywedge \sim \mathbf{B}) \tag{25}$$

$$\Pi(\lambda x.\mathbf{P}[x]) \doteq \ \sim \Sigma(\lambda x. \sim \mathbf{P}[x]) \tag{26}$$

Proposition 3.9 proves an important β-equivalence that can be useful when computing interpretations of certain phrases containing an existentially quantified variable.

Proposition 3.9. *For all terms* A *and* B *of type* o *such that* $x \in FV(\mathrm{A})$ *and* $x \notin FV(\mathrm{B})$ *for an* x *of type* ι*, the following equivalence holds:*

$$\Sigma(\lambda x.\overline{\mathrm{A}[x]}) \curlywedge \overline{\mathrm{B}} =_\beta \Sigma(\lambda x.\overline{\mathrm{A}[x]} \curlywedge \overline{\mathrm{B}})$$

Proof.

$$\begin{aligned}
& \Sigma(\lambda x.\overline{\mathrm{A}[x]}) \curlywedge \overline{\mathrm{B}} \\
\to_\beta^* \; & (\lambda e\phi.\exists(\lambda x.\mathrm{A}[x] \wedge \phi(x :: e))) \curlywedge \overline{\mathrm{B}} && \text{(by (23c))} \\
= \; & \lambda e\phi.(\lambda e\phi.\exists(\lambda x.\mathrm{A}[x] \wedge \phi(x :: e)))e(\lambda e.\overline{\mathrm{B}}e\phi) && \text{(by (23b))} \\
\to_\beta^* \; & \lambda e\phi.(\lambda e\phi.\exists(\lambda x.\mathrm{A}[x] \wedge \phi(x :: e)))e(\lambda e.\mathrm{B} \wedge \phi e) && \text{(by (24a))} \\
\to_\beta^* \; & \lambda e\phi.\exists(\lambda x.\mathrm{A}[x] \wedge (\mathrm{B} \wedge \phi(x :: e))) && (27)
\end{aligned}$$

$$\begin{aligned}
& \Sigma(\lambda x.\overline{\mathrm{A}[x]} \curlywedge \overline{\mathrm{B}}) \\
= \; & \Sigma(\lambda x.\lambda e\phi.\overline{\mathrm{A}[x]}e(\lambda e.\overline{\mathrm{B}}e\phi)) && \text{(by (23b))} \\
\to_\beta^* \; & \Sigma(\lambda x.\lambda e\phi.(\lambda e\phi.\mathrm{A}[x] \wedge \phi e)e(\lambda e.\mathrm{B} \wedge \phi e)) && \text{(by (24a))} \\
\to_\beta^* \; & \Sigma(\lambda x.\lambda e\phi.\mathrm{A}[x] \wedge (\mathrm{B} \wedge \phi e)) && (28) \\
\to_\beta^* \; & \lambda e\phi.\exists(\lambda x.\mathrm{A}[x] \wedge (\mathrm{B} \wedge \phi(x :: e))) && \text{(by (23c))}
\end{aligned}$$

□

Proposition 3.10. *For all* h *and* v *of type* o*, and for all* u *of type* γ*, the following holds:*

$$\overline{h}u(\lambda e.v) = h \wedge v$$

where e *is a variable of type* γ*.*

Proof. The proof is by induction on the structure of the term h.

- h is a proposition of the form $\mathrm{P}t_1 \ldots t_n$.

$$\begin{aligned}
\overline{\mathrm{P}t_1 \ldots t_n}u(\lambda e.v) &= (\lambda e\phi.\mathrm{P}t_1 \ldots t_n \wedge \phi e)u(\lambda e.v) && \text{(by (24a))} \\
&\to_\beta (\lambda \phi.\mathrm{P}t_1 \ldots t_n \wedge \phi u)(\lambda e.v) \\
&\to_\beta \lambda \phi.\mathrm{P}t_1 \ldots t_n \wedge (\lambda e.v)u \\
&\to_\beta \lambda \phi.\mathrm{P}t_1 \ldots t_n \wedge v
\end{aligned}$$

- h is a negated proposition $\neg w$.

$$\begin{aligned}
\overline{\neg w}u(\lambda e.v) &= \sim \overline{w}u(\lambda e.v) && \text{(by (24b))}\\
&= (\lambda e\phi.\neg(\overline{w}e(\lambda e'.\top)) \wedge \phi e)u(\lambda e.v) && \text{(by (23a))}\\
&\rightarrow_\beta (\lambda\phi.\neg(\overline{w}u(\lambda e'.\top)) \wedge \phi u)(\lambda e.v)\\
&\rightarrow_\beta \neg(\overline{w}u(\lambda e'.\top)) \wedge (\lambda e.v)u\\
&\rightarrow_\beta \neg(\overline{w}u(\lambda e'.\top)) \wedge v\\
&= \neg(w \wedge \top) \wedge v && \text{(by I.H.)}\\
&= \neg w \wedge v && (29)
\end{aligned}$$

- h is a conjunction of two propositions $w \wedge z$.

$$\begin{aligned}
(\overline{w \wedge z})u(\lambda e.v) &= (\overline{w} \curlywedge \overline{z})u(\lambda e.v) && \text{(by (24c))}\\
&= (\lambda e\phi.\overline{w}e(\lambda e'.\overline{z}e'\phi))u(\lambda e.v) && \text{(by (23b))}\\
&\rightarrow_\beta (\lambda\phi.\overline{w}u(\lambda e'.\overline{z}e'\phi))(\lambda e.v)\\
&\rightarrow_\beta \overline{w}u(\lambda e'.\overline{z}e'(\lambda e.v))\\
&= \overline{w}u(\lambda e'.z \wedge v) && \text{(by I.H., } e \notin FV(v))\\
&= w \wedge (z \wedge v) && \text{(by I.H., } e' \notin FV(z \wedge v))\\
&\equiv (w \wedge z) \wedge v
\end{aligned}$$

- h is an existentially quantified formula of the form $\exists(\lambda x.\mathrm{P}[x])$.

$$\begin{aligned}
\overline{\exists(\lambda x.\mathrm{P}[x])}u(\lambda e.v) &= (\Sigma(\lambda x.\overline{\mathrm{P}[x]}))u(\lambda e.v) && \text{(by (24d))}\\
&= (\lambda e\phi.\exists(\lambda x.\mathbf{P}[x](x :: e)\phi))u(\lambda e.v) && \text{(by (23c))}\\
&\rightarrow_\beta (\lambda\phi.\exists(\lambda x.\mathbf{P}[x](x :: u)\phi))(\lambda e.v)\\
&\rightarrow_\beta \exists(\lambda x.\mathbf{P}[x](x :: u)(\lambda e.v))\\
&= \exists(\lambda x.\mathrm{P}[x] \wedge v) && \text{(by I.H.)}\\
&= \exists(\lambda x.\mathrm{P}[x]) \wedge v && (x \notin FV(v))
\end{aligned}$$

$\square$

Corollary 3.11. *For all propositions t of type o, and for all terms u of type γ, the following folds:*

$$\bar{t}u(\lambda e.\top) \equiv t$$

where e is a variable of type γ.

Proof. Take v equal to $\top$ in Proposition 3.10. Then

$$\bar{t}u(\lambda e.\top) = t \wedge \top \equiv t$$

$\square$

Definition 3.12. A dynamic proposition $\mathbf{t}$ is true in a model $\mathcal{M}$, denoted $\mathcal{M} \models_{dyn} \mathbf{t}$, if and only if $\mathcal{M} \models \mathbf{t}u(\lambda e.\top)$ for every u of type γ.

Theorem 3.13 (Conservation)**.** *A proposition t is true in a model $\mathcal{M}$ if and only if its dynamic variant $\bar{t}$ is true in the same model:*

$$\mathcal{M} \models t \textit{ iff } \mathcal{M} \models_{dyn} \bar{t}$$

Proof. If $\mathcal{M} \models t$, then, by Corollary 3.11, $\mathcal{M} \models \bar{t}u(\lambda e.\top)$. Therefore, by Definition 3.12, $\mathcal{M} \models_{dyn} \bar{t}$.

If $\mathcal{M} \models_{dyn} \bar{t}$, then, by Definition 3.12, $\mathcal{M} \models \bar{t}u(\lambda e.\top)$. Therefore, by Corollary 3.11, $\mathcal{M} \models t$. $\square$

The conservation theorem proves that a static proposition and its dynamic version defined in this section hold in the same models.

3.2 Donkey Sentences

Tables 2 and 3 show respectively static and dynamic (according to G) interpretations for the lexical items in the donkey sentence (5):

(5) *Every farmer who owns a donkey beats it.*

Note that the type of every dynamic term is analogous to its static type. The only difference is that each atomic type of a dynamic term is dynamized according to Definition 3.5. All terms in Table 3, except the interpretation of the pronoun *it*, are dynamized following the rules in Definition 3.7. These rules allow the presentation of dynamic terms in a compact way. They ensure that dynamic terms are structurally analogous to their static counterparts and, therefore, are more intuitive. The dynamic interpretation of *it* is constructed not by directly following the dynamization rules, because *it* is a unconventional lexical item: there is an anaphor to be solved. Therefore, $\widetilde{[\![it]\!]}$[4] contains the selection function sel that takes a context (from which a referent has to be chosen) as an argument.

[4]Here and further on, dynamic interpretations of unconventional lexical items are marked with tilde.

Lexical item	Static type	Static interpretation
farmer	$\iota \to o$	$\mathbf{f}$
donkey	$\iota \to o$	$\mathbf{d}$
owns	$((\iota \to o) \to o) \to ((\iota \to o) \to o) \to o$	$\lambda \mathrm{YX}.\mathrm{X}(\lambda x.\mathrm{Y}(\lambda y.\mathbf{o}xy))$
beats	$((\iota \to o) \to o) \to ((\iota \to o) \to o) \to o$	$\lambda \mathrm{YX}.\mathrm{X}(\lambda x.\mathrm{Y}(\lambda y.\mathbf{b}xy))$
every	$(\iota \to o) \to ((\iota \to o) \to o)$	$\lambda \mathrm{PQ}.\forall(\lambda x.\mathrm{P}x \to \mathrm{Q}x)$
a	$(\iota \to o) \to ((\iota \to o) \to o)$	$\lambda \mathrm{PQ}.\exists(\lambda x.\mathrm{P}x \wedge \mathrm{Q}x)$
who	$(((\iota \to o) \to o) \to o) \to (\iota \to o) \to (\iota \to o)$	$\lambda \mathrm{RQ}x.\mathrm{Q}x \wedge \mathrm{R}(\lambda \mathrm{P}.\mathrm{P}x)$
it	$(\iota \to o) \to o$	$\lambda \mathrm{P}.\mathrm{P}?$

Table 2: Static lexical interpretations.

Lexical item	Dynamic type	Dynamic interpretation in G
farmer	$\bar{\iota} \to \bar{o}$	$\bar{\mathbf{f}}$
donkey	$\bar{\iota} \to \bar{o}$	$\bar{\mathbf{d}}$
owns	$((\bar{\iota} \to \bar{o}) \to \bar{o}) \to ((\bar{\iota} \to \bar{o}) \to \bar{o}) \to \bar{o}$	$\lambda \mathbf{YX}.\mathbf{X}(\lambda x.\mathbf{Y}(\lambda y.\bar{\mathbf{o}}xy))$
beats	$((\bar{\iota} \to \bar{o}) \to \bar{o}) \to ((\bar{\iota} \to \bar{o}) \to \bar{o}) \to \bar{o}$	$\lambda \mathbf{YX}.\mathbf{X}(\lambda x.\mathbf{Y}(\lambda y.\bar{\mathbf{b}}xy))$
every	$(\bar{\iota} \to \bar{o}) \to ((\bar{\iota} \to \bar{o}) \to \bar{o})$	$\lambda \mathbf{PQ}.\Pi(\lambda x.\mathbf{P}x \Rightarrow \mathbf{Q}x)$
a	$(\bar{\iota} \to \bar{o}) \to ((\bar{\iota} \to \bar{o}) \to \bar{o})$	$\lambda \mathbf{PQ}.\Sigma(\lambda x.\mathbf{P}x \curlywedge \mathbf{Q}x)$
who	$(((\bar{\iota} \to \bar{o}) \to \bar{o}) \to \bar{o}) \to (\bar{\iota} \to \bar{o}) \to (\bar{\iota} \to \bar{o})$	$\lambda \mathbf{RQ}x.\mathbf{Q}x \curlywedge \mathbf{R}(\lambda \mathbf{P}.\mathbf{P}x)$
it	$\lambda \mathbf{P}.\lambda e\phi.\mathbf{P}(\mathsf{sel}_{it}e)e\phi$	

Table 3: Dynamic lexical interpretations in framework G.

Taking these dynamic interpretations to compute the meaning of Sentence (5), term (30) β-reduces to term (31), which normalizes to (32):

$$\overline{[\![beats]\!]}\ \widetilde{[\![it]\!]}(\overline{[\![every]\!]}((\overline{[\![who]\!]}(\overline{[\![owns]\!]}(\overline{[\![a]\!]}\ \overline{[\![donkey]\!]})))\overline{[\![farmer]\!]})) \quad (30)$$

$$\to^*_\beta \Pi(\lambda x.(\bar{\mathbf{f}}x \curlywedge \Sigma(\lambda z.\bar{\mathbf{d}}z \curlywedge \bar{\mathbf{o}}xz)) \Rightarrow (\lambda e\phi.\bar{\mathbf{b}}x(\mathsf{sel}_{it}e)e\phi)) \quad (31)$$

$$\to^*_\beta \lambda e\phi.\forall(\lambda x.\mathbf{f}x \to \forall(\lambda z.(\mathbf{d}z \wedge \mathbf{o}xz) \to \mathbf{b}x(\mathsf{sel}_{it}(x :: z :: e)))) \wedge \phi e \quad (32)$$

Resulting term (32) is equivalent to (20) obtained in framework G_0 interpretations. Indeed, framework G is equivalent to de Groote's [1] framework G_0. However, it is advantageous over G_0 due to the compact notations for dynamic terms. These notations significantly systematize the framework and make the interpretations more concise and intuitive. Moreover, the systematic translations of static terms into dynamic terms makes it possible to prove a conservation result 3.13 for G, that guarantees that static and dynamic interpretations are satisfied in the same models.

4 Comparison With Other Work

[2]

5 Higher Order Case

6 Higher order case and Conservation

This is a test. This is another test. Yes indeed.

References

[1] P. de Groote. Towards a montagovian account of dynamics. In *Semantics and Linguistic Theory XVI*, 2006.

[2] J. Groenendijk and M. Stokhof. Dynamic predicate logic. *Linguistics and Philosophy*, 14(1):39–100, 1991.