Continuations and adverbial modifiers

Christophe Raffalli, Kurt Ranalter

Continuations and adverbial modifiers

Christophe Raffalli & Kurt Ranalter
christophe.raffalli@univ-savoie.fr
kurt@dcs.qmul.ac.uk

Abstract

We extend de Groote's approach to natural language or Montague semantics with labels and combine it with Parsons event semantics. We argue that within this framework continuations allow one to deal with adverbs.

1 Introduction

The use of λ-calculus in Montague semantics does not only allow one to parse natural language sentences but also to provide a formal semantics for them as statements in predicate logic. It has been argued by de Groote in [dG01] and by Barker in [Bar02] that the type raising mechanism introduced by Montague in [Mon73] is closely related to continuations, a technical device used in the study of programming language that also provides means to give a computational interpretation of proofs in classical logic. The cornerstone in Montague's proposal consists in the observation that noun phrases and proper nouns can be treated in the same fashion if their corresponding types are raised by a sort of double-negation translation; with respect to the encoding of lexical entries as λ-terms this means that they have to carry around a parameter, a so-called continuation, that takes into account the environment of the actual computation. Montague showed that with the help of type raising one can give a uniform treatment of quantification in English. For instance, a singular indefinite article occurring in a noun phrase deep inside a sentence gets interpreted as an existential quantifier that may have scope over larger parts of the sentence than just the noun phrase, thus accommodating the fact that the surface structure of a sentence may differ from the structure of the proposition it expresses.

Since the pioneering work of de Groote and Barker it has been argued that continuations play a major role in natural language semantics and that they provide a uniform treatment for various linguistic phenomena (see for instance the list of problems tackled in [Bar02, Bar04, Sha04, dG06, BM07]). In this note we consider a new item on the list of phenomena that can benefit from a continuation-based approach to natural language and show how one can deal with action sentences containing certain adverbial modifiers. The problem we consider is basically as follows. In general, if a sentence contains a negated verb then the negation should apply to the whole sentence. However, when both a negation and an adverb occur in the sentence then one would like to get also a semantic reading where the negation applies only to the adverb. We introduce a type-theoretic framework that is based on de Groote's $\lambda\mu$-calculus approach

advocated in [dG01] and that accounts for such a behavior. Using $\lambda\mu$-calculus instead of pure λ-calculus has the advantage that type raising can be kept at a minimum level and that, as a consequence thereof, one can work with less complex terms (compare for instance the encoding of lexical entries in [dG01] with the one given in [Bar02]).

It is worth mentioning that de Groote exploits the fact that the $\lambda\mu$-calculus with both μ and μ' reductions is not confluent: surface and inverse scope readings of a sentence containing an existential and a universal quantifier are obtained by varying the order of application of the μ and μ' reductions. We modify de Groote's framework in such a way that this feature can be avoided. The basic idea is to use a call-by-value variant of the $\lambda\mu$-calculus (see for instance chapter 6 of [Py98]): introducing values provides means to eliminate the critical (μ, μ') pair and thus to get a confluent calculus. To account for the different readings we introduce labels that allow one to push certain λ-applications inside a $\lambda\mu$-term and thus to get the β-redex needed for the computation of the semantic reading of a sentence. Hence, instead of varying the order of application of the μ and μ' reductions we vary the order of the λ-applications when assembling the various parts of the sentence under consideration. Note that it is essential for our approach to work with sets of parse trees.

To be able to deal properly with action sentences we need to be more specific about their logical form. Based on a remark by Ramsey, Davidson proposes to regard action sentences as propositions that quantify over some underlying event (see essay 6 in [Dav01]). Parsons takes the idea further in [Par90] and develops a coherent framework that shows how one can explain various linguistic data in terms of it.[1] Roughly speaking, the main idea of the framework is that an action sentence such as *Brutus stabbed Caesar* may be analyzed as follows: for some event e we have (a) that e is a stabbing, (b) that Brutus is the agent of e, and (c) that Caesar is the object or theme of e. Hence, instead of encoding a verb expressing an action as a predicate one encodes it as an existential quantification over an event. The advantage of the approach is that one can easily account for inferences such as if *Brutus stabbed Caesar violently* then *Brutus stabbed Caesar* that cause troubles to other approaches.

We show in detail how one can get the intended semantic readings of the French sentences *Nicolas mange vite une pomme*, *Nicolas ne mange pas une pomme*, and *Nicolas ne mange pas vite une pomme* within our framework. The first example sentence serves the purpose to illustrate the abovementioned treatment of action sentences. The second and the third one serve the purpose to illustrate the behavior with respect to negation: in the former case we get that there does not exist an event which is the eating of an apple by Nicolas; in the latter that there exists an event which is the eating of an apple by Nicolas, but that the event under consideration is not a fast one. This semantic reading of the third example sentence accounts for the fact that we may infer from the sentence that Nicolas eats the apple slowly. Note that we can also get a semantic reading of it that is analogous to the one for the second example sentence. The labels may even be refined in such a way as to provide means to impose restrictions that make it possible to rule out certain semantic readings.

[1] We would like to thank Graham White for making us aware of Parsons' work.

The paper is divided in two parts. In section 2 we introduce the logical framework. In section 3 we deal with linguistic applications: subsection 3.1 is about quantification and subsection 3.2 about adverbs and negation.

2 Logical framework

We provide a concise overview of our type-theoretical framework, an extension of the $\lambda\mu$-calculus endowed with labels and a generalized form of μ-abstraction.

Definition 2.1 The set $\mathcal{L}$ of labels is defined as $\{\epsilon, \mathtt{n}, \mathtt{s}, \mathtt{o}, \mathtt{v}, \mathtt{a}, \mathtt{k}, *\}$ and the set $\mathcal{P}$ of positions as $\{\epsilon, \mathtt{x}, \mathtt{l}, \mathtt{r}\}$. Throughout the paper we shall use the variables k or l to denote labels and q or p to denote positions.

Remark 2.2 Labels are used to encode certain informations in types and thus in terms. Indeed, `n`, `v`, and `a` stand for *noun phrase*, *verb*, and *adverb*. Further, `s` and `o` are refinements of `n` that allow one to distinguish between a noun phrase occurring in *subject* position and one occurring in *object* position. Note that `k` and $*$ are of a different sort: `k` is used to deal with continuations and $*$ to deal with negations. The marker `x` stands for an *unspecified* position, whereas `l` and `r` stand for *left* and *right*. Positions allow one to keep track of directionality issues and, as a consequence thereof, we can replace Lambek's / and \ by the labeled arrows $\rightarrow_k^{\mathtt{r}}$ and $\rightarrow_k^{\mathtt{l}}$, where k is an element of $\mathcal{L}$. We prefer to work with labeled arrows because this keeps the logical meaning of types obvious for the nonspecialist. Further, we use the convention that if ϵ occurs as a label or position in a type or term then we omit writing it.

Definition 2.3 Types are defined by the grammar

$$A ::= at \mid X(\vec{x}) \mid A \rightarrow_k^q A \qquad B ::= \forall \vec{X}.A \qquad (q \in \mathcal{P}, k \in \mathcal{L})$$

where at ranges over the set $\{\iota, o, \nu, \varsigma\}$ of atomic types.

Remark 2.4 The type ι stands for individuals and the type o for truth values. Since we are interested in linguistic applications the type $\perp$ of observable entities is defined as $\perp =_{def} o$. The informal interpretation of the atomic types ν and ς is as follows. ν is a special type that allows one to quantify over event variables. ς allows one to deal with the fact that in French negations are formed by the two natural language particles *ne* and *pas*, i.e. ς is the type of a partial sentence that is lacking one of the two words forming a negation.

Remark 2.5 In order to save space we introduce the following shorthand notation: $\kappa =_{def} \nu \rightarrow o$. It is also worth mentioning that both the atomic types ι and ν may depend on parameters that provide means to express morpho-syntactic and sentence-level features. Indeed, we shall say a bit more about this crucial issue towards the end of section 3.2.

Remark 2.6 We use ML-like polymorphism to be able to reuse terms to interpret words in more contexts. Note that the typing of schema (polymorphic terms) is denoted by the sign $\vdash_{s_2}$ instead of $\vdash$.

$$\frac{}{\Gamma, x\colon A \vdash x\colon A} \qquad \frac{}{\vdash_{s_2} \mathsf{c}\colon \forall \vec{X}.A}$$

$$\frac{\Gamma, x\colon A \vdash M\colon A'}{\Gamma \vdash \lambda^q_k x.M\colon A \to^q_k A'} \qquad \frac{\Gamma \vdash M\colon A \to^q_k A' \quad \Gamma \vdash N\colon A}{\Gamma \vdash (M\ {}^q_k\ N)\colon A'}$$

$$\frac{\Gamma, \alpha\colon A^\perp \vdash M\colon A}{\Gamma, \alpha\colon A^\perp \vdash [\alpha]M\colon \perp} \qquad \frac{\Gamma, \alpha\colon A^\perp \vdash M\colon A_1 \to^{q_1}_{k_1} \dots A_n \to^{q_n}_{k_n} \perp}{\Gamma \vdash \mu_{(k_1 \dots k_n)}\alpha.M\colon A_1 \to^{q_1}_{k_1} \dots A_n \to^{q_n}_{k_n} A}$$

$$\frac{\emptyset \vdash M\colon A}{\vdash_{s_2} \Lambda \vec{X}.M\colon \forall \vec{X}.A} \qquad \frac{\vdash_{s_2} M\colon \forall \vec{X}.A}{\vdash (M\ \vec{A})\colon A[X_i(\vec{x}) := A_i(\vec{p})]}$$

Figure 1: Typing rules

Definition 2.7 Given $q, p \in \mathcal{P}$ and $k, l \in \mathcal{L}$, we quotient the set of types by the smallest congruence (or contextual equivalence) relation $\sim$ such that if $k \neq l$ then $A_1 \to^q_k (A_2 \to^p_l A_3) \sim A_2 \to^p_l (A_1 \to^q_k A_3)$.

Remark 2.8 Taking equivalence classes of types is particularly useful since, as a consequence, both $\iota \to^{\mathtt{l}}_{\mathtt{s}} \iota \to^{\mathtt{r}}_{\mathtt{o}} o$ and $\iota \to^{\mathtt{r}}_{\mathtt{o}} \iota \to^{\mathtt{l}}_{\mathtt{s}} o$ belong to the same equivalence class, the one standing for transitive verbs. Hence, whether one applies first the subject and then the object or vice versa does not matter at the level of types. This is useful, for instance, to deal with subordinates and treat both "who" and "which" uniformly. However, the order matters with respect to normalization and thus we can obtain various semantic readings in the case of ambiguous sentences: the rough idea is that a sentence is represented by a set of parse trees and that the evaluation of these parse trees may then lead to different interpretations. This issue will also be discussed in section 3.1.

Definition 2.9 Terms M and values V (a subset of M) are defined by the following grammars:

$$M ::= x \mid \mathsf{c} \mid \lambda^q_k x.M \mid (M\ {}^q_k\ M) \mid \mu_w \alpha.M \mid [\alpha]M \mid \Lambda \vec{X}.M \mid (M\ \vec{A})$$
$$V ::= x \mid \mathsf{c} \mid \lambda^q_k x.M \mid [\alpha]M \mid \Lambda \vec{X}.M \qquad (q \in \mathcal{P}, k \in \mathcal{L}, w \in \mathcal{L}^*)$$

Remark 2.10 The typing rules for terms are provided in figure 1 and the call-by-value (CBV) reductions we are interested in are listed in table 1. Let us point out that, as a consequence of the generalized form of μ-abstraction we have that, although in our examples we consider transitive verbs only, the framework applies without further modifications to intransitive verbs as well. Since λ- and μ-abstractions depend on labels $k \in \mathcal{L}$ and words $w \in \mathcal{L}^*$ we need to consider the commutations c_1 and c_2 of terms. In the case of the μ' reduction we have that none of the variables x_i occurring in $\vec{x}$ also occurs freely in N; the auxiliary term $\lambda_w \vec{x}.N$ can simply be regarded as a lifting of the term N that

$((\Lambda\vec{X}.M)\,\vec{A})$	$\rightarrow_\forall$	$M[X_i(\vec{x}) := A_i(\vec{p})]$	—
$((\lambda^q_k x.M)\,{}^p_l\,V)$	$\rightarrow_{c_1}$	$\lambda^q_k x.(M\,{}^p_l\,V)$	$k \neq l$
$((\lambda^q_k x.M)\,{}^q_k\,V)$	$\rightarrow_{\beta_v}$	$M[x := V]$	—
$((\mu_{(w_1 k w_2)}\alpha.M)\,{}^q_k\,V)$	$\rightarrow_{c_2}$	$\mu_{(w_1 w_2)}\alpha.(M\,{}^q_k\,V)$	—
$((\mu_w\alpha.M)\,{}^q_k\,V)$	$\rightarrow_{\mu_v}$	$\mu_w\beta.M[[\alpha]M' := [\beta](M'\,{}^q_k\,V)]$	$k \notin w$
$((\lambda_w\vec{x}.N)\,{}^q_k\,(\mu_w\alpha.M))$	$\rightarrow_{\mu'}$	$\mu_w\beta.M[[\alpha]M' := [\beta](N\,{}^q_k\,M')]$	$\vec{x} \notin N, k \notin w$
$\mu\alpha.M$	$\rightarrow_\epsilon$	$M[[\alpha]M' := M']$	if $\mu\alpha.M\colon\bot$

Table 1: CBV reductions

A	$\lambda^{\mathtt{l}}_{\mathtt{s}}x.\lambda^{\mathtt{r}}_{\mathtt{o}}y.(\mathtt{ai}\ x\ y)$ of type $\tau(M) = \iota \rightarrow^{\mathtt{l}}_{\mathtt{s}} \iota \rightarrow^{\mathtt{r}}_{\mathtt{o}} o$
U	$\lambda^{\mathtt{r}}_{\mathtt{n}}x.\mu\alpha.\exists y.[(x\ y) \wedge [\alpha]y]$ of type $\tau(U) = (\iota \rightarrow o) \rightarrow^{\mathtt{r}}_{\mathtt{n}} \iota$
C	$\lambda^{\mathtt{r}}_{\mathtt{n}}x.\mu\alpha.\forall y.[(x\ y) \rightarrow [\alpha]y]$ of type $\tau(C) = (\iota \rightarrow o) \rightarrow^{\mathtt{r}}_{\mathtt{n}} \iota$

Table 2: Encoding as terms I

provides means to match the typing constraints imposed by the generalized form of μ-abstraction. An instance of a μ' reduction with nonempty w can be found in figure 6 of example 3.7 below.

Remark 2.11 We shall make extended use of macro definitions for some of the logical connectives. $\exists x.M(x) =_{def} (\widetilde{\exists}\ \lambda x.M(x))$ where the constant $\exists$ is of type $\forall X.(X \rightarrow o) \rightarrow o$ and $\widetilde{\exists}$ is shorthand[2] for either $(\exists\,\iota)$ or $(\exists\,\nu)$, depending on whether one quantifies over individuals or events; universal quantification can be defined analogously. $M_1 \wedge M_2 =_{def} (\wedge\ M_1\ M_2)$, $M_1 \rightarrow M_2 =_{def} (\longrightarrow\ M_1\ M_2)$ and $\neg M =_{def} (\neg\ M)$ where $\wedge$ and $\longrightarrow$ are constants of type $o \rightarrow o \rightarrow o$ and $\neg$ is a constant of type $o \rightarrow o$.

3 Linguistic applications

3.1 Quantifiers and scope

We show how labels can be exploited to obtain the various semantic readings of an ambiguous sentence and compare our framework briefly with the one de Groote proposes in [dG01].

Example 3.1 Consider the sentence *Chaque homme aime une femme* to which we associate the set of parse trees consisting of:

$$[[[\text{ Chaque homme }]\text{ aime }][\text{ une femme }]] \tag{1}$$

[2]This is needed because our calculus is in Church style, i.e. with type annotation in terms, which is necessary in call-by-value: otherwise the μ' reduction would be unsound.

$$
\begin{array}{ll}
(A\,{}^{\mathtt{l}}_{\mathtt{s}}\,(C\,{}^{\mathtt{l}}_{\mathtt{n}}\,\mathtt{ho})\,{}^{\mathtt{r}}_{\mathtt{o}}\,\underline{(U\,{}^{\mathtt{r}}_{\mathtt{n}}\,\mathtt{fe})}) & \\
\rightarrow_{\beta_v} & \underline{(A\,{}^{\mathtt{l}}_{\mathtt{s}}\,(C\,{}^{\mathtt{r}}_{\mathtt{n}}\,\mathtt{ho})\,{}^{\mathtt{r}}_{\mathtt{o}}\,(\mu\alpha.\exists x.[(\mathtt{fe}\ x) \wedge [\alpha]x]))} \\
\rightarrow_{\mu'} & \mu\alpha.\exists x.[(\mathtt{fe}\ x) \wedge [\alpha](A\,{}^{\mathtt{l}}_{\mathtt{s}}\,\underline{(C\,{}^{\mathtt{r}}_{\mathtt{n}}\,\mathtt{ho})}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)] \\
\rightarrow_{\beta_v} & \mu\alpha.\exists x.[(\mathtt{fe}\ x) \wedge [\alpha]\underline{(A\,{}^{\mathtt{l}}_{\mathtt{s}}\,(\mu\beta.\forall y.[(\mathtt{ho}\ y) \rightarrow [\beta]y])\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)}] \\
\rightarrow_{\mu',\,\mu_v} & \mu\alpha.\exists x.[(\mathtt{fe}\ x) \wedge [\alpha](\mu\beta.\forall y.[(\mathtt{ho}\ y) \rightarrow [\beta]\underline{(A\,{}^{\mathtt{l}}_{\mathtt{s}}\,y\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)}])] \\
\rightarrow_{\beta_v^2} & \mu\alpha.\exists x.[(\mathtt{fe}\ x) \wedge [\alpha](\mu\beta.\forall y.[(\mathtt{ho}\ y) \rightarrow [\beta](\mathtt{ai}\ x\ y)])] \\
\rightarrow_{\epsilon^2} & \exists x.[(\mathtt{fe}\ x) \wedge \forall y.[(\mathtt{ho}\ y) \rightarrow (\mathtt{ai}\ x\ y)]]
\end{array}
$$

Figure 2: Reduction — quantification

$$[[\text{ Chaque homme }][\text{ aime }[\text{ une femme }]]] \tag{2}$$

In (1) the subject is applied first to the transitive verb whereas in (2) the object is applied first. If we encode *aime*, *une*, and *chaque* by the terms A, U, and C provided in table 2 and, further, use the constants $\mathtt{fe}$, $\mathtt{ho}$ of type $\iota \rightarrow o$ then the term $(A\,{}^{\mathtt{l}}_{\mathtt{s}}\,(C\,{}^{\mathtt{r}}_{\mathtt{n}}\,\mathtt{ho})\,{}^{\mathtt{r}}_{\mathtt{o}}\,(U\,{}^{\mathtt{r}}_{\mathtt{n}}\,\mathtt{fe}))$ of type o corresponds to the parse tree given in (1). As shown in figure 2 it reduces to

$$\exists x.[(\mathtt{fe}\ x) \wedge \forall y.[(\mathtt{ho}\ y) \rightarrow (\mathtt{ai}\ x\ y)]]$$

The term obtained by the reduction yields the so-called inverse scope reading of the sentence. The surface scope reading

$$\forall y.[(\mathtt{ho}\ y) \rightarrow \exists x.[(\mathtt{fe}\ x) \wedge (\mathtt{ai}\ x\ y)]]$$

is obtained by applying the same procedure to the term $(A\,{}^{\mathtt{r}}_{\mathtt{o}}\,(U\,{}^{\mathtt{r}}_{\mathtt{n}}\,\mathtt{fe})\,{}^{\mathtt{l}}_{\mathtt{s}}\,(C\,{}^{\mathtt{r}}_{\mathtt{n}}\,\mathtt{ho}))$ of type o which corresponds to the parse tree given in (2). Note that this works because $\tau(A) = \iota \rightarrow^{\mathtt{l}}_{\mathtt{s}} \iota \rightarrow^{\mathtt{r}}_{\mathtt{o}} o \sim \iota \rightarrow^{\mathtt{r}}_{\mathtt{o}} \iota \rightarrow^{\mathtt{l}}_{\mathtt{s}} o$.

Remark 3.2 In contrast to de Groote's approach in [dG01], it is essential for us to work with a set of parse trees. In this setup the congruence relation on types provides means to account for both semantic readings of the example sentence. It is worth mentioning that we have neither used one of the commutations c_1, c_2 nor exploited the fact of having a generalized form of μ-abstraction.

Remark 3.3 We have tried to keep things as simple as possible and thus have not discussed how one can deal with sentences that are not about actions: Parsons holds the view that these can be treated in a way similar to action sentence and proposes to encode verbs expressing states of affairs by quantifying over state variables.

3.2 Adverbs and negation

We consider now the three example sentences mentioned in the introduction. The treatment of quantification over events is inspired by de Groote's account of context representation provided in [dG06].

U	$\lambda^{\mathtt{r}}_{\mathtt{n}} x.\mu\alpha.\exists y.[(x\ y) \wedge [\alpha]y]$ of type $\tau(U) = (\iota \to o) \to^{\mathtt{r}}_{\mathtt{n}} \iota$
M	$\lambda^{\mathtt{l}}_{\mathtt{s}} x.\lambda^{\mathtt{r}}_{\mathtt{o}} y.\lambda^{\mathtt{x}}_{\mathtt{k}} c.\exists e.[\mathcal{M}(x,y,e) \wedge (c\ e)]$ of type $\tau(M) = \iota \to^{\mathtt{l}}_{\mathtt{s}} \iota \to^{\mathtt{r}}_{\mathtt{o}} \kappa \to^{\mathtt{x}}_{\mathtt{k}} o$
V	$\Lambda X(l).\lambda^{\mathtt{l}}_{\mathtt{v}} x.\lambda^{\mathtt{x}}_{\mathtt{k}} c.\mu_{(\mathtt{s}l)}\alpha.(x\,{}^{\mathtt{x}}_{\mathtt{k}}\,(\lambda e.[\alpha](\mathtt{vi}\ e) \wedge (c\ e)))$ of type $\tau(V) = \forall X(l).(\iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)) \to^{\mathtt{l}}_{\mathtt{v}} \iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)$
N	$\Lambda X(l).\lambda^{\mathtt{r}}_{\mathtt{v}} x.(x\,{}^{\mathtt{x}}_{*}\,\star)$ of type $\tau(N) = \forall X(l).(\varsigma \to^{\mathtt{x}}_{*} \iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)) \to^{\mathtt{r}}_{\mathtt{v}} \iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)$
P	$\Lambda X(l).\lambda^{\mathtt{l}}_{\mathtt{v}} x.\lambda^{\mathtt{x}}_{*} y.\mu\alpha.\neg[\alpha]x$ of type $\tau(P) = \forall X(l).(\iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)) \to^{\mathtt{l}}_{\mathtt{v}} \varsigma \to^{\mathtt{x}}_{*} \iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)$
P^*	$\Lambda X(l).\lambda^{\mathtt{l}}_{\mathtt{v}} x.\lambda^{\mathtt{r}}_{\mathtt{a}} y.\lambda^{\mathtt{x}}_{*} z.\lambda^{\mathtt{x}}_{\mathtt{k}} c.(\mathcal{N}(l)\ ((y\,{}^{\mathtt{l}}_{\mathtt{v}}\,x)\,{}^{\mathtt{x}}_{\mathtt{k}}\,c))$ of type $\tau(P^*) = \forall X(l).(\iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)) \to^{\mathtt{l}}_{\mathtt{v}} (\tau(V)\backslash\forall) \to^{\mathtt{r}}_{\mathtt{a}} \varsigma \to^{\mathtt{x}}_{*} \iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)$

Table 3: Encoding as terms II

Notation 3.4 For the encoding we need the following mapping from lexical entries to closed $\lambda\mu$-terms:

$$\mathit{une} \mapsto U \qquad \mathit{mange} \mapsto M \qquad \mathit{vite} \mapsto V \qquad \mathit{ne} \mapsto N \qquad \mathit{pas} \mapsto P, P^*$$

The formal definitions of the terms U, M, V, N, P, and P^* are summarized in table 3 where we make use of the following macro definitions and typing of constants:

$$\mathcal{M}(x,y,e) =_{def} (\mathtt{ma}\ e) \wedge (\mathtt{ag}\ e\ x) \wedge (\mathtt{th}\ e\ y) \quad \mathcal{N}(l) =_{def} \begin{cases} \lambda_{\mathtt{s}} x.\neg & \text{if } l = \epsilon \\ \lambda_{(\mathtt{so})} \vec{x}.\neg & \text{if } l = \mathtt{o} \end{cases}$$

$$\tau(\mathtt{ag}) = \tau(\mathtt{th}) = \nu \to \iota \to o \qquad \tau(\mathtt{ma}) = \tau(\mathtt{vi}) = \nu \to o \qquad \tau(\star) = \varsigma$$

The macro $\mathcal{N}(l)$ lifts the type of the constant $\neg$ introduced in remark 2.11; as mentioned in remark 2.10 it serves the purpose to match certain typing constraints and, further, takes into account both intransitive and transitive verbs. Note that for *pas* we have two different encodings: in $\tau(P^*)$ we have that $\tau(V)\backslash\forall$ stands for $\tau(V)$ without the second-order quantifier. Further, every term is well-typed: see for instance figure 3. We also have a collection of words encoded as constants: *Nicolas* $\mapsto$ $\mathtt{ni}$ with $\tau(\mathtt{ni}) = \iota$ and *pomme* $\mapsto$ $\mathtt{po}$ with $\tau(\mathtt{po}) = \iota \to o$.

Example 3.5 Consider the sentence *Nicolas mange vite une pomme* to which we associate the parse tree [[Nicolas [mange vite]][une pomme]]. Since the term V associated with *vite* is prefixed by a second-order quantifier we cannot form the application $(V\,{}^{\mathtt{l}}_{\mathtt{v}}\,M)$ directly. But, if $\widetilde{V}$ is shorthand for the term

$$(V\ A(\mathtt{o})) : (\iota \to^{\mathtt{l}}_{\mathtt{s}} \iota \to^{\mathtt{r}}_{\mathtt{o}} \kappa \to^{\mathtt{x}}_{\mathtt{k}} o) \to^{\mathtt{r}}_{\mathtt{v}} \iota \to^{\mathtt{l}}_{\mathtt{s}} \iota \to^{\mathtt{r}}_{\mathtt{o}} \kappa \to^{\mathtt{x}}_{\mathtt{k}} o \quad (A(l) = \iota \to^{\mathtt{r}}_{l} \kappa \to^{\mathtt{x}}_{\mathtt{k}} o)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash x{:}\,\iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)}}{\Gamma \vdash x{:}\,\kappa \to^{\mathtt{x}}_{\mathtt{k}} \iota \to^{\mathtt{l}}_{\mathtt{s}} X'(l)}\,(\sim) \qquad \Gamma \vdash \lambda e.[\alpha](\mathtt{vi}\ e) \wedge (c\ e){:}\,\kappa}}{x{:}\,\iota \to^{\mathtt{l}}_{\mathtt{s}} X(l), c{:}\,\kappa, \alpha{:}\,o^{\perp} \vdash (x^{\mathtt{x}}_{\mathtt{k}}\, \lambda e.[\alpha](\mathtt{vi}\ e) \wedge (c\ e)){:}\,\iota \to^{\mathtt{l}}_{\mathtt{s}} X'(l)}}{x{:}\,\iota \to^{\mathtt{l}}_{\mathtt{s}} X(l), c{:}\,\kappa \vdash \mu_{(\mathtt{sl})}\alpha.(x^{\mathtt{x}}_{\mathtt{k}}\, \lambda e.[\alpha](\mathtt{vi}\ e) \wedge (c\ e)){:}\,\iota \to^{\mathtt{l}}_{\mathtt{s}} X'(l)}\,(\perp = o)}}{x{:}\,\iota \to^{\mathtt{l}}_{\mathtt{s}} X(l) \vdash \lambda^{\mathtt{x}}_{\mathtt{k}} c.\mu_{(\mathtt{sl})}\alpha.(x^{\mathtt{x}}_{\mathtt{k}}\, \lambda e.[\alpha](\mathtt{vi}\ e) \wedge (c\ e)){:}\,\iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)}\,(\sim)}}{\emptyset \vdash \lambda^{\mathtt{l}}_{\mathtt{v}} x.\lambda^{\mathtt{x}}_{\mathtt{k}} c.\mu_{(\mathtt{sl})}\alpha.(x^{\mathtt{x}}_{\mathtt{k}}\, \lambda e.[\alpha](\mathtt{vi}\ e) \wedge (c\ e)){:}\,(\iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)) \to^{\mathtt{l}}_{\mathtt{v}} \iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)}}{\vdash_{s_2} \Lambda X(l).\lambda^{\mathtt{l}}_{\mathtt{v}} x.\lambda^{\mathtt{x}}_{\mathtt{k}} c.\mu_{(\mathtt{sl})}\alpha.(x^{\mathtt{x}}_{\mathtt{k}}\, \lambda e.[\alpha](\mathtt{vi}\ e) \wedge (c\ e)){:}\,\forall X(l).(\iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)) \to^{\mathtt{l}}_{\mathtt{v}} \iota \to^{\mathtt{l}}_{\mathtt{s}} X(l)}$$

Figure 3: Typing of V

then the term $S = ((\widetilde{V}\,{}^{\mathtt{l}}_{\mathtt{v}}\, M)\,{}^{\mathtt{l}}_{\mathtt{s}}\, \mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\, (U\,{}^{\mathtt{r}}_{\mathtt{n}}\, \mathtt{po}))$ is of type $\kappa \to^{\mathtt{x}}_{\mathtt{k}} o$ and, furthermore, matches the above parse tree. As shown in figure 4 the term S reduces to

$$\mu\alpha.\exists x.[(\mathtt{po}\ x) \wedge [\alpha](\lambda^{\mathtt{x}}_{\mathtt{k}} c.\mu\beta.\exists e.[\mathcal{M}(\mathtt{ni}, x, e) \wedge [\beta](\mathtt{vi}\ e) \wedge (c\ e)])]$$

and, if we first apply the continuation $\lambda x.\top$ to it and then erase the μ's with the ϵ-reduction, we obtain the following semantic reading or interpretation of the sentence: $\exists x.[(\mathtt{po}\ x) \wedge \exists e.[\mathcal{M}(\mathtt{ni}, x, e) \wedge (\mathtt{vi}\ e) \wedge \top]]$.

Example 3.6 Consider the sentence *Nicolas ne mange pas une pomme* to which we associate the parse tree [[Nicolas [ne [mange pas]]][une pomme]]. As above we have that the term $((\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\, (\widetilde{P}\,{}^{\mathtt{l}}_{\mathtt{v}}\, M))\,{}^{\mathtt{l}}_{\mathtt{s}}\, \mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\, (U\,{}^{\mathtt{r}}_{\mathtt{n}}\, \mathtt{po}))$ of type $\kappa \to^{\mathtt{x}}_{\mathtt{k}} o$ matches the parse tree and reduces as shown in figure 5. Again, by first applying the continuation $\lambda x.\top$ to the result of the reduction given in figure 5 and then erasing the μ's from it, we obtain the following semantic reading of the sentence: $\exists x.[(\mathtt{po}\ x) \wedge \neg\exists e.[\mathcal{M}(\mathtt{ni}, x, e) \wedge \top]]$.

Example 3.7 Consider the sentence *Nicolas ne mange pas vite une pomme* and its parse tree [[Nicolas [ne [[mange pas] vite]]][une pomme]]. As above we have that the term $((\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\, (\widetilde{P^*}\,{}^{\mathtt{l}}_{\mathtt{v}}\, M\,{}^{\mathtt{r}}_{\mathtt{a}}\, \widetilde{V}))\,{}^{\mathtt{l}}_{\mathtt{s}}\, \mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\, (\widetilde{U}\,{}^{\mathtt{r}}_{\mathtt{n}}\, \mathtt{po}))$ of type $\kappa \to^{\mathtt{x}}_{\mathtt{k}} o$ matches the parse tree and reduces as shown in figure 6. Again, by first applying the continuation $\lambda x.\top$ to the result of the reduction given in figure 6 and then erasing the μ's from it, we obtain the following semantic reading of the sentence: $\exists x.[(\mathtt{po}\ x) \wedge \exists e.[\mathcal{M}(\mathtt{ni}, x, e) \wedge \neg(\mathtt{vi}\ e) \wedge \top]]$.

Remark 3.8 We leave it to the reader to verify that the sentence from example 3.7 may be parsed as [[Nicolas [[[ne [mange pas]]] vite]][une pomme]], thus giving rise to the term $((\widetilde{V}\,{}^{\mathtt{l}}_{\mathtt{v}}\, (\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\, (\widetilde{P}\,{}^{\mathtt{l}}_{\mathtt{v}}\, M)))\,{}^{\mathtt{l}}_{\mathtt{s}}\, \mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\, (U\,{}^{\mathtt{r}}_{\mathtt{n}}\, \mathtt{po}))$ of type $\kappa \to^{\mathtt{x}}_{\mathtt{k}} o$ and hence yielding the semantic reading $\exists x.[(\mathtt{po}\ x) \wedge \neg\exists e.[\mathcal{M}(\mathtt{ni}, x, e) \wedge (\mathtt{vi}\ e) \wedge \top]]$.

Remark 3.9 Let us point out that the framework can be refined in such a way as to avoid certain semantic readings. For instance, if we would like to get

$$
\begin{array}{ll}
& ((\widetilde{V}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M)\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,\underline{(U\,{}^{\mathtt{r}}_{\mathtt{n}}\,\mathtt{po})}) \\
\rightarrow_{\beta_v} & \underline{((\widetilde{V}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M)\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,(\mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha]x]))} \\
\rightarrow_{\mu'} & \mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha]\underline{((\widetilde{V}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M)\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)}] \\
\rightarrow^{*} & \mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha](\lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu\beta.\exists e.[\mathcal{M}(\mathtt{ni},x,e)\wedge[\beta](\mathtt{vi}\ e)\wedge(c\ e)])]
\end{array}
$$

where the last reduction expands as follows:

$$
\begin{array}{ll}
& (\underline{(\widetilde{V}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M)}\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x) \\
\rightarrow_{\beta_v} & ((\lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu_{(\mathtt{so})}\beta.\underline{(M\,{}^{\mathtt{x}}_{\mathtt{k}}\,(\lambda e.[\beta](\mathtt{vi}\ e)\wedge(c\ e))))}\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x) \\
\rightarrow_{c^2_1,\beta_v} & ((\lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu_{(\mathtt{so})}\beta.\lambda^{\mathtt{l}}_{\mathtt{s}}y.\lambda^{\mathtt{r}}_{\mathtt{o}}z.\exists e.[\mathcal{M}(y,z,e)\wedge\underline{((\lambda e.[\beta](\mathtt{vi}\ e)\wedge(c\ e))\ e)}])\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x) \\
\rightarrow_{\beta_v} & \underline{((\lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu_{(\mathtt{so})}\beta.\lambda^{\mathtt{l}}_{\mathtt{s}}y.\lambda^{\mathtt{r}}_{\mathtt{o}}z.\exists e.[\mathcal{M}(y,z,e)\wedge[\beta](\mathtt{vi}\ e)\wedge(c\ e)])\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)} \\
\rightarrow_{c^2_1,c^2_2} & \lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu\beta.\underline{((\lambda^{\mathtt{l}}_{\mathtt{s}}y.\lambda^{\mathtt{r}}_{\mathtt{o}}z.\exists e.[\mathcal{M}(y,z,e)\wedge[\beta](\mathtt{vi}\ e)\wedge(c\ e)])\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)} \\
\rightarrow_{\beta^2_v} & \lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu\beta.\exists e.[\mathcal{M}(\mathtt{ni},x,e)\wedge[\beta](\mathtt{vi}\ e)\wedge(c\ e)]
\end{array}
$$

Figure 4: Reduction — adverb

only the semantic reading provided in example 3.7 then one should not be able to apply a negated verb such as $(\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\widetilde{P}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M))$ to an adverb such as $\widetilde{V}$. This can be achieved by adding parameters to the type ν that allow one to set a flag whenever a verb has been negated or modified by an adverb. The above situation can thus be ruled out by imposing the restriction that an adverbial modifier can never be applied to a negated verb. Similarly, by adding parameters to the type ι one can take care of morpho-syntactic features.

4 Conclusion

In this short note we have introduced an extension of the standard $\lambda\mu$-calculus and shown that, together with the tools advocated by Parsons in [Par90], one can actually extend de Groote's approach outlined in [dG01] in such a way as to deal also with adverbs and negation. With respect to de Groote's framework we have that the use of positions allows one to restore directionality and, more importantly, that the use of labels provides means to avoid the undesirable requirement of having a non-confluent calculus.

Nevertheless, there is still room for improvement in our framework:

- First, that we have to lift the type of a term in the μ' reduction seems to be a rather odd requirement. Example 3.7 shows that we cannot avoid it unless we give up the idea of having a generalized form of μ-abstraction. We could have avoided the problem altogether by omitting intransitive verbs but from both a practical and a theoretical point of view it is desirable to have a framework that can cope with transitive as well as intransitive

$$
\begin{array}{ll}
 & ((\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\widetilde{P}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M))\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,\underline{(U\,{}^{\mathtt{r}}_{\mathtt{n}}\,\mathtt{po})}) \\
\to_{\beta_v} & \underline{((\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\widetilde{P}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M))\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,(\mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha]x]))} \\
\to_{\mu'} & \mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha]((\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,\underline{(\widetilde{P}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M)})\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)] \\
\to_{\beta_v} & \mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha]((\underline{\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\lambda^{\mathtt{x}}_{*}y.\mu\beta.\neg[\beta]M)})\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)] \\
\to_{\beta_v} & \mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha](\underline{((\lambda^{\mathtt{x}}_{*}y.\mu\beta.\neg[\beta]M)\,{}^{\mathtt{x}}_{*}\,\star)}\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)] \\
\to_{\beta_v} & \mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha]\underline{((\mu\beta.\neg[\beta]M)\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)}] \\
\to_{\mu^2_v} & \mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha](\mu\beta.\neg[\beta]\underline{(M\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)})] \\
\to_{\beta^2_v} & \mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha](\mu\beta.\neg[\beta](\lambda^{\mathtt{x}}_{\mathtt{k}}c.\exists e.[\mathcal{M}(\mathtt{ni},x,e)\wedge(c\ e)]))]
\end{array}
$$

Figure 5: Reduction — negation

$$
\begin{array}{ll}
 & ((\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\widetilde{P^{*}}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M\,{}^{\mathtt{r}}_{\mathtt{a}}\,\widetilde{V}))\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,\underline{(U\,{}^{\mathtt{r}}_{\mathtt{n}}\,\mathtt{po})}) \\
\to_{\beta_v} & \underline{((\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\widetilde{P^{*}}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M\,{}^{\mathtt{r}}_{\mathtt{a}}\,\widetilde{V}))\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,(\mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha]x]))} \\
\to_{\mu'} & \mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha]((\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,\underline{(\widetilde{P^{*}}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M\,{}^{\mathtt{r}}_{\mathtt{a}}\,\widetilde{V})})\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)] \\
\to_{\beta^2_v} & \mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha]\underline{((\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\lambda^{\mathtt{x}}_{*}y.\lambda^{\mathtt{x}}_{\mathtt{k}}c.(\mathcal{N}(\mathtt{o})\ ((\widetilde{V}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M)\,{}^{\mathtt{x}}_{\mathtt{k}}\,c))))\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)}] \\
\to^{*} & \mu\alpha.\exists x.[(\mathtt{po}\ x)\wedge[\alpha](\lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu\beta.\exists e.[\mathcal{M}(\mathtt{ni},x,e)\wedge[\beta]\neg(\mathtt{vi}\ e)\wedge(c\ e)])]
\end{array}
$$

where the last reduction expands as follows:

$$
\begin{array}{ll}
\boxed{1} & (\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\lambda^{\mathtt{x}}_{*}y.\lambda^{\mathtt{x}}_{\mathtt{k}}c.(\mathcal{N}(\mathtt{o})\ \underline{((\widetilde{V}\,{}^{\mathtt{l}}_{\mathtt{v}}\,M)\,{}^{\mathtt{x}}_{\mathtt{k}}\,c)}))) \\
\to_{\beta_v} & (\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\lambda^{\mathtt{x}}_{*}y.\lambda^{\mathtt{x}}_{\mathtt{k}}c.(\mathcal{N}(\mathtt{o})\ ((\lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu_{(\mathtt{so})}\beta.\underline{(M\,{}^{\mathtt{x}}_{\mathtt{k}}\,(\lambda e.[\beta](\mathtt{vi}\ e)\wedge(c\ e))))}\,{}^{\mathtt{x}}_{\mathtt{k}}\,c)))) \\
\to_{c^2_1,\beta^2_v} & (\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\lambda^{\mathtt{x}}_{*}y.\lambda^{\mathtt{x}}_{\mathtt{k}}c.(\mathcal{N}(\mathtt{o})\ \underline{((\lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu_{(\mathtt{so})}\beta.\lambda^{\mathtt{l}}_{\mathtt{s}}z.\lambda^{\mathtt{r}}_{\mathtt{o}}u.\exists e.[\mathcal{M}(z,u,e)\wedge[\beta](\mathtt{vi}\ e)\wedge(c\ e)])\,{}^{\mathtt{x}}_{\mathtt{k}}\,c)}))) \\
\to_{\beta_v} & (\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\lambda^{\mathtt{x}}_{*}y.\lambda^{\mathtt{x}}_{\mathtt{k}}c.\underline{(\mathcal{N}(\mathtt{o})\ (\mu_{(\mathtt{so})}\beta.\lambda^{\mathtt{l}}_{\mathtt{s}}z.\lambda^{\mathtt{r}}_{\mathtt{o}}u.\exists e.[\mathcal{M}(z,u,e)\wedge[\beta](\mathtt{vi}\ e)\wedge(c\ e)])}))) \\
\to_{\mu'} & \underline{(\widetilde{N}\,{}^{\mathtt{r}}_{\mathtt{v}}\,(\lambda^{\mathtt{x}}_{*}y.\lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu_{(\mathtt{so})}\beta.\lambda^{\mathtt{l}}_{\mathtt{s}}z.\lambda^{\mathtt{r}}_{\mathtt{o}}u.\exists e.[\mathcal{M}(z,u,e)\wedge[\beta]\neg(\mathtt{vi}\ e)\wedge(c\ e)]))} \\
\to_{\beta_v} & \underline{((\lambda^{\mathtt{x}}_{*}y.\lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu_{(\mathtt{so})}\beta.\lambda^{\mathtt{l}}_{\mathtt{s}}z.\lambda^{\mathtt{r}}_{\mathtt{o}}u.\exists e.[\mathcal{M}(z,u,e)\wedge[\beta]\neg(\mathtt{vi}\ e)\wedge(c\ e)])\,{}^{\mathtt{x}}_{*}\,\star)} \\
\to_{\beta_v} & \lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu_{(\mathtt{so})}\beta.\lambda^{\mathtt{l}}_{\mathtt{s}}z.\lambda^{\mathtt{r}}_{\mathtt{o}}u.\exists e.[\mathcal{M}(z,u,e)\wedge[\beta]\neg(\mathtt{vi}\ e)\wedge(c\ e)] \\
\boxed{2} & \underline{((\lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu_{(\mathtt{so})}\beta.\lambda^{\mathtt{l}}_{\mathtt{s}}z.\lambda^{\mathtt{r}}_{\mathtt{o}}u.\exists e.[\mathcal{M}(z,u,e)\wedge[\beta]\neg(\mathtt{vi}\ e)\wedge(c\ e)])\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)} \\
\to_{c^2_1,c^2_2} & \lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu\beta.\underline{((\lambda^{\mathtt{l}}_{\mathtt{s}}z.\lambda^{\mathtt{r}}_{\mathtt{o}}u.\exists e.[\mathcal{M}(z,u,e)\wedge[\beta]\neg(\mathtt{vi}\ e)\wedge(c\ e)])\,{}^{\mathtt{l}}_{\mathtt{s}}\,\mathtt{ni}\,{}^{\mathtt{r}}_{\mathtt{o}}\,x)} \\
\to_{\beta_v} & \lambda^{\mathtt{x}}_{\mathtt{k}}c.\mu\beta.\exists e.[\mathcal{M}(\mathtt{ni},x,e)\wedge[\beta]\neg(\mathtt{vi}\ e)\wedge(c\ e)]
\end{array}
$$

Figure 6: Reduction — negated adverb

verbs without having to duplicate the encodings for adverbial modifiers.

However, the current rule probably makes the calculus incomplete if we want enough reductions to satisfy the sub-formula property (this needs to be clarified). There are two ways out of this problem : finding a more general μ' rule or trying to use a call-by-name strategy which means deep changes for all terms but allows to encode $\mu_w \alpha.M$ using λ and μ.

- Second, we would like to have a better control of the order of quantifiers in the semantic reading of a sentence and also the part of the sentence which should be negated. The general consensus is that this can be achieved by introducing delimited continuations. Section 3 of [HG08] provides a call-by-value variant of a $\lambda\mu$-calculus for delimited continuations that seems well suited for our purpose.

- Finally, the use of both λ and μ simplifies the terms associated to a lexical entry. However, it is still a lot of work to adjust all definitions together and the choice of an appropriate evaluation strategy is a very sensible issue. Moreover, a lot of linguistic features are considered in many papers and it would be interesting now to try to summarize all these works and produce a system that can accommodate a large set of linguistic phenomena with a more intelligible technique for producing the required terms.

References

[Bar02] C. Barker. Continuations and the nature of quantification. *Natural Language Semantics*, 10:211–242, 2002.

[Bar04] C. Barker. Continuations in natural language. In H. Thielecke, editor, *Proceedings of the 4th ACM SIGPLAN Continuation Workshop*, 2004.

[BM07] R. Bernardi and M. Moortgat. Continuation semantics for symmetric categorial grammar. In D. Leivant and R. de Queiroz, editors, *Proceedings of the 14th Workshop on Logic, Language, Information and Computation*, volume 4576 of *LNCS*, pages 53–71. Springer, 2007.

[Dav01] D. Davidson. *Essays on Actions and Events*. Oxford University Press, second edition, 2001.

[dG01] P. de Groote. Type raising, continuations, and classical logic. In R. van Roy and M. Stokhof, editors, *Proceedings of the 13th Amsterdam Colloquium*, pages 97–101, 2001.

[dG06] P. de Groote. Towards a Montagovian account of dynamics. In *Proceedings of Semantics and Linguistics Theory XVI*, 2006.

[HG08] H. Herbelin and S. Ghilezan. An approach to call-by-name delimited continuations. In G. C. Necula and P. Wadler, editors, *Proceedings of the 35th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 383–394, 2008.

[Mon73] R. Montague. The proper treatment of quantification in ordinary English. In J. Hintikka, J. Moravcsik, and P. Suppes, editors, *Approaches to Natural Language: Proceedings of the 1970 Stanford Workshop on Grammar and Semantics*, 1973.

[Par90] T. Parsons. *Events in the Semantics of English: A Study in Subatomic Semantics*. MIT Press, 1990.

[Py98] W. Py. *Confluence en $\lambda\mu$-calcul*. PhD thesis, University of Savoy, 1998.

[Sha04] C. Shan. Delimited continuations in natural language: quantification and polarity sensitivity. In H. Thielecke, editor, *Proceedings of the 4th ACM SIGPLAN Continuation Workshop*, 2004.