

# An Analysis of Selectional Restrictions with Dependent Type Semantics

Eriko Kinoshita<sup>1(✉)</sup>, Koji Mineshima<sup>1,2</sup>, and Daisuke Bekki<sup>1,2</sup>

<sup>1</sup> Ochanomizu University, Bunkyo, Japan

kinoshita.eriko@is.ocha.ac.jp

<sup>2</sup> CREST, Japan Science and Technology Agency, Kawaguchi, Japan

**Abstract.** Predicates in natural languages impose selectional restrictions on their arguments. In this paper, we analyze selectional restrictions of predicates within the framework of Dependent Type Semantics, a framework of natural language semantics based on dependent type theory. We also introduce operators that shift the meanings of predicates and analyze two phenomena, coercion and copredication for logical polysemous nouns, that present challenges to simple analysis of selectional restrictions.

## 1 Introduction

Predicates in natural languages impose selectional restrictions on their arguments. For example, the transitive verb *marry* expects its subject and object to be expressions that denote humans. Thus, from the utterance of (1), we can infer that Bob and Ann are both human.

(1) Bob married Ann.

One potential way to explain this inference is to treat selectional restrictions of predicates as entailment. According to this analysis, the verb *marry* is assigned the meaning in (2a) and the whole sentence in (1) has the interpretation in (2b).

- (2) a.  $\lambda y \lambda x. \mathbf{marry}(x, y) \wedge \mathbf{human}(x) \wedge \mathbf{human}(y)$   
b.  $\mathbf{marry}(\mathit{bob}, \mathit{ann}) \wedge \mathbf{human}(\mathit{bob}) \wedge \mathbf{human}(\mathit{ann})$

A problem with this analysis is that it cannot handle the inference in (3).

(3) Bob didn't marry Ann.  $\Rightarrow$  Bob and Ann are human.

From the negation of (1), one can also infer that Bob and Ann are human. If selectional restrictions of predicates were part of entailment, we would assign the interpretation (4) to the negative sentence in (3). This does not account for the inference in (3).

- (4)  $\neg(\mathbf{marry}(\mathit{bob}, \mathit{ann}) \wedge \mathbf{human}(\mathit{bob}) \wedge \mathbf{human}(\mathit{ann}))$

In general, the contents of selectional restrictions project out of the scope of negation, modals, and conditionals (Asher [2], Magidor [7]). This is a common feature of inferences known as *presupposition projection* (see, e.g., Beaver [3] for an overview).

The goal of this paper is to propose an analysis that treats selectional restrictions as presupposition within the framework of Dependent Type Semantics (DTS; Bekki [4], Bekki and Mineshima [5]). Using this framework, we also present a formal analysis of two lexical phenomena related to selectional restriction, namely, coercion and copredication for logical polysemy.

## 2 Selectional Restriction: Types vs. Predicates

Although the presuppositional analysis of selectional restriction goes back at least as far as McCawley [9], it seems fair to say that its precise formulation has been mostly neglected in the simply typed setting of standard formal semantics, where only  $e$  (entity) and  $t$  (truth-value) are taken as base types.

Recently, some proposals in the literature have suggested ways to handle selectional restrictions with extended type-theoretic frameworks (Asher [1], Luo [6], Retoré [11]). There are two possible approaches here. One is to represent selectional restrictions as *types*; as two examples, using **animate** and **human** as base types, one can assign a type **animate**  $\rightarrow$  **prop** to the predicate **cry** and **human**  $\rightarrow$  **human**  $\rightarrow$  **prop** to the predicate **marry**. According to this approach, violation of a selectional restriction is to be treated as a type mismatch. One problem with this approach is the problem of *subtyping*. That is, to combine the predicate **cry** of type **animate**  $\rightarrow$  **prop** with the term **john** of type **human**, one needs a subtyping relation **human**  $<$  **animate** and extra subtyping rules (cf. Luo [6], Retoré [11]). One drawback is that with additional subtyping rules, the resulting compositional semantics becomes complicated.

Alternatively, one can preserve the base type for entities and represent selectional restrictions as *predicates* over entities. This view seems to be underdeveloped, but it has the advantage that it can dispense with subtyping and preserve the clear, well-understood conception of syntax-semantics mapping. Our theory is based on this second approach.

## 3 Dependent Type Semantics

The main challenge here is how to provide a presuppositional analysis of selectional restrictions combined with the selectional-restriction-as-predicate view. We use DTS (Bekki [4], Bekki and Mineshima [5]) as a theoretical framework, which provides two crucial tools: *dependent types* (which are a generalization of simple types) and *underspecified terms*. DTS is a proof-theoretic semantics of natural language based on dependent type theory (Martin-Löf [8]). It characterizes the meaning of a sentence from the perspective of *inferences*.

DTS uses two kinds of dependent types.

- (i)  $\Pi$ -type (dependent function type), written as  $(x : A) \rightarrow B$ , is a generalized form of a function type  $A \rightarrow B$ ; a term of type  $(x : A) \rightarrow B$  is a function  $f$  that takes a term  $a$  of type  $A$  and returns a term  $f(a)$  of type  $B(a)$ .
- (ii)  $\Sigma$ -type (dependent product type), written as  $(x : A) \times B$  or  $\left[ \begin{smallmatrix} x : A \\ B \end{smallmatrix} \right]$ , is a generalized form of a product type  $A \times B$ ; a term of type  $(x : A) \times B$  is a pair  $(t, u)$  such that  $t$  is of type  $A$  and  $u$  is of type  $B(t)$ . The projection operators  $\pi_1$  and  $\pi_2$  are defined in such a way that  $\pi_1(t, u) = t$  and  $\pi_2(t, u) = u$ .

Under the so-called propositions-as-types principle (Martin-Löf [8]), types and propositions are identified; a term  $t$  having type  $A$  (i.e.,  $t : A$ ) serves as a *proof term* for the proposition  $A$ .

In the dependently typed setting,  $\Pi$ -type and  $\Sigma$ -type correspond to universal and existential quantifiers, respectively. For example, in DTS, the sentence in (5a) is given the semantic representation (SR) in (5b):

- (5) a. Every man entered.  
 b.  $\left( u : \left[ \begin{smallmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{smallmatrix} \right] \right) \rightarrow \mathbf{enter}(\pi_1(u))$

The term  $u$  here has a  $\Sigma$ -type: it consists of a term (let it be  $x$ ) having type **entity** and some proof term having type **man**( $x$ ) that depends on  $x$ . The term  $\pi_1(u)$  in **enter**( $\pi_1(u)$ ) picks up the entity that is the first component of  $u$ . In DTS, common nouns such as *man* are treated as predicates rather than as types. In other words, that a term  $x$  has a property *man* is represented as a proposition **man**( $x$ ), rather than as a judgement  $x : \mathbf{man}$ . See Bekki and Mineshima [5] for more discussions on the interpretation of common nouns in our framework.

For  $\Pi$ -types and  $\Sigma$ -types, we use the following formation rules ( $\Pi F$ ,  $\Sigma F$ ), introduction rules ( $\Pi I$ ,  $\Sigma I$ ), and elimination rules ( $\Pi E$ ,  $\Sigma E$ ).

$$\begin{array}{c}
 \frac{\overline{x : A}^{(i)}}{A : s_1 \quad B : s_2} (\Pi F), i \\
 \\
 \frac{\overline{x : A}^{(i)}}{(x : A) \rightarrow B : s \quad M : B} (\Pi I), i \\
 \\
 \frac{M : (x : A) \rightarrow B \quad N : A}{MN : B[N/x]} (\Pi E)
 \end{array}
 \qquad
 \begin{array}{c}
 \frac{\overline{x : A}^{(i)}}{A : \mathbf{type} \quad B : s} (\Sigma F), i \\
 \\
 \frac{M : A \quad N : B[M/x]}{(M, N) : \left[ \begin{smallmatrix} x : A \\ B(x) \end{smallmatrix} \right] : s} (\Sigma I) \\
 \\
 \frac{M : \left[ \begin{smallmatrix} x : A \\ B(x) \end{smallmatrix} \right]}{\pi_1(M) : A} (\Sigma E)
 \end{array}
 \qquad
 \frac{M : \left[ \begin{smallmatrix} x : A \\ B(x) \end{smallmatrix} \right]}{\pi_2(M) : B[\pi_1(M)/x]} (\Sigma E)$$

Here,  $s$ ,  $s_1$  and  $s_2$  are kind or type (see Bekki and Mineshima [5] for more details).

DTS has an underspecified term  $@$  to handle anaphora and presupposition. We use type annotation for underspecified terms; we write  $@ : A$ , where the underspecified term  $@$  is annotated with its type  $A$ . By using underspecified

terms, we can uniformly handle semantic phenomena that depend on the preceding contexts.

Presupposition and anaphora are resolved by constructing a proof term for  $@ : A$  with type checking and then replacing  $@ : A$  by the constructed term. Type checking ensures that an SR is well-formed (i.e., having type **type**). For underspecified terms, we use the rule

$$\frac{A : s \quad A \text{ true}}{(@ : A) : A} (@)$$

where  $s \in \{\text{kind}, \text{type}\}$ . The judgement  $A \text{ true}$  triggers a proof search to construct a term having the type  $A$  in a given context. The constructed term is to be replaced with  $@$  in the final representation. The annotated type  $A$  may contain another underspecified term, for which the type checking is triggered by the judgement  $A : s$  (e.g.,  $A : \text{type}$ ) in the  $@$  rule.

As an illustration, consider the sentence in (6a). For this sentence, one can compositionally derive the SR in (6b).<sup>1</sup>

(6) a. He whistled.

b. **whistle**  $\left( \pi_1 \left( @ : \left[ \begin{array}{l} x : \text{entity} \\ \text{man}(x) \end{array} \right] \right) \right)$

The SR (6b) contains an underspecified term  $@$  annotated with the  $\Sigma$ -type corresponding to the proposition that there is an entity  $x$  such that  $x$  is a man. For the SR in (6b), the type checking runs as follows.

$$\frac{\frac{\frac{\text{entity} : \text{type}}{\text{entity} : \text{type}} (CON) \quad \frac{\frac{\text{man} : \text{entity} \rightarrow \text{type}}{\text{man}(x) : \text{type}} (CON) \quad \frac{x : \text{entity}}{x : \text{entity}} (IE)}{\text{man}(x) : \text{type}} (\Sigma F), 1 \quad \vdots \quad \frac{\left[ \begin{array}{l} x : \text{entity} \\ \text{man}(x) \end{array} \right] : \text{type}}{\left[ \begin{array}{l} x : \text{entity} \\ \text{man}(x) \end{array} \right] \text{ true}} (@)}{\frac{\left( @ : \left[ \begin{array}{l} x : \text{entity} \\ \text{man}(x) \end{array} \right] \right) : \left[ \begin{array}{l} x : \text{entity} \\ \text{man}(x) \end{array} \right]}{\pi_1 \left( @ : \left[ \begin{array}{l} x : \text{entity} \\ \text{man}(x) \end{array} \right] \right) : \text{entity}} (\Sigma E)} (\text{IE})$$

The application of the  $@$  rule in this derivation triggers a proof search for the judgement:

$$\left[ \begin{array}{l} x : \text{entity} \\ \text{man}(x) \end{array} \right] \text{ true}.$$

Assuming that we have  $john : \text{entity}$  and  $t : \text{man}(john)$  in the background global context, we can construct a term  $(john, t)$  having the  $\Sigma$ -type in question, i.e., a type annotated for the underspecified term  $@$ . This term serves as an antecedent of the pronoun *he*. Replacing  $@$  with the specific term  $(john, t)$ , the semantic representation in (6b) ends up with **whistle** $(\pi_1(john, t))$ , which reduces to **whistle** $(john)$ . In this way, we can derive the interpretation for the sentence containing a pronoun in (6a).

<sup>1</sup> See Bekki [4] for details on the compositional derivations of SRs in DTS.

## 4 Selectional Restriction in DTS

To handle selectional restrictions of predicates as presuppositions, we need to calculate whether selectional restrictions are satisfied at the stage of type checking. We propose that selectional restrictions of predicates are specified in the lexicon. For instance, we can define lexical entries of intransitive and transitive verbs as follows.

	syntax	semantic representation
<i>cry</i>	$S \backslash NP$	$\lambda x. \mathbf{cry}(x, @ : \mathbf{animate}(x))$
<i>marry</i>	$(S \backslash NP) / NP$	$\lambda y. \lambda x. \mathbf{marry}(y, @_i : \mathbf{human}(y))(x, @_j : \mathbf{human}(x))$

To be concrete, we use Combinatory Categorial Grammar (CCG; Steedman [12]) as a syntactic framework. The types of the predicates **cry** and **marry** in the above SRs are defined as follows.

$$\begin{aligned} \mathbf{cry} &: \left[ \begin{array}{l} x : \mathbf{entity} \\ \mathbf{animate}(x) \end{array} \right] \rightarrow \text{type} \\ \mathbf{marry} &: \left[ \begin{array}{l} y : \mathbf{entity} \\ \mathbf{human}(y) \end{array} \right] \rightarrow \left[ \begin{array}{l} x : \mathbf{entity} \\ \mathbf{human}(x) \end{array} \right] \rightarrow \text{type} \end{aligned}$$

For example, the predicate **cry** takes a pair consisting of an entity  $x$  and a proof term for the proposition **animate**( $x$ ) as an argument and returns a type (as a proposition). In the lexical entry for the intransitive verb *cry*, the proof term for the proposition **animate**( $x$ ) is underspecified; given that there is an underspecified term  $@ : \mathbf{animate}(x)$  in the SR, we have to prove **animate**( $x$ ) during the stage of type checking in order to ensure that the subject of *cry* is animate.

As an illustration, consider the sentence in (1). For this sentence, we can derive the following SR in a compositional way.

$$(7) \quad \mathbf{marry}(ann, @_1 : \mathbf{human}(ann))(bob, @_2 : \mathbf{human}(bob))$$

The following is the compositional derivation of this SR.

$$\frac{\frac{\frac{\text{Bob}}{NP} : bob}{S \backslash NP} : \lambda x. \mathbf{marry}(ann, @_1 : \mathbf{human}(ann))(x, @_2 : \mathbf{human}(x))}{\frac{\frac{\frac{\text{married}}{(S \backslash NP) / NP} : \lambda y. \lambda x. \mathbf{marry}(x, @_1 : \mathbf{human}(x))(y, @_2 : \mathbf{human}(y))}{\frac{\text{Ann}}{NP} : ann}}{>}} : \mathbf{marry}(ann, @_1 : \mathbf{human}(ann))(bob, @_2 : \mathbf{human}(bob)) <$$

Now, type checking to ensure that the SR in (7) is well-formed runs as follows.

$$\begin{array}{c}
 \frac{\text{marry} : \left[ \begin{array}{c} x : \mathbf{e} \\ \mathbf{h}(x) \end{array} \right] \rightarrow \left[ \begin{array}{c} y : \mathbf{e} \\ \mathbf{h}(y) \end{array} \right] \rightarrow \text{type}}{\text{marry}(a, @_1 : \mathbf{h}(a)) : \left[ \begin{array}{c} y : \mathbf{e} \\ \mathbf{h}(y) \end{array} \right] \rightarrow \text{type}} \quad (Con) \quad \frac{\overline{a : \mathbf{e}} \quad (Con) \quad (\textcircled{a}_1 : \mathbf{h}(a)) : \mathbf{h}(a) \quad (\Sigma I)}{(a, @_1 : \mathbf{h}(a)) : \left[ \begin{array}{c} x : \mathbf{e} \\ \mathbf{h}(x) \end{array} \right]} \quad (\Pi E) \quad \frac{\overline{b : \mathbf{e}} \quad (Con) \quad (\textcircled{a}_2 : \mathbf{h}(b)) : \mathbf{h}(b) \quad (\Sigma I)}{(b, @_2 : \mathbf{h}(b)) : \left[ \begin{array}{c} x : \mathbf{e} \\ \mathbf{h}(x) \end{array} \right]} \quad (\Pi E) \\
 \hline
 \text{marry}(a, @_1 : \mathbf{h}(a))(b, @_2 : \mathbf{h}(b)) : \text{type}
 \end{array}$$

Here, we abbreviate **entity** as **e**, **human** as **h**, *ann* as *a*, and *bob* as *b*. There are two open branches containing underspecified terms,  $@_1$  and  $@_2$ , which show that we must search the preceding context to construct proof terms for **human**(*bob*) and **human**(*ann*). That is to say, for the semantic representation to be well-formed, it is presupposed that *x* and *y*, which are, respectively, the subject and the object of the verb *marry*, are both human. In this way, the selectional restriction of a predicate is derived as a presupposition.

Similarly, the SR of the negative sentence in (3) is given as follows.

$$(8) \quad \neg \text{marry}(ann, @_2 : \mathbf{human}(ann))(bob, @_1 : \mathbf{human}(bob))$$

According to the formation rule of negation,  $A$  and  $\neg A$  have the same well-formedness condition.

$$\frac{A : \text{type}}{\neg A : \text{type}} \quad (\neg F)$$

That is, if we have  $A : \text{type}$ , then we have  $\neg A : \text{type}$  as well. Therefore, the type checking for the negative SR in (8) ends up with the derivation that triggers a proof search in the same way as the type checking for the SR in (6b) given in Sect. 3. In this way, one can derive the inference pattern of presupposition projection out of the scope of negation. A similar explanation applies to the case of modals and conditionals.

Interestingly, a negative sentence of the form in (9) has two readings (cf. McCawley [9]).

$$(9) \quad \text{The chair does not cry.}$$

First, this sentence has a reading in which the selectional restriction projects out of the scope of negation, hence resulting in a violation of selectional restriction. In our terms, after composing the meaning of (9), one obtains the SR  $\neg \text{cry}(\text{chair}, @ : \mathbf{animate}(\text{chair}))$ ; according to the formation rule of negation, the content of selectional restriction, that is, **animate**(*chair*), projects out of the scope of negation. Thus, for the SR to be well-formed, one needs to construct a proof term of **animate**(*chair*), which is not available in the standard context. Hence, it is predicted that under this reading, a violation of selectional restriction occurs in the sense that the derived SR is not well-formed.

Second, and more interesting, (9) can have a reading in which the selectional restriction does not project and is therefore interpreted inside the scope of negation. The presuppositional analysis correctly predicts this reading; by local accommodation, we can derive the SR  $\neg(\mathbf{animate}(\text{chair}) \wedge \text{cry}(\text{chair}))$

for (9). In this case, one does not have to construct a proof of **animate**(*chair*); hence, it is correctly predicted that under this reading, the utterance of (9) is meaningful and can be true. A detailed explanation of local accommodation in the framework of DTS is beyond the scope of this paper.

## 5 Coercion and Copredication for Logical Polysemy

### 5.1 Coercion

There are two phenomena that are not explained by a simple analysis of selectional restrictions of predicates. The first one is coercion (Nunberg [10]). For example, if we have a context in which there is a man who ate the omelet in a cafe, we can understand the meaning of (10a) as (10b).

- (10) a. The omelet escaped.  
b. The man who ate the omelet escaped.

To account for this phenomena, we define an operator, called *argument operator*, that transforms one predicate into another. The argument operators  $arg_1$  for a one-place predicate and  $arg_2$  for a two-place predicate are defined as follows.

$$arg_1 \equiv \lambda P. \lambda x. P \left( \pi_1 \pi_1 \left( @_5 : \left[ \begin{array}{c} z : \left[ \begin{array}{c} x : \mathbf{e} \\ @_2^{pr}(x) \end{array} \right] \\ (@_4 : \left[ \begin{array}{c} x : \mathbf{e} \\ @_1^{pr}(x) \end{array} \right] \rightarrow \left[ \begin{array}{c} x : \mathbf{e} \\ @_2^{pr}(x) \end{array} \right] \rightarrow \mathbf{type}) (x, (@_3^{pr}(x))) (z) \end{array} \right] \right) \right) \right)$$

$$arg_2 \equiv \lambda P. \lambda y. \lambda x. P \left( \pi_1 \pi_1 \left( @_7 : \left[ \begin{array}{c} z : \left[ \begin{array}{c} x : \mathbf{e} \\ @_3^{pr}(x) \end{array} \right] \\ (@_6 : \left[ \begin{array}{c} x : \mathbf{e} \\ @_1^{pr}(x) \end{array} \right] \rightarrow \left[ \begin{array}{c} x : \mathbf{e} \\ @_2^{pr}(x) \end{array} \right] \rightarrow \left[ \begin{array}{c} x : \mathbf{e} \\ @_3^{pr}(x) \end{array} \right] \rightarrow \mathbf{type}) (y, (@_4 : @_1^{pr}(y))) (x, (@_5 : @_2^{pr}(x))) (z) \end{array} \right] \right) \right) \right) (x)$$

Here an underspecified term  $@_i^{pr}$  is an abbreviation for  $@_i : \mathbf{e} \rightarrow \mathbf{type}$ .

Let us first focus on the definition of the argument operator  $arg_1$  for one-place predicates. In the definition of  $arg_1$ , the underspecified terms  $@_1$  and  $@_2$  in  $@_1^{pr}$  and  $@_2^{pr}$  are annotated with type  $\mathbf{e} \rightarrow \mathbf{type}$ ; these are underspecified terms for properties. Intuitively, given a one-place predicate  $P$  and its argument  $x$  of type  $\mathbf{e}$ , the argument operator  $arg_1$  produces a new predicate  $P'$  that existentially introduces a new entity  $z$  having some relation  $R$  to  $x$ .

When one underspecified term appears inside a type annotated with another underspecified term, the inside term must be resolved first. Specifically, the underspecified terms contained in the argument operator  $arg_1$  are resolved in the following way.

1. First, given the entity  $x$  (e.g., *the omelet* in (10a)), find a suitable property  $F$  (e.g., *edible*) holding for  $x$ . This property  $F$  replaces  $@_1^{pr}$ .
2. Second, if there is a proof term for the proposition that  $x$  has the property  $F$  (e.g., *the omelet is edible*), it replaces  $@_3^{pr}$ .

3. Also, a property  $G$  to be substituted for  $@_2^{pr}$  is needed. The property  $G$  (e.g., *animate*) has to be chosen so that the newly introduced entity (the first element of the term  $z$ ) satisfies  $G$ .
4. Next, find a relation  $R$  that is to be substituted for  $@_4^{pr}$ . In our example, a relation (e.g., *eat*) that has selectional restrictions specified by predicates **edible**( $x$ ) and **animate**( $y$ ) is needed. This relation  $R$  replaces  $@_4$ .
5. Finally, construct a term to be substituted for  $@_5$ . This is a tuple consisting of an entity  $z$  whose first element satisfies the property  $G$  and a proof term for the proposition that the relation  $R$  holds between  $x$  and  $z$ .

In this way,  $arg_1$  transforms the predicate **escape** into a predicate whose argument is an animate entity that has the eating relation to the omelet.

Let us explain the derivation in more detail. To begin with, we can derive the SR of the sentence (10a) as follows.

$$\begin{array}{c}
 \frac{\frac{\text{escaped}}{S \setminus NP} \quad \frac{\epsilon}{(S \setminus NP) \setminus (S \setminus NP)}}{\frac{: \lambda x. \mathbf{escape}(x, @_5 : \mathbf{animate}(x)) \quad : arg_1}{S \setminus NP}} < \\
 \frac{\frac{\text{The omelet}}{NP} \quad : o \quad : arg_1(\lambda x. \mathbf{escape}(x, @_6 : \mathbf{animate}(x)))}{S} < \\
 : arg_1(\lambda x. \mathbf{escape}(x, @_6 : \mathbf{animate}(x)))(o)
 \end{array}$$

By unfolding the definition of  $arg_1$ , the sentence in (10a) is assigned the SR in (11).

$$(11) \quad \mathbf{escape}(Z_1, @_6 : \mathbf{animate}(Z_1))$$

Here,  $Z_1$  abbreviates

$$\pi_1 \pi_1 \left( @_5 : \left[ z : \left[ \begin{array}{c} x : \mathbf{e} \\ @_2^{pr}(x) \end{array} \right] \right. \right. \left. \left. \left( @_4 : \left[ \begin{array}{c} x : \mathbf{e} \\ @_1^{pr}(x) \end{array} \right] \rightarrow \left[ \begin{array}{c} x : \mathbf{e} \\ @_2^{pr}(x) \end{array} \right] \rightarrow \mathbf{type} \right)(o, (@_3 : @_1^{pr}(o)))(z) \right] \right)$$

Let us suppose that we have the following information in the global context  $\mathcal{K}_1$ :

$$\begin{aligned}
 \mathcal{K}_1 &\equiv \text{type} : \text{kind}, \mathbf{e} : \text{type}, \\
 &j : \mathbf{e}, o : \mathbf{e}, \\
 &\mathbf{animate} : \mathbf{e} \rightarrow \text{type}, \mathbf{edible} : \mathbf{e} \rightarrow \text{type}, \\
 &\mathbf{eat} : \left[ \begin{array}{c} y : \mathbf{e} \\ \mathbf{edible}(y) \end{array} \right] \rightarrow \left[ \begin{array}{c} x : \mathbf{e} \\ \mathbf{animate}(x) \end{array} \right] \rightarrow \text{type}, \\
 &\mathbf{escape} : \left[ \begin{array}{c} x : \mathbf{e} \\ \mathbf{animate}(x) \end{array} \right] \rightarrow \text{type}, \\
 &p_1 : \mathbf{animate}(j), p_2 : \mathbf{edible}(o), p_3 : \mathbf{eat}(o, p_2)(j, p_1).
 \end{aligned}$$

Now type checking is triggered to determine whether the SR (10) is well-formed. This is an example of nested presupposition, and underspecified terms are resolved outward from the most deeply embedded. Here, we focus on the step to find a relation  $R$  that is substituted for the following underspecified term:

$$\mathbb{Q}_4 : \left[ \frac{x : \mathbf{e}}{\mathbb{Q}_1^{pr}(x)} \right] \rightarrow \left[ \frac{x : \mathbf{e}}{\mathbb{Q}_2^{pr}(x)} \right] \rightarrow \mathbf{type}.$$

The type checking tree for the relevant part looks as follows:

$$\frac{\frac{\mathcal{D}_1 \quad \mathcal{D}_2}{\left[ \frac{x : \mathbf{e}}{\mathbb{Q}_1^{pr}(x)} \right] : \mathbf{t} \quad \left[ \frac{x : \mathbf{e}}{\mathbb{Q}_2^{pr}(x)} \right] \rightarrow \mathbf{t} : \mathbf{k}} \quad (PIF) \quad \vdots}{\left[ \frac{x : \mathbf{e}}{\mathbb{Q}_1^{pr}(x)} \right] \rightarrow \left[ \frac{x : \mathbf{e}}{\mathbb{Q}_2^{pr}(x)} \right] \rightarrow \mathbf{t} : \mathbf{k}} \quad (\mathbb{Q})$$

$$\frac{\left( \mathbb{Q}_4 : \left[ \frac{x : \mathbf{e}}{\mathbb{Q}_1^{pr}(x)} \right] \rightarrow \left[ \frac{x : \mathbf{e}}{\mathbb{Q}_2^{pr}(x)} \right] \rightarrow \mathbf{t} \right) : \left[ \frac{x : \mathbf{e}}{\mathbb{Q}_1^{pr}(x)} \right] \rightarrow \left[ \frac{x : \mathbf{e}}{\mathbb{Q}_2^{pr}(x)} \right] \rightarrow \mathbf{t}}{\quad} \quad (\mathbb{Q})$$

where we use the abbreviations  $\mathbf{k}$  for kind and  $\mathbf{t}$  for **type**. The type checking for  $\mathcal{D}_1$  runs as follows.

$$\frac{\frac{\overline{\mathbf{e} : \mathbf{t}} \quad (CON) \quad \overline{\mathbf{t} : \mathbf{k}} \quad (CON) \quad \vdots}{\mathbf{e} \rightarrow \mathbf{t} : \mathbf{k}} \quad (PIF) \quad \mathbf{e} \rightarrow \mathbf{t} \text{ true} \quad (\mathbb{Q}) \quad \frac{\overline{x : \mathbf{e}} \quad (1)}{\mathbb{Q}_1^{pr} : \mathbf{e} \rightarrow \mathbf{t}} \quad (PIE)}{\frac{\overline{\mathbf{e} : \mathbf{t}} \quad (CON) \quad \mathbb{Q}_1^{pr} : \mathbf{e} \rightarrow \mathbf{t}}{\left[ \frac{x : \mathbf{e}}{\mathbb{Q}_1^{pr}(x)} \right] : \mathbf{t}} \quad (\Sigma F), 1}$$

Similarly, the type checking for  $\mathcal{D}_2$  runs as follows.

$$\frac{\frac{\overline{\mathbf{e} : \mathbf{t}} \quad (CON) \quad \overline{\mathbf{t} : \mathbf{k}} \quad (CON) \quad \vdots}{\mathbf{e} \rightarrow \mathbf{t} : \mathbf{k}} \quad (PIF) \quad \mathbf{e} \rightarrow \mathbf{t} \text{ true} \quad (\mathbb{Q}) \quad \frac{\overline{x : \mathbf{e}} \quad (1)}{\mathbb{Q}_2^{pr} : \mathbf{e} \rightarrow \mathbf{t}} \quad (PIE)}{\frac{\overline{\mathbf{e} : \mathbf{t}} \quad (CON) \quad \mathbb{Q}_2^{pr} : \mathbf{e} \rightarrow \mathbf{t}}{\left[ \frac{x : \mathbf{e}}{\mathbb{Q}_2^{pr}(x)} \right] : \mathbf{t}} \quad (\Sigma F), 1}$$

$$\frac{\left[ \frac{x : \mathbf{e}}{\mathbb{Q}_2^{pr}(x)} \right] : \mathbf{t} \quad \overline{\mathbf{t} : \mathbf{k}} \quad (CON)}{\left[ \frac{x : \mathbf{e}}{\mathbb{Q}_2^{pr}(x)} \right] \rightarrow \mathbf{t} : \mathbf{k}} \quad (PIF)$$

The judgements  $\mathbf{e} \rightarrow \mathbf{t} \text{ true}$  in  $\mathcal{D}_1$  and  $\mathcal{D}_2$  trigger a proof search; given a suitable global context, we can find the antecedents **edible** of type  $\mathbf{e} \rightarrow \mathbf{t}$  for  $\mathbb{Q}_1^{sr}$ , and **animate** of type  $\mathbf{e} \rightarrow \mathbf{t}$  for  $\mathbb{Q}_2^{sr}$ . Replacing each underspecified term with its antecedent predicate, the above type checking tree is transformed as follows.

$$\frac{\mathcal{D}_1 \quad \mathcal{D}_2}{\left[ \frac{x : \mathbf{e}}{\text{edible}(x)} \right] : \mathbf{t} \quad \left[ \frac{x : \mathbf{e}}{\text{animate}(x)} \right] \rightarrow \mathbf{t} : \mathbf{k}} \quad (PIF) \quad \vdots}{\left[ \frac{x : \mathbf{e}}{\text{edible}(x)} \right] \rightarrow \left[ \frac{x : \mathbf{e}}{\text{animate}(x)} \right] \rightarrow \mathbf{t} : \mathbf{k} \quad \left[ \frac{x : \mathbf{e}}{\text{edible}(x)} \right] \rightarrow \left[ \frac{x : \mathbf{e}}{\text{animate}(x)} \right] \rightarrow \mathbf{t} \text{ true} \quad (\mathbb{Q})}$$

$$\frac{\left( \mathbb{Q}_4 : \left[ \frac{x : \mathbf{e}}{\text{edible}(x)} \right] \rightarrow \left[ \frac{x : \mathbf{e}}{\text{animate}(x)} \right] \rightarrow \mathbf{t} \right) : \left[ \frac{x : \mathbf{e}}{\text{edible}(x)} \right] \rightarrow \left[ \frac{x : \mathbf{e}}{\text{animate}(x)} \right] \rightarrow \mathbf{t}}{\quad} \quad (\mathbb{Q})$$

Then, we can find an antecedent **eat** for @<sub>4</sub> that has a type

$$\left[ \begin{array}{l} x : \mathbf{e} \\ \mathbf{edible}(x) \end{array} \right] \rightarrow \left[ \begin{array}{l} x : \mathbf{e} \\ \mathbf{animate}(x) \end{array} \right] \rightarrow \mathbf{t}$$

in the context  $\mathcal{K}_1$ . In a similar way, we can find a proof term for other @-terms:  $p_2$  for @<sub>3</sub>,  $((j, p_1), p_3)$  for @<sub>5</sub>, and  $p_1$  for @<sub>6</sub>. By eliminating each @-term in (11) and reducing  $\beta$ -redexes, we obtain the SR **espace**( $j, p_1$ ) as a fully specified semantic representation for the sentence (10a).

## 5.2 Copredication for Logical Polysemy

The second phenomenon we consider is copredication of logically polysemous nouns. There are nouns having multiple meanings in natural language; the occurrences can be classified into accidental and logical polysemy (Asher [1]). For example, the noun *bank* in (12a) is accidentally polysemous, and the noun *book* in (12b) is logically polysemous.

- (12) a. # The bank is closed and is muddy.  
b. Mary memorized and burned the book.

The sentence (12b) shows that the logically polysemous noun *book* allows copredication, despite the fact that *memorize* and *burn* require different objects (i.e., informational objects and physical objects, respectively) as their object argument. To account for this fact, we can apply argument operators to the verbs *memorize* and *burn*, thereby avoiding the violation of selection restrictions.

We introduce the logical polysemies of nouns as functions. For example, we assign the following functions to the noun *book*.

$$\mathbf{book}_{\mathbf{infoOf}} : (x : \mathbf{e}) \rightarrow (\mathbf{book}(x) \rightarrow \left[ \begin{array}{l} y : \mathbf{e} \\ \mathbf{infoOf}(x)(y) \end{array} \right])$$

$$\mathbf{book}_{\mathbf{phyObjOf}} : (x : \mathbf{e}) \rightarrow (\mathbf{book}(x) \rightarrow \left[ \begin{array}{l} y : \mathbf{e} \\ \mathbf{phyObjOf}(x)(y) \end{array} \right])$$

The function **book<sub>infoOf</sub>** (resp., **book<sub>phyObjOf</sub>**) takes an entity  $x$  and a proof of **book**( $x$ ) and returns an entity  $y$  that is the informational aspect (resp., the physical aspect) of  $x$ .

Now we can derive the SR of the sentence (12b) as follows.

$$\begin{array}{c}
\frac{\text{memorized}}{S \backslash NP / NP} \quad \frac{\epsilon}{(S \backslash NP / NP) \backslash (S \backslash NP / NP)} \\
: MEM \quad : arg_2 \\
\hline
S \backslash NP / NP \\
: arg_2(MEM)
\end{array}
<
\frac{\text{and}}{CONJ}
\frac{\text{burned}}{S \backslash NP / NP} \quad \frac{\epsilon}{(S \backslash NP / NP) \backslash (S \backslash NP / NP)} \\
: BUR N \quad : arg_2 \\
\hline
S \backslash NP / NP \\
: arg_2(BUR N)
\end{array}
<$$

$$\frac{\text{and}}{CONJ}
\frac{\text{burned}}{S \backslash NP / NP} \quad \frac{\epsilon}{(S \backslash NP / NP) \backslash (S \backslash NP / NP)} \\
: \lambda p. \lambda q. \lambda y. \lambda x. \left[ \begin{array}{l} p(y)(x) \\ q(y)(x) \end{array} \right]$$

$$\frac{S \backslash NP / NP}{: \lambda y. \lambda x. \left[ \begin{array}{l} arg_2(MEM)(y)(x) \\ arg_2(BUR N)(y)(x) \end{array} \right]} \quad \langle \Phi \rangle$$

$$\frac{\text{the book}}{NP} \\
: b$$

$$\frac{\text{Mary}}{NP} \\
: m$$

$$\frac{S \backslash NP}{: \lambda x. \left[ \begin{array}{l} arg_2(MEM)(b)(x) \\ arg_2(BUR N)(b)(x) \end{array} \right]}$$

$$\frac{S}{: \left[ \begin{array}{l} arg_2(MEM)(b)(m) \\ arg_2(BUR N)(b)(m) \end{array} \right]} <$$

where

$$MEM \equiv \lambda y. \lambda x. \text{memorize}(y, @_i : \left[ \begin{array}{l} w : \mathbf{e} \\ \text{infoOf}(w)(y) \end{array} \right])(x, @_j : \text{animate}(x)),$$

and

$$BUR N \equiv \lambda y. \lambda x. \text{burn}(y, @_i : \left[ \begin{array}{l} w : \mathbf{e} \\ \text{phyObjOf}(w)(y) \end{array} \right])(x, @_j : \text{animate}(x)).$$

Thus, the sentence in (12b) is assigned the following SR.

$$(13) \left[ \begin{array}{l} \text{memorize}(Z_2, @_{15} : \left[ \begin{array}{l} x : \mathbf{e} \\ \text{infoOf}(x)(Z_2) \end{array} \right])(m, @_{16} : \text{animate}(m)) \\ \text{burn}(Z_3, @_{17} : \left[ \begin{array}{l} x : \mathbf{e} \\ \text{phyObjOf}(x)(Z_3) \end{array} \right])(m, @_{18} : \text{animate}(m)) \end{array} \right]$$

where  $Z_2$  abbreviates

$$\pi_1 \pi_1 \left( @_7 : \left[ \begin{array}{l} z : \left[ \begin{array}{l} x : \mathbf{e} \\ @_2^{pr}(x) \end{array} \right] \\ (@_6 : \left[ \begin{array}{l} x : \mathbf{e} \\ @_1^{pr}(x) \end{array} \right] \rightarrow \left[ \begin{array}{l} x : \mathbf{e} \\ @_2^{pr}(x) \end{array} \right] \rightarrow \left[ \begin{array}{l} x : \mathbf{e} \\ @_3^{pr}(x) \end{array} \right] \rightarrow \mathbf{t} \end{array} \right] (b, (@_4 : @_1^{pr}(b)))(m, (@_5 : @_2^{pr}(m)))(z) \end{array} \right),$$

and  $Z_3$  abbreviates

$$\pi_1 \pi_1 \left( @_{14} : \left[ \begin{array}{l} z : \left[ \begin{array}{l} x : \mathbf{e} \\ @_9^{pr}(x) \end{array} \right] \\ (@_{13} : \left[ \begin{array}{l} x : \mathbf{e} \\ @_8^{pr}(x) \end{array} \right] \rightarrow \left[ \begin{array}{l} x : \mathbf{e} \\ @_9^{pr}(x) \end{array} \right] \rightarrow \left[ \begin{array}{l} x : \mathbf{e} \\ @_{10}^{pr}(x) \end{array} \right] \rightarrow \mathbf{t} \end{array} \right] (b, (@_{11} : @_8^{pr}(b)))(m, (@_{12} : @_9^{pr}(m)))(z) \end{array} \right).$$

Let us suppose that we have the following information in the global context  $\mathcal{K}_2$ :

$$\begin{aligned}
\mathcal{K}_2 &\equiv \mathbf{t} : \mathbf{k}, \mathbf{e} : \mathbf{t}, \\
m &: \mathbf{e}, b : \mathbf{e}, i_b : \mathbf{e}, p_b : \mathbf{e}, \\
\text{animate} &: \mathbf{e} \rightarrow \mathbf{t}, \text{book} : \mathbf{e} \rightarrow \mathbf{t}, \\
\text{infoOf} &: \mathbf{e} \rightarrow \mathbf{e} \rightarrow \mathbf{t}, \text{phyObjOf} : \mathbf{e} \rightarrow \mathbf{e} \rightarrow \mathbf{t},
\end{aligned}$$

$$\begin{aligned}
&\mathbf{memorize} : \left[ \begin{array}{l} y : \mathbf{e} \\ w : \mathbf{e} \\ \mathbf{infoOf}(w)(y) \end{array} \right] \rightarrow \left[ \begin{array}{l} x : \mathbf{e} \\ \mathbf{animate}(x) \end{array} \right] \rightarrow \mathbf{t}, \\
&\mathbf{burn} : \left[ \begin{array}{l} y : \mathbf{e} \\ w : \mathbf{e} \\ \mathbf{phyObjOf}(w)(y) \end{array} \right] \rightarrow \left[ \begin{array}{l} x : \mathbf{e} \\ \mathbf{animate}(x) \end{array} \right] \rightarrow \mathbf{t}, \\
&\mathbf{book}_{\mathbf{infoOf}} : (x : \mathbf{e}) \rightarrow (\mathbf{book}(x) \rightarrow \left[ \begin{array}{l} y : \mathbf{e} \\ \mathbf{infoOf}(x)(y) \end{array} \right]), \\
&\mathbf{book}_{\mathbf{phyObjOf}} : (x : \mathbf{e}) \rightarrow (\mathbf{book}(x) \rightarrow \left[ \begin{array}{l} y : \mathbf{e} \\ \mathbf{phyObjOf}(x)(y) \end{array} \right]), \\
&p_1 : \mathbf{animate}(m), \quad p_2 : \mathbf{book}(o), \\
&p_3 : \mathbf{infoOf}(b)(i_b), \quad p_4 : \mathbf{phyObjOf}(b)(p_b).
\end{aligned}$$

Then we can find a proof term for each @-term in SR  $Z_2$  as follows. Here  $@_i \mapsto T$  means that the underspecified term  $@_i$  is replaced with a term  $T$ .

$$@_1 \mapsto \mathbf{book},$$

$$@_2 \mapsto \mathbf{animate},$$

$$@_3 \mapsto \mathbf{infoOf}(b),$$

$$@_4 \mapsto p_2,$$

$$@_5 \mapsto p_1,$$

$$@_6 \mapsto \lambda y. \lambda x. \lambda z. \left[ \begin{array}{l} u : \mathbf{book}(y) \\ \mathbf{book}_{\mathbf{infoOf}}(y)(u) =_{\mathbf{e}} z \end{array} \right],$$

$$\begin{aligned}
& @_7 \mapsto \\
& ((i_b, p_3), (\lambda y. \lambda x. \lambda z. \left[ \begin{array}{l} u : \mathbf{book}(y) \\ \mathbf{book}_{\mathbf{infoOf}}(y)(u) =_{\mathbf{e}} z \end{array} \right] )(b, p_2)(m, p_1)(i_b, p_3)).
\end{aligned}$$

And we can also find a proof term for each @-term in SR  $Z_3$ :

$$@_8 \mapsto \mathbf{book},$$

$$@_9 \mapsto \mathbf{animate},$$

$$@_{10} \mapsto \mathbf{phyObjOf}(b),$$

$$@_{11} \mapsto p_2,$$

$$@_{12} \mapsto p_1,$$

$$@_{13} \mapsto \lambda y. \lambda x. \lambda z. \left[ \begin{array}{l} u : \mathbf{book}(y) \\ \mathbf{book}_{\mathbf{phyObjOf}}(y)(u) =_{\mathbf{e}} z \end{array} \right],$$

$$\begin{aligned}
& @_{14} \mapsto \\
& ((p_b, p_3), (\lambda y. \lambda x. \lambda z. \left[ \begin{array}{l} u : \mathbf{book}(y) \\ \mathbf{book}_{\mathbf{phyObjOf}}(y)(u) =_{\mathbf{e}} z \end{array} \right] )(b, p_2)(m, p_1)(p_b, p_4)).
\end{aligned}$$

The rest of the underspecified terms can also be replaced with specific terms as follows.

$$@_{15} \mapsto p_3,$$

$$@_{16} \mapsto p_1,$$

$$@_{17} \mapsto p_4,$$

$$@_{18} \mapsto p_1.$$

By eliminating each @-term in (13) and reducing  $\beta$ -redexes, we obtain the following as a fully specified semantic representation for the sentence (12b).

$$(14) \quad \left[ \begin{array}{l} \mathbf{memorize}(i_b, p_3)(m, p_1) \\ \mathbf{burn}(p_b, p_4)(m, p_1) \end{array} \right]$$

## 6 Conclusion

In this paper, we proposed an analysis that treats the selectional restrictions of predicates as presuppositions. In addition, using argument operators, we gave a unified analysis of lexical phenomena that are not accounted for by simple analyses of selectional restrictions. Future work includes extending our analysis to phenomena such as metaphors, which Asher [1] opened up a way to analyze in type theoretical settings.

**Acknowledgements.** We thank the two anonymous reviewers for helpful comments and suggestions. We also thank the audience of LENLS13 for their valuable comments and discussions. This work was supported by CREST, Japan Science and Technology Agency.

## References

1. Asher, N.: *Lexical Meaning in Context: A Web of Words*. Cambridge University Press, Cambridge (2011)
2. Asher, N.: Selectional restrictions, types and categories. *J. Appl. Logic* **12**(1), 75–87 (2014)
3. Beaver, D.I.: *Presupposition and Assertion in Dynamic Semantics*. Studies in Logic, Language and Information. CSLI Publications & FoLLI, Stanford (2001)
4. Bekki, D.: Representing anaphora with dependent types. In: Asher, N., Soloviev, S. (eds.) *LACL 2014. LNCS*, vol. 8535, pp. 14–29. Springer, Heidelberg (2014). doi:[10.1007/978-3-662-43742-1\\_2](https://doi.org/10.1007/978-3-662-43742-1_2)
5. Bekki, D., Mineshima, K.: Context-passing and underspecification in dependent type semantics. In: Chatzikyriakidis, S., Luo, Z. (eds.) *Modern Perspectives in Type-Theoretical Semantics. SLP*, vol. 98, pp. 11–41. Springer, Cham (2017). doi:[10.1007/978-3-319-50422-3\\_2](https://doi.org/10.1007/978-3-319-50422-3_2)
6. Luo, Z.: Formal semantics in modern type theories with coercive subtyping. *Linguist. Philos.* **35**(6), 491–513 (2012)
7. Magidor, O.: *Categority Mistakes*. Oxford University Press, Oxford (2013)

8. Martin-Löf, P.: Intuitionistic Type Theory. Bibliopolis, Naples (1984)
9. McCawley, J.D.: Concerning the base component of a transformational grammar. *Found. Lang.* **4**(3), 243–269 (1968)
10. Nunberg, G.: Transfers of meaning. *J. Semant.* **12**(2), 109–132 (1995)
11. Retoré, C.: The montagovian generative lexicon *lambda tyn*: a type theoretical framework for natural language semantics. In: 19th International Conference on Types for Proofs and Programs (TYPES 2013), pp. 202–229 (2014)
12. Steedman, M.: Surface Structure and Interpretation. The MIT Press, Cambridge (1996)

New Frontiers in Artificial Intelligence

JSAI-isAI 2016 Workshops, LENLS, HAT-MASH, AI-Biz,  
JURISIN and SKL, Kanagawa, Japan, November 14-16,  
2016, Revised Selected Papers

Kurahashi, S.; Ohta, Y.; Arai, S.; Satoh, K.; Bekki, D.  
(Eds.)

2017, VIII, 345 p. 96 illus., 55 illus. in color., Softcover  
ISBN: 978-3-319-61571-4