COERCION AS PROOF SEARCH IN DEPENDENT TYPE SEMANTICS

ERIKO KINOSHITA, KOJI MINESHIMA & DAISUKE BEKKI
Ochanomizu University

ABSTRACT

This paper presents an analysis of coercion and related phenomena in the framework of Dependent Type Semantics (DTS). Using underspecified terms in DTS, we present an analysis of selectional restriction as presupposition; we then combine it with a type called transfer frame to provide an analysis of coercion. Our analysis focuses on the fact that coercion is triggered not only by type mismatch between predicates and their arguments, but also by more general inference with contextual information. We show how the analysis can be extended to copredication of logical polysemy and complement coercion. Finally, we will suggest that this analysis can shed light on an aspect of complicity that is invoked in interpreting coercion and other meaning-shifting phenomena.

[1] INTRODUCTION

Recently, various extended type systems have been developed to explain phenomena related to coercion (Pustejovsky 1993; Asher 2011; Luo 2012; Asher & Luo 2012; Cooper 2012; Retoré 2014; Mery & Retoré 2017). These systems are enriched with more fine-grained types than is commonly assumed in formal semantics, which enables us to analyze selectional restrictions of predicates as type matching. According to the widely held view, an operation for dealing with coercion is triggered when a predicate and its argument cause a type mismatch during the course of semantic composition. For example, the predicate escape demands that its subject is an animate entity; thus, the literal reading of (1-a) produces a type mismatch. When it is uttered in a context where there is a man who ate an omelet in a café, however, (1-a) can be understood as meaning (1-b) (Nunberg 1995).

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

The coercion-as-type-mismatch view naturally accounts for this reading.

However, there is a problem with the naive assumption that coercion is only triggered by type mismatch: it can be contextually triggered without any type mismatch, so as to find a more relevant interpretation. For example, (2-a) has

(2) a. The man escaped.
   b. The man who ate an omelet escaped.

The coercion-as-type-mismatch view does not naturally account for this reading.

The transfer frame approach (Kinoshita et al. 2021) can explain this contextually triggered coercion.
the reading in (2-b), even though the literal reading does not produce a type mismatch.

(2)   a. The lion escaped.
b. The actor who plays the lion at *The Lion King* escaped (from the theater).

This suggests that coercion is not only driven by type matching in semantic composition but also by inference with contextual information.

This paper proposes a formal analysis of coercion in the framework of Dependent Type Semantics (DTS) (Bekki 2014; Bekki & Mineshima 2017), a framework of proof-theoretic semantics that combines dependent types with the mechanism of underspecification for handling anaphora and presupposition. In this framework, selectional restrictions are analyzed as presuppositions, and the satisfaction of presuppositions is calculated at the stage of type checking that involves proof search. Using this framework, we analyze coercion at the stage of type checking rather than at the stage of semantic composition. We argue that this analysis captures the inferential aspect of coercion as shown by phenomena such as (2). We also show that it can be extended to related phenomena, including copredication and complement coercion.

The structure of the paper is as follows. Section [2] gives an overview of the basic framework of DTS, focusing on how semantic underspecification works to represent and compute anaphoric elements. Section [3] provides an analysis of selectional restrictions and shows how the presuppositional behavior of selectional restrictions is derived in DTS. With this background, we present an analysis of coercion triggered by contextual inference in Section [4] and then extend it to an analyses of copredication [5.1] and complement coercion [5.2]. Finally, in Section [6] we will suggest that our analysis accounts for an aspect of interpretation called complicity involved in coercion.

[2] DEPENDENT TYPE SEMANTICS

[2.1] Dependent types

Dependent types have been used to analyze various aspects of natural languages (Ranta 1994; Cooper 2012; Luo 2012). Compared with simple types, dependent types can express types depending on terms. For instance, man\(x\) is a type depending on a term \(x\). We say man\(x\) has type type, \(x\) is a term of type entity, and man is a predicate of type entity → type. Under the so-called Curry-Howard correspondence, a type can be identified with a proposition; thus, the type man\(x\) is regarded as the proposition that \(x\) is a man. A term having such a type is called a proof term. For instance, \(p : \text{man}(x)\) expresses that a proof term \(p\) has the type man\(x\), that is, \(p\) is a proof for the proposition that \(x\) is a man.
Two kinds of dependent types play an important role: (i) II-type (dependent function type), written \((x : A) \to B\), is a generalized form of a function type \(A \to B\); a term of type \((x : A) \to B\) is a function \(f\) that takes a term \(a\) of type \(A\) and returns a term \(f(a)\) of type \(B[a/x]\). (ii) \(\Sigma\)-type (dependent product type), written \((x : A) \times B\) or \([x : A][x : B]\), 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 \(B[t/x]\). The projection functions \(\pi_1\) and \(\pi_2\) are defined in such a way that \(\pi_1(t, u) = t\) and \(\pi_2(t, u) = u\). When the variable \(x\) does not occur free in \(B\), \((x : A) \to B\) and \((x : A) \times B\) can be reduced to \(A \to B\) and \(A \times B\), respectively. Given the identification of types with propositions, \(A \to B\) and \(A \times B\) correspond to implication and conjunction, respectively. See Martin-Löf (1984) and Ranta (1994) for more details on dependent types.

In contrast to model-theoretic semantics, the semantics of natural language based on dependent types can be called proof-theoretic semantics, where the meaning of a sentence is regarded as a proof-condition, a condition specified by inference rules, rather than as a truth-condition. By taking proof-conditions as a central aspect of meaning, semantic theories based on dependent types are particularly suitable for capturing inferential aspects of interpretations.

Figure 1 shows formation rules \((\Pi F, \Sigma F)\), introduction rules \((\Pi I, \Sigma I)\) and elimination rules \((\Pi E, \Sigma E)\) for II-type and \(\Sigma\)-type we use in this paper.

It is important to see that II-type and \(\Sigma\)-type correspond to universal and existential quantification, respectively. For example, the sentences in (3-a) can be analyzed as having a semantic representation (SR) in (3-b).

(3)  a. Every man entered.

[1] For conjunction \(A \times B\), we also use a two-dimensional notation \(A \atop B\) for readability. For nested \(\Sigma\)-types, we abbreviate \(x : A \atop y : B\atop C\) as \(x : A \atop y : B\atop C\).
b. \((u: [x: \text{entity} \newline \text{man}(x)]) \rightarrow \text{enter}(\pi_1(u))\)

The term \(u\) has a \(\Sigma\)-type: it is a pair of a term (let it be \(x\)) having type \(\text{entity}\) and a proof term having type \(\text{man}(x)\) that depends on \(x\). The \(\Pi\)-type universally quantifies over such pairs of terms. The term \(\pi_1(u)\) in the argument position of \(\text{enter}\) picks up the entity that is the first element of \(u\), namely, the entity \(x\). Thus, the entire SR in (3-b) says that for every entity \(x\) such that \(x\) is a man, \(x\) entered. This is equivalent to the standard truth-condition for (3-a).

[2.2] Underspecification

DTS (Bekki 2014; Bekki & Mineshima 2017) differs from the previous studies such as Ranta (1994) and Luo (2012) in that it has underspecified terms @ to represent context-dependent elements.\(^2\) For example, the sentence in (4-a) is given the SR in (4-b) that contains an underspecified term @.

(4) a. He whistled.
   b. \(\text{whistle}(\pi_1(@: [x: \text{entity} \newline \text{man}(x)]))\)

The @-term plays the role of a hole to be filled in at a later stage of interpretation called type checking. The type \(A\) in @ : \(A\), called type annotation, specifies the type of the underspecified term @. In this case, the term @ has the \(\Sigma\)-type standing for the proposition that there is an entity \(x\) such that \(x\) is a man. Since a term having the \(\Sigma\)-type is a pair of objects, the first element is picked up by the projection function \(\pi_1\) and the predicate \(\text{whistle}\) applies to it.

Such an underspecified SR is compositionally derived from a parse tree for a sentence.\(^3\) Then the well-formedness of an SR is proved via type-checking. This launches a derivation of \(A : \text{type}\) for the underspecified SR \(A\), given a background context called a global context. If \(A\) contains an underspecified term @, this process of type-checking invokes a process of proof search, i.e., a process of constructing a proof term for @. This is the process of filling a hole indicated by @ with a suitable term having the type annotated to @.

In the case of (4-b), the type checking to ensure that this SR is well-formed, i.e., it has type type, runs as in Figure 2. Here we use the @-rule shown in Figure 1. This

---

\(^{[2]}\) Another difference between DTS and previous proposals is that in DTS, a common noun like man is analyzed as a predicate rather than as a type. Thus, (3-a) is analyzed as (3-b), not as \((x: \text{man}) \rightarrow \text{enter}(x)\). See Bekki & Mineshima (2017) and Chatzikyriakidis & Luo (2017b) for a detailed comparison of two approaches.

\(^{[3]}\) See Bekki (2014); Bekki & Mineshima (2017) for an approach to compositionality in DTS based on Combinatory Categorial Grammar (CCG).
The judgement $A \text{ true}$ means that there exists a proof term having the type $A$. Thus, this triggers a process of proof search to construct a term having the annotated type. Starting from the root of the derivation tree in Figure 2, one needs to fill the gap in $\Lambda$ by finding a term having the annotated $\Sigma$-type. This means that one has to construct a pair $(a, t)$ such that $a$ is an entity and $t$ is a proof of the proposition that $a$ is a man. Suppose that we can construct such a term $(a, t)$ from the global context where $a : \text{entity}$ and $t : \text{man}(a)$ hold. By replacing @ in (4-b) with the constructed term $(a, t)$, we obtain $\text{whistle}(\pi_1(a, t))$, which computes to $\text{whistle}(a)$. This yields a fully specified SR for (4-a) in this context.

Figure 3 shows the two-stage process of deriving a well-formed SR. Note that semantic composition at the first stage is deterministic, while proof search at the second stage can be non-deterministic if there are more than one way to resolve an underspecified term. See Bekki & Mineshima (2017) for more discussions on anaphora, including E-type anaphora, donkey anaphora and bridging inferences.

---

[4] If the annotated type contains another @-term, a separate process of proof search is launched by the sub-derivation starting from $A : \text{type}$ in the left premise of the @-rule.

[5] Note that the other two branches are closed using judgements such as $\text{whistle} : \text{entity} \rightarrow \text{type}$ and $\text{man} : \text{entity} \rightarrow \text{type}$, which we assume are contained in the global context.
[3] **SELECTIONAL RESTRICTIONS**

[3.1] **Types versus predicates**

There are at least two approaches to representing selectional restrictions in extended type-theoretic frameworks, i.e., those frameworks that go beyond a simply-typed setting with entities and truth-values as base types. The first one is to represent selectional restrictions as types (Asher 2011; Luo 2012; Retoré 2014); for instance, taking animate as a base type, one can assign a type animate → prop to the predicate escape. According to this approach, the violation of selectional restrictions leads to type mismatch. To avoid type mismatch during the course of semantic composition, then, one needs to enrich the system with subtyping. For instance, to combine the predicate escape of type animate → prop with the term john of type human, one needs a subtyping relation human < animate and extra subtyping rules. A disadvantage is that these additional subtyping rules may make the resulting compositional semantics more complicated.

The other approach is to represent selectional restrictions as predicates; for instance, if one uses the base type for entities, the selectional restriction of the verb escape can be represented as animate(x), i.e., as a predicate over entities. Although it is underdeveloped, this second approach has an advantage in that it dispenses with subtyping and thus can preserve the well-understood mechanism of the syntax-semantics interface. We are going to build on this second approach and integrate it with the two-stage architecture of interpretation in DTS.

[3.2] **Selectional restrictions as presupposition**

Selectional restrictions show the behavior of presuppositions. One piece of evidence comes from presupposition projection. For instance, one can infer that John is animate not only from (5-a) but also from (5-b).

\[
(5) \quad \text{a. John escaped. } [\Rightarrow \text{John is animate}] \\
\text{b. John didn’t escape. } [\Rightarrow \text{John is animate}]
\]

If selectional restrictions of predicates were part of entailment, the verb escape would be assigned the meaning in (6-a), which would predict that (5-b) has the semantic representation in (6-b).

\[
(6) \quad \text{a. } \lambda x. \text{escape}(x) \land \text{animate}(x) \\
\text{b. } \neg(\text{escape}(\text{john}) \land \text{animate}(\text{john}))
\]

Clearly, this does not account for the inference in (5-b). More generally, selectional restrictions project out of the scope of entailment-canceling operators such as negation, modals, and conditionals (Magidor 2013; Asher 2014).

---

[6] The choice between predicates and types is related to the analysis of common nouns in type theory; see footnote 2 and references cited there.
Further evidence comes from the fact that selectional restrictions show the behavior of presupposition filtering as exemplified by (7):

(7) If numbers are coloured, then the number two is green. (Magidor 2013, p.135)

The violation of a selectional restriction observed in the number two is green is absent in (7) due to presupposition filtering, for which the naive type-mismatch analysis would yield a wrong prediction.

[3.3] Transfer Frame

In simple type theory, the predicate escape denotes a function from entities to truth-values, i.e., a function of the type e → t. In dependent type theory, we can put an additional condition for selectional restriction on arguments; thus we assign the following semantic type to the predicate escape.

(8) escape : [x : entity animate(x)] → type.

The predicate escape is a function that takes a pair of an entity \(x\) and a proof of \(x\)’s being animate and returns a type (i.e., a proposition). In DTS, we can leave underspecified the second argument, i.e., a proof term for selectional restriction; that is, by filling the second argument with an underspecified term @, we do not have to find a specific proof term for selectional restriction at the stage of semantic composition.

Now we can define the lexical entry for an intransitive verb as in (9).

(9) The lexical entry for the verb escape (Version 1)

\[ [[\text{escape}]] = \lambda x.\text{escape}(x, @ : \text{animate}(x)) \]

Here the type of \([[\text{escape}]]\) is entity → type. When the verb is combined with the subject NP, say, John, whose SR is \([[\text{John}]] = j\) of type entity, we can derive the following:

(10) \([[\text{escape}]]([[\text{John}]])) = \text{escape}(j, @ : \text{animate}(j)).

For this SR to be well-formed, one has to find a proof term for the type \(\text{animate}(j)\) annotated to @. This correctly predicts the behavior of selectional restriction for the verb escape.

However, this analysis does not cover the interpretations of coercion including ones driven by contextual inference such as (2). Accordingly, we general-

---

[7] Note that this type is equivalent to \((x : \text{entity}) \rightarrow \text{animate}(x) \rightarrow \text{type}\).

[8] This is the entry proposed in Kinoshita et al. (2016).

[9] See Kinoshita et al. (2016) for more details.
ize the entry in (9) and propose a new lexical entry for predicates, using a type $\mathcal{T F}^x_{F\text{-target}}$, called Transfer Frame, relativized to an entity $x$ and a predicate $F\text{-target}$. In the case of escape, the new lexical entry is given in (11).

(11) The lexical entry for the verb escape (Version 2)

$$[[\text{escape}]] = \lambda x. \text{escape}(\pi_1(\_ : \mathcal{T F}^x_{\text{animate}}))$$

The type $\mathcal{T F}^x_{F\text{-target}}$ is defined as follows.\(^{11}\)

\[\begin{align*}
\mathcal{T F}^x_{F\text{-target}} \triangleq & \quad [t\text{-pair} : [ y : \text{entity} \\
& \quad \quad \quad \quad \quad \text{F-target}(y) ] ] \\
& \quad [F\text{-source} : \text{entity} \rightarrow \text{type}] \\
& \quad [p : F\text{-source}(x)] \\
& \quad [R : [ w : \text{entity} \\
& \quad \quad \quad \quad \quad \text{F-source}(w) ] ] \rightarrow [ y : \text{entity} \\
& \quad \quad \quad \quad \quad \text{F-target}(y) ] ] \rightarrow \text{type} \\
& \quad [R(x, p)t\text{-pair}]
\end{align*}\]

Some remarks are in order. The predicate in $F\text{-target}$ stands for the selectional restriction imposed on the argument of the verb in question. The term $x$, called the source term, is the entity denoted by the NP in the argument position of a given sentence. The role of $\mathcal{T F}^x_{F\text{-target}}$ is to link the source term $x$ to a term $y$ (called the target term) that satisfies the selectional restriction in $F\text{-target}$ by means of a relation $R$ holding between $x$ and $y$.

More specifically, given a source term $x$ and a predicate $F\text{-target}$, the underspecified term $\_ : \mathcal{T F}^x_{F\text{-target}}$ searches a 5-tuple $(t\text{-pair}, F\text{-source}, p, R, q)$ that satisfies the following conditions: (i) $t\text{-pair}$ is a pair of a target term $y$ and a proof term of the proposition that $y$ has the predicate $F\text{-target}$; (ii) $F\text{-source}$ is a predicate of type $\text{entity} \rightarrow \text{type}$; (iii) $p$ is a proof term of the proposition that the source term $x$ has the predicate $F\text{-source}$; (iv) $R$ is a relation between the source term $x$ and the target term $y$; (v) $q$ is a proof term of $R(x, p)t\text{-pair}$, i.e., the proposition that $R$ holds between the source term $x$ and the target term $y$.

If such a 5-tuple is constructed, the main predicate applies to its first projec-

\[^{10}\] The transfer frame generalizes the operator $\text{arg}$ (called argument operator) introduced in Kinoshita et al. (2016). A problem with the argument operator $\text{arg}$ in Kinoshita et al. (2016) is that it does not derive the meaning of coercion when a sentence does not cause type mismatch. So it does not cover the case of coercion driven by contextual inference shown in (2).

\[^{11}\] It should be noted that $\mathcal{T F}$ is defined as a $\Sigma$-type, although the notation used here is similar to record types (Cooper 2012). See also the remark on the notation in footnote 1.

\[^{12}\] Note that the relation $R$ takes as argument two pairs of an entity and a proof term; the proof term ensures that the entity satisfies the selectional restriction of $R$.

\[^{13}\] In the definition of $\mathcal{T F}$, we use a $\Sigma$-type of the form $(X : A \rightarrow \cdots \rightarrow \text{type}) \times B$, which is known to cause Girard’s paradox (Hook & Howe 1986). We can avoid the inconsistency by introducing the hierarchy of universes; but discussing this problem in detail will take us too far afield from the focus of this paper.
Coercion as proof search

\[ \begin{array}{c}
\vdash \text{escape} : \left[ x : \text{entity} \right. \\
\text{animate}(x) \left. \right] \rightarrow \text{type}
\end{array} \]

\[ \pi_1(\ominus : \mathcal{T} \mathcal{F}^j_{\text{animate}}) : \left[ x : \text{entity} \right. \\
\text{animate}(x) \left. \right] \quad (\Pi E) \]

**Figure 4:** The type checking for (13)

tion, namely, \textit{t-pair}, as shown in the lexical entry in (11). Since \textit{t-pair} is a pair of an entity \( y \) (the target term) and a proof term for \( F\text{-target}(y) \), it matches with the type of the predicate \textit{escape} in (8).

**[3.4] Literal interpretation**

To see how the analysis works, consider first the sentence in (5-a) with its literal interpretation. Using the lexical entry of \textit{escape} in (11), one can derive the SR of (5-a) as (13), where \( j \) is the term for \textit{John}.

\[ \text{escape}(\pi_1(\ominus : \mathcal{T} \mathcal{F}^j_{\text{animate}})) \]

The type checking of this underspecified SR goes as shown in Figure 4, in a similar way to the one in Figure 2. In this derivation, \( \mathcal{T} \mathcal{F}^j_{\text{animate}} \text{true} \) triggers in \( \Lambda \) a process of proof search to construct a 5-tuple satisfying the transfer frame. In the literal reading of (5-a), the target term should be identical to the source term \( j \), both being an animate entity; in such a case, we can use the equality relation between animate entities for \( R \). Thus, assuming that it is established in the global context that \textit{John} is an animate entity, we can construct a 5-tuple as shown in Figure 5.\(^{14}\) By substituting \( \ominus \) in (13) with this 5-tuple, we obtain an SR \( \text{escape}(\pi_1((j, t), \text{anim}, t, =, \text{refl}(j, t))) \), which computes to \( \text{escape}(j, t) \), where \( t \) is a proof term for \textit{animate}(\textit{j}). In this way, the literal interpretation of (5-a) is correctly derived.

**[3.5] Presupposition projection**

Our analysis can account for the presuppositional inferences triggered by selectional restrictions. For presupposition projection, let us consider the case of negation. The SR of the negative sentence in (5-b) is given as (14).

\[ \text{escape}(\pi_1(\ominus : \mathcal{T} \mathcal{F}^j_{\text{animate}})) \]

\[ \text{escape}(\pi_1((j, t), \text{anim}, t, =, \text{refl}(j, t))) \]

\[ \text{escape}(j, t) \]

\[ \text{animate}(j) \]

\[ \text{OSLa volume 10(2), 2018} \]

\[ \text{Here we write entity as e, type as t, animate as anim, and a tuple (a, (b, c)) as (a, b, c). The equality in dependent type theory is relativized to a type, but as a notational convenience, we make this type implicit: the equality = here should be understood as = \left( x : \text{entity} \right) \times \text{anim}(x). The constructor refl for equality is also implicitly relativized to this type; see Nordström et al. (1990) for the treatment of equality in dependent type theory.} \]
(13) \[ \neg \text{escape}(\Theta : \mathcal{T} \mathcal{F}^j_{\text{animate}}) \]

The negation has the following formation rule.\(^{15}\)

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

This rule says \(\neg A\) is a type if \(A\) is a type, which means that both \(A\) and \(\neg A\) have the same well-formedness condition. Hence, it is correctly predicted that (14) has the same presupposition as the positive counterpart (13); namely, in both cases, it is required to find a proof term for the proposition \(\text{animate}(j)\). This means that John is animate projects out of the scope of negation.

Our analysis can also deal with the case of presupposition filtering in (7). To simplify the matter, consider the sentence (15-a), whose SR is given in (15-b).

(15)  
\begin{enumerate}
\item If the number two is colored, the number two is green.
\item \((u : \text{colored}(\text{two})) \rightarrow \text{green}(\pi_1(\Theta : \mathcal{T} \mathcal{F}^{\text{two}}_{\text{colored}}))\)
\end{enumerate}

Here we assume that the predicate \text{green} imposes the property of being colored on its argument. The sentence (15-a) as a whole can be used without violating this selectional restriction of the predicate \text{green}.

This filtering behavior is correctly predicated as shown in the type checking in Figure 6. Here, we assume that the relation \(R\) is set to the equality relation; then, the essential part of the derivation is to find a proof term for \(\text{colored}(\text{two})\),

---

\(^{15}\) Since \(\neg A\) is defined as \(A \rightarrow \bot\), this rule can be seen as a special case of \(\Pi F\) in Figure 1.
\[
\text{(15-b)} \quad \text{coercion as proof search} \\
\text{colored (two)} : \text{type} \quad \text{green (\pi_1 (@ : T \mathcal{F}_{\text{colored}}^{\text{two}})) : \text{type} (\Pi F, 1)
\]

FIGURE 6: The type checking for the SR (15-b)

namely, the proposition that the term two satisfies the selectional restriction of the predicate green. In this case, the information in the antecedent, colored\text{(two)}, is passed to the consequent using the formation rule (\Pi F) in Figure 1; according to this rule, \((x : A) \rightarrow B\) is a type if (i) \(A\) is a type and (ii) \(B\) is a type under the assumption that there is some proof term \(x\) for \(A\). Thus, the SR as a whole can be proved to be well-formed without using any information from the global context.

It has been observed that a negative sentence like (16) has a reading where the apparent violation of selectional restrictions does not lead to infelicity.\(^{16}\)

(16) Rocks can’t have diabetes. (McCawley 1968, p.62)

This phenomena can be naturally accounted for by the mechanism of presupposition accommodation. Let us assume that (16) has the SR in (17), where \(F_{\text{rock}}\) is a predicate such that it is meaningful to ask if rocks have that property.

\[
\neg \text{have (rock, @ : T } \mathcal{F}_{\text{diabetes}}^{\text{rock}}) 
\]

Again, we can take \(R\) to be the equality relation. Then, in the same way as the example in (14), for this SR to be well-formed, it is required to find a proof term for the proposition \(F_{\text{rock}}(\text{diabetes})\). In this case, however, rocks are not the kind of things that can have diabetes, that is, we fail to find a suitable proof of \(F_{\text{rock}}(\text{diabetes})\) in the global context. In that case, the process of local accommodation is available, by means of which the presupposed content does not project and hence is interpreted inside the scope of negation, resulting in the SR \(\neg (F_{\text{rock}}(\text{diabetes}) \land \text{have (rock, diabetes)})\). Although presenting a comprehensive account of presupposition accommodation within the framework of DTS is beyond the scope of this paper\(^{17}\), the view to take the selectional restriction as presupposition from a proof-theoretic perspective has appealing features in accounting for a variety of presuppositional inference patterns arising from the constructions we saw in this section.

\[^{16}\] We thank Robin Cooper for pointing out this issue at the CoPo2017 workshop.

\[^{17}\] But see Mineshima (2008) for an account of presupposition accommodation in a proof-theoretic setting.
Coercion by type mismatch

Let us move on to the case of coercion triggered by type mismatch. Consider the sentence (1-a). Using the lexical entry in (11), the SR of (1-a) is derived as escape(π₁(@ : TFFEanimate)), where o is the term for the omelet. In this case, while the predicate escape requires its argument to be animate, the source term the omelet is not an animate entity. Accordingly, unlike the case of literal interpretation, taking R to be the equality relation leads to the violation of selectional restriction. We need to find a suitable relation R in TFFE in the context.

Suppose that there is a salient entity john who eats the omelet in the context; thus, the global context has the following information.

We can use eat instead of the equality relation for R and construct a 5-tuple ((j, p₁), edible, p₂, eat, p₃) as a term satisfying TFFEanimate. By replacing @ with this 5-tuple and by computation, we can get the SR escape(j, p₁), which captures the intended reading.

Coercion by contextual inference

Next, let us look at an example of coercion by contextual inference. Consider the example in (2-a). The SR of (2-a) is given as escape(π₁(@ : TFFEanimate)), where l is the term for the lion. In this case, the predicate escape takes an animate entity as its argument. Since the source term the lion is an animate entity, the relation R can be set to the identity relation, ending up with the literal interpretation. Alternatively, R can be set to the relation x plays y at The Lion King, which leads to the coercion interpretation. Thus we can derive the coercion interpretation even when type mismatch does not occur.

The literal interpretation conflicts with the common background knowledge that there is no real lion in the theater. So the coercion interpretation is usually preferred. Then, suppose that we have the following information in the global context.

We can use eat instead of the equality relation for R and construct a 5-tuple ((j, p₁), edible, p₂, eat, p₃) as a term satisfying TFFEanimate. By replacing @ with this 5-tuple and by computation, we can get the SR escape(j, p₁), which captures the intended reading.
Given this global context, we need to find a 5-tuple satisfying $T F^l_{\text{animate}}$. Using $\text{play}$ instead of the equality relation for $R$, we can construct a 5-tuple $((\text{actor}, p_1), \text{object}, p_2, \text{play}, p_3)$ as a term satisfying $T F^l_{\text{animate}}$. In a similar way to the case of coercion triggered by type mismatch, we can obtain the SR $\text{escape}(\text{actor}, p_1)$ for the sentence (20-a). The use of transfer frame enables us to handle coercion triggered by type mismatch and coercion triggered by contextual inference in a uniform way.

[5] COPREDICATION AND COMPLEMENT COERCION

In this section, we show how the current proposal accounts for copredication and complement coercion, two phenomena that have been widely discussed in the literature on coercion and selectional restrictions.

[5.1] Copredication of logical polysemy

Nouns having multiple meanings can be classified into accidental and logical polysemy (Asher 2011). For example, the noun bank in (20-a) is accidentally polysemous, while the noun book in (20-b) is logically polysemous.

(20) a. # The bank is closed and muddy.
    b. The book is interesting and heavy.

The structure where two or more equally connected predicates have different selectional restrictions for the same argument is called copredication. The sentence (20-b) shows that the logically polysemous noun book allows copredication, despite the fact that interesting and heavy require different objects (i.e., an informational object and a physical object) as their object argument.

Let us examine sentence (20-b). Like the lexical entry of escape in (11), we define the lexical entries of interesting and heavy as follows.

(21) a. $[[\text{interesting}]] = \lambda x.\text{interesting}(\pi_1(\@ : T F^{x \to \text{Info}}_{\text{Info}}))$
    b. $[[\text{heavy}]] = \lambda x.\text{heavy}(\pi_1(\@ : T F^{x \to \text{PhyObj}}_{\text{PhyObj}}))$

Here, Info and PhyObj are predicates of type entity $\to$ type that represent being an informational object and being a physical object, respectively. The SR of (20-b) is derived as (22), where $b$ is the term for the book and $\times$ stands for conjunction (i.e., product type).

(22) $\text{interesting}(\pi_1(\@_1 : T F^b_{\text{Info}})) \times \text{heavy}(\pi_1(\@_2 : T F^b_{\text{PhyObj}}))$
For the logical polysemous noun book, we assume that the global context has (23-a) and (23-b), which are read as Books have informational aspects and Books have physical object aspects, respectively. Thus, \( i_b \) and \( p_b \) are functions that map a book to its informational aspect and its physical aspect, respectively.

(23)  
\[
\begin{align*}
\text{a. } i_b &: (x : \text{entity}) \to \text{book}(x) \to [y : \text{entity}] \\
&\to \text{Info}(y) \\
\text{b. } p_b &: (x : \text{entity}) \to \text{book}(x) \to [y : \text{entity}] \\
&\to \text{PhyObj}(y)
\end{align*}
\]

Furthermore, the following predicates, \( x \) is an informational aspect of \( y \) (IA) and \( x \) is a physical object aspect of \( y \) (PA) can be defined as follows:

(24)  
\[
\begin{align*}
\text{a. } \text{IA} &: [x : \text{entity}] \to [y : \text{entity}] \to \text{type} \\
&\quad \text{where } \text{IA} \equiv \lambda x.\lambda y. i_b(\pi_1 x)(\pi_2 x) \equiv (z : \text{entity}) \times \text{Info}(z) \ y \\
\text{b. } \text{PA} &: [x : \text{entity}] \to [y : \text{entity}] \to \text{type} \\
&\quad \text{where } \text{PA} \equiv \lambda x.\lambda y. p_b(\pi_1 x)(\pi_2 x) \equiv (z : \text{entity}) \times \text{PhyObj}(z) \ y
\end{align*}
\]

In addition to these, let us suppose that the global context has the following information:

(25)  
\[
b : \text{entity}, \ p_1 : \text{book}(b)
\]

Applying the functions \( i_b \) and \( p_b \) to \( b \) and \( p_1 \) yields the informational aspect and the physical aspect of the book \( b \), namely, \( i_b(b)(p_1) \) and \( p_b(b)(p_1) \), respectively.

The type check of (22) goes as in the ones given in the previous sections. We use the relation \( \text{IA} \) to specify \( R \) for \( @_1 \) and the relation \( \text{PA} \) for \( @_2 \). Then, we can construct 5-tuples \( (i_b(b)(p_1), \text{book}, p_1, \text{IA}, \text{refl}(i_b(b)(p_1))) \) and \( (p_b(b)(p_1), \text{book}, p_1, \text{PA}, \text{refl}(p_b(b)(p_1))) \) for \( @_1 : \mathcal{T} \mathcal{F}^b_{\text{Info}} \) and \( @_2 : \mathcal{T} \mathcal{F}^b_{\text{PhyObj}} \), respectively. The final SR for (20-b) is:

(26)  
\[
\text{interesting}(i_b(b)(p_1)) \times \text{heavy}(p_b(b)(p_1)).
\]

This asserts that the informational aspect of the book is interesting and its physical aspect is heavy, which captures the intended reading of (20-b).

There has been an ongoing debate on a construction in which a logical polysemous nouns like book appears with a numeral quantifier (Asher 2011; Cooper 2011; Gotham 2017; Chatzikyriakidis & Luo 2015, 2017a). Consider:

(27)  
Three heavy books are interesting.
To analyze this kind of sentences, it is necessary to consider what *three books* means, more specifically, whether it counts informational aspects of the books, physical aspects of the books, or both. This is called the *counting problem*. Gotham (2017) argues that a sentence like (27) is true only if there are three books that are individuated both physically and informationally; thus, (27) would be false (i) when there are three copies of the same informational book that are both heavy and interesting or (ii) when there is one physical volume that binds together three informational books that are both heavy and interesting.

Liebesman & Magidor (2017) doubt that (27) has these truth-conditions, arguing that (27) can be true in a situation like (i) or (ii); furthermore, they argue that it is not easy to test the existence of the reading that requires three books individuated both physically and informationally. We agree with Liebesman & Magidor (2017) that the linguistic intuition about examples like (27) is not so clear and more discussion would be needed about the data. So we will leave as future work solving the counting problem in our framework.

[5.2] **Complement coercion**

A verb like *enjoy*, *begin* and *finish* takes an NP-complement denoting an entity, although it require its argument to denote an event. For example, (28) can be interpreted as “Bob is enjoying smoking the cigarette”, though the NP-complement *the cigarette* literally denotes an entity.

(28) Bob is enjoying the cigarette.

The analysis we have developed so far can be naturally extended to handle this type of examples: we define a transfer frame \( \mathcal{T}_{\text{Fev}} \) for events as in (29):

\[
\mathcal{T}_{\text{Fev}} \equiv \begin{bmatrix}
\text{t-pair} : [ e : \text{event} \\
F\text{-target}(e) ] \\
F\text{-source} : \text{entity} \to \text{type} \\
p : F\text{-source}(x) \\
R : [ w : \text{entity} \\
F\text{-source}(w) ] \to [ e : \text{event} \\
F\text{-target}(e) ] \to \text{type} \\
R(x, p)\text{t-pair}
\end{bmatrix}
\]

The difference from the transfer frame \( \mathcal{T}_F \) for entities is that the first element of \( t\text{-pair} \) is an event, rather than an entity.

We assume that the predicate *enjoy* requires an event as its object argument, thus having the following type.

(30) \( \text{enjoy} : [ e : \text{event} ] \to [ x : \text{entity} ] \to \text{type} \)
Here, $\varphi_{enjoy}$ is a function from events to types; $\varphi_{enjoy}(e)$ means the event $e$ satisfies the selectional restriction associated with the verb *enjoy*. Then, the lexical entry of the transitive verb *enjoy* is given as follows.

$$enjoy : \lambda y \lambda x.\varphi_{enjoy}(\pi_1(@_1: T \mathcal{F} ev^{\varphi_{enjoy}}))(\pi_1(@_2: T \mathcal{F}^x_{animate}))$$

For (28) we can derive the SR

$$enjoy(\pi_1(@_1: T \mathcal{F} ev^{cig}_{\varphi_{enjoy}}))(\pi_1(@_2: T \mathcal{F}^{bob}_{animate}))$$

where *bob* is the term for Bob and *cig* is the term for the cigarette.

Now suppose that we have the following information in the global context.

$$bob : entity, cig : entity, smk : event, smoking : event \rightarrow \left[ \begin{array}{c} x : entity \\ animate(x) \end{array} \right] \rightarrow \left[ \begin{array}{c} x : entity \\ cigarette(x) \end{array} \right] \rightarrow \text{type},$$

$$p_1 : \text{animate}(bob), p_2 : \text{cigarette}(cig),$$

$$p_3 : \text{smoking}(smk)(bob, p_1)(cig, p_2), p_4 : \varphi_{enjoy}(smk).$$

Here, $\text{smoking}(smk)(bob, p_1)(cig, p_2)$ means that *smk* is an event of smoking whose subject is *bob* being animate and whose object is *cig* being a cigarette.

The underspecified terms $@_1$ having type $T \mathcal{F} ev^{cig}_{\varphi_{enjoy}}$ and $@_2$ having type $T \mathcal{F}^{bob}_{animate}$ require constructing 5-tuples from the global context. Since *bob* has the property *animate*, $@_2$ finds a 5-tuple $((bob, p_1), \text{animate}, p_1, =, \text{refl}(bob))$ using the equality relation, in the same way as the type checking shown in Section [3.4]. On the other hand, $@_1$ requires constructing a term having an event type that is related to the cigarette in the context. We can construct a 5-tuple shown in (34) for $@_1$.

$$((smk, p_4), \text{cigarette}, p_2, \lambda u \lambda v.\text{smoking}(\pi_1 v)(bob, p_1)u, p_3)$$

Replacing $@_1$ and $@_2$ in (32) with these 5-tuples, we can finally get the SR:

$$enjoy(bob, p_1)(smk, p_4).$$

This means that Bob enjoy the event of smoking the cigarette, which gives the desired interpretation of (28).

---

[18] Here we use the notation $\varphi_F$ to represent the selectional restriction associated with the predicate $F$. Generally speaking, for an object $x$ to satisfy the condition $\varphi_F$ means that it is meaningful to ask whether $x$ is an $F$ or not. To give a characterization of the condition $\varphi$ for each predicate is a long-standing issue. See Magidor (2013, pp.146–148) for some discussion.

[19] For simplicity we treat a definite description as a constant, rather than as an underspecified term like one for a pronoun in (4).
[6] Concluding Remark: Complicity

We have proposed the architecture of interpretation based on proof search for the analysis of coercion and related phenomena. According to the proposed view, the overall process of interpretation consists of two phases: (i) semantic composition, a compositional derivation of an underspecified SR $\varphi$ from a given sentence $S$, and (ii) type checking with proof search, a process of checking the well-formedness of $\varphi$ that may involve searching a suitable term for an underspecified term $@$ in $\varphi$. From the viewpoint of utterance interpretation, the hearer has to do inference (proof search) to understand an utterance, not just to verify the truth of the content of the utterance. On this account, the process of coercion and other meaning shift is not necessarily triggered by type mismatch during semantic composition but also by contextual inference to search for a relevant interpretation of the utterance.

One notable feature of this architecture of interpretation is that a type checking inference is needed even if hearers do not accept the speaker’s utterance, that is, even if they do not think it is true. We suggest that the process of invoking an inference on the side of the hearers to understand an utterance captures an aspect of utterance interpretation called complicity (Camp 2017). A most typical instance of the effect of complicity is found in a metaphorical interpretation. For instance, consider the sentence in (36).

(36) Chairman, you are a Bolshevik! (Cohen 1978, p.8)

If the hearer feels that (36) is in a bad mood, then he is complicit in interpreting bad words even if he does not feel bad about the chairman in question.

The effect of complicity also arises in other phenomena in which a meaning shift occurs. Consider the case of coercion:

(37) a. The {pig/omelet} asked for another helping of food.
    b. {The man who looks like a pig/The man who ate the omelet} asked for another helping of food.

Even in the absence of bad words, the effect of complicity is produced; when interpreting (37-a) as (37-b), the hearer is forced to accommodate herself to the speaker’s way of thinking to grasp what is said by the utterance. The strategy to invoke complicity in conversation is important in accounting for why the speaker chooses to use a construction like (37-a), rather than (37-b).

In this type of coercion, one has to find a suitable relation $R$ that links a source term to a target term on the fly, in order to find a relevant interpretation of the utterance given a background context. That is, in our implementation, one has to construct a relation $R$ in a 5-tuple satisfying the transfer frame from a given global context. This may be called a creative meaning shift. The speaker’s strategy
to invoke complicity, namely, the strategy to purposely make the hearer compute
the meaning shift can be seen in the case of such creative meaning shift.

By contrast, in the case of the literal interpretation that we have seen in Sec-
section [3.4], the option to take $R$ as the equality relation is always available. Simil-
arily, in the case of examples of copredication like (20-b), the relation $R$ has not to
be constructed on the fly but can be already defined in the global context. These
may be classified as conventional meaning shift, in a similar way to dead (or idio-
matic) metaphor in contrast to creative metaphor.

Though a full defense of the present view will depend on how it can analyze
other inferential phenomena such as metaphor, which is left for future work, we
suggest that coercion can best be characterized in inferential terms. The proof-
theoretic conception we developed in this paper has appealing features in ac-
counting for a variety of coercion and other meaning-shifting phenomena in a
formal theory, and it can contribute to a better understanding of aspects of inter-
pretation such as complicity and creative meaning shift.

ACKNOWLEDGMENTS

An earlier version of the paper was presented at the CoPo2017 workshop. We
are grateful to Nicholas Asher, Stergios Chatzikyriakidis, Robin Cooper, Matthew
Gotham and the audience at the CoPo2017 workshop for their useful feedback and
helpful suggestions. Special thanks to Alexandra Spalek and Matthew Gotham for
organizing the workshop.

REFERENCES

Asher, Nicholas. 2011. *Lexical meaning in context: A web of words*. Cambridge Uni-
versity Press.

Asher, Nicholas. 2014. Selectional restrictions, types and categories. *Journal of
Applied Logic* 2(1). 75–87.

Asher, Nicholas & Zhaohui Luo. 2012. Formalisation of coercions in lexical se-
manics. *Sinn und Bedeutung* 17. 63–80.

Bekki, Daisuke. 2014. Representing anaphora with dependent types. In Nich-
olas Asher & Sergei Soloviev (eds.), *Logical Aspects of Computational Linguistics
(LACL2014)*, 14–29. Springer.

Bekki, Daisuke & Koji Mineshima. 2017. Context-passing and underspecification
in dependent type semantics. In Stergios Chatzikyriakidis & Zhaohui Luo (eds.),
*Modern perspectives in type-theoretical semantics*, 11–41. Springer.

Camp, Elisabeth. 2017. Why metaphors make good insults: perspectives, presup-
position, and pragmatics. *Philosophical Studies* 174(1). 47–64.
Chatzikyriakidis, Stergios & Zhaohui Luo. 2015. Individuation criteria, dot-types and copredication: A view from modern type theories. In Proceedings of the 14th international conference on mathematics of language, 39–50.

Chatzikyriakidis, Stergios & Zhaohui Luo. 2017a. Identity criteria of CNs: Quantification and copredication. Talk presented at the CoPo2017 Workshop.

Chatzikyriakidis, Stergios & Zhaohui Luo. 2017b. On the interpretation of common nouns: Types versus predicates. In Stergios Chatzikyriakidis & Zhaohui Luo (eds.), Modern perspectives in type-theoretical semantics, 43–70. Springer.

Cohen, Ted. 1978. Metaphor and the cultivation of intimacy. Critical Inquiry 5(1). 3–12.

Cooper, Robin. 2011. Copredication, quantification and frames. In Logical Aspects of Computational Linguistics (LACL2011), 64–79. Springer.

Cooper, Robin. 2012. Type theory and semantics in flux. Handbook of the Philosophy of Science 14. 271–323.

Gotham, Matthew. 2017. Composing criteria of individuation in copredication. Journal of Semantics 34(2). 333–371.

Hook, James G & Douglas J Howe. 1986. Impredicative strong existential equivalent to Type: Type. Tech. rep. Cornell University.

Kinoshita, Eriko, Koji Mineshima & Bekki Daisuke. 2016. An analysis of selectional restrictions with dependent type semantics. In Proceedings of the 13th international workshop on Logic and Engineering of Natural Language Semantics (LENLS13), 100–113.

Liebesman, David & Ofra Magidor. 2017. Copredication and property inheritance. Philosophical Issues 27(1). 131–166.

Luo, Zhaohui. 2012. Formal semantics in modern type theories with coercive subtyping. Linguistics and Philosophy 35(6). 491–513.

Magidor, Ofra. 2013. Category mistakes. Oxford University Press.

Martin-Löf, Per. 1984. Intuitionistic type theory. Bibliopolis.

McCawley, James D. 1968. The role of semantics in a grammar. In Emmon Bach & R. Harms (eds.), Universals in linguistic theory, 124–169. Holt, Rinehart, and Winston.
Mery, Bruno & Christian Retoré. 2017. Classifiers, sorts, and base types in the Montagovian generative lexicon and related type theoretical frameworks for lexical compositional semantics. In Stergios Chatzikyriakidis & Zhaohui Luo (eds.), Modern perspectives in type-theoretical semantics, 163–188. Springer.

Mineshima, Koji. 2008. A presuppositional analysis of definite descriptions in proof theory. In K. Satoh, A. Inokuchi, K. Nagao & T. Kawamura (eds.), New frontiers in artificial intelligence: JSAI 2007 conference and workshops, revised selected papers, 214–227. Springer-Verlag.

Nordström, Bengt, Kent Petersson & Jan Smith. 1990. Programming in Martin-Löf's type theory. Oxford University Press.

Nunberg, Geoffrey. 1995. Transfers of meaning. Journal of Semantics 12(2). 109–132.

Pustejovsky, James. 1993. Type coercion and lexical selection. In Semantics and the lexicon, 73–94. Springer.

Ranta, Aarne. 1994. Type theoretical grammar. Oxford University Press.

Retoré, Christian. 2014. The Montagovian generative lexicon $\lambda t_y^n$: a type theoretical framework for natural language semantics. 19th International Conference on Types for Proofs and Programs (TYPES2013) 202–229.

AUTHOR CONTACT INFORMATION

Eriko Kinoshita
Ochanomizu University
kinoshita.eriko@is.ocha.ac.jp

Koji Mineshima
Ochanomizu University
mineshima.koji@ocha.ac.jp

Daisuke Bekki
Ochanomizu University
bekki@is.ocha.ac.jp