Paychecks, Presupposition, and Dependent Types

Ribeka Tanaka, Koji Mineshima and Daisuke Bekki

June 1, 2018

Paychecks, Presupposition, and Dependent Types*

Ribeka Tanaka[1], Koji Mineshima[2], and Daisuke Bekki[2]

[1]Kyoto Unviersity
[2]Ochanomizu University
tanaka@nlp.ist.i.kyoto-u.ac.jp, mineshima.koji@ocha.ac.jp, bekki@is.ocha.ac.jp

1 Introduction

The interpretation of a paycheck pronoun poses a problem in that while it is clear that the pronoun is anaphorically related to some preceding expression, there is no antecedent expression that shares the denotation with the pronoun. For instance, in (1), the pronoun *her* does not refer to a particular individual but is interpreted as *her mother*, where *her* is anaphoric on *every girl*.

(1) Every boy^1 loves his$_1$ mother. Every girl^2 hates <u>her</u>.

In this paper, we account for the interpretation of paycheck pronouns by using *dependent function types* (Π-*types*) in dependent type theory (Martin-Löf, 1984). The semantic framework we will adopt in this study is Dependent Type Semantics (DTS; Bekki and Mineshima, 2017). DTS has been developed as an alternative framework to model-theoretic dynamic semantics such as DRT (Kamp and Reyle, 1993) and DPL (Groenendijk and Stokhof, 1991), providing a compositional account of anaphora and presupposition from a proof-theoretic perspective. In particular, DTS provides a unified analysis of various uses of pronouns, including coreference, bound variable anaphora, E-type anaphora, and donkey anaphora (Bekki, 2014; Bekki and Mineshima, 2017). The goal of the paper is to extend this analysis to paycheck sentences such as (1) and to show that it gives an adequate analysis of paycheck pronouns without introducing any special machinery for handling them.

The proposed account differs from the previous approaches in the following respects:

1. The syntax and semantics of paycheck pronouns is the same as that of standard referential pronouns: a pronoun is given a single meaning in the lexicon. Neither complex meaning (cf. Cooper, 1979; Engdahl, 1986) nor type-shifting mechanism (cf. Jacobson, 2000; Charlow, 2017) is necessary for handling paycheck pronouns.

2. Some authors argue that a paycheck pronoun picks up a contextually salient function, i.e., a 'paycheck' function which maps individuals to their paychecks (cf. Cooper, 1979; Engdahl, 1986); we argue that in cases like (1), the 'paycheck' function is derived from the *presupposition* arising from the possessive NP that contains a free variable bound by an outside quantifier; e.g., *his mother* in (1).

In §2, we will briefly introduce the basic analysis of anaphora and presupposition in DTS. In particular, we will see how dependent function types (Π-types) can contribute to the interpretation of pronominal anaphora in quantificational subordination. In §3, we show that this independently motivated account of pronominal anaphora can extend to the case of paycheck anaphora without introducing additional formal mechanisms to the system.[1]

*We thank the anonymous reviewers for their valuable comments and suggestions. This work was partially supported by JST CREST Grant Number JPMJCR1301, Japan.

[1]To make fully explicit how the system works, we will mostly focus on basic examples of paycheck pronouns. Thus, it is not

2 Dependent Type Semantics

2.1 Dependent types

DTS is a proof-theoretic natural language semantics based on dependent type theory (Martin-Löf, 1984). Dependent type theory is a formal system that extends simple type theory with the notion of *types depending on terms*.[2] For example, we say $\mathbf{man}(x)$ is a type depending on a term x. Under the so-called Curry-Howard correspondence (the *propositions-as-types* principle), a type can be regarded as a proposition; the type $\mathbf{man}(x)$ is used to represent the proposition that x is a man. A term inhabiting the type $\mathbf{man}(x)$ is called a *proof term*. For instance, $t : \mathbf{man}(x)$ expresses that a proof term t has the type $\mathbf{man}(x)$, i.e., t is a proof for the proposition that x is a man. Proof terms play a crucial role in representing a *context* for resolving pronominal anaphora.

There are two type constructors in the system, Σ and Π, which play a key role in the analysis developed below. For these types, we use the following notations:

Σ-type (dependent product type)	**Π-type (dependent function type)**
$(x : A) \times B$	$(x : A) \to B$

For readability, a Σ-type $(x : A) \times B$ is also written as $\begin{bmatrix} x : A \\ B \end{bmatrix}$.

Some remarks are in order about what each type means.

1. Σ-type, $(x : A) \times B$, is a generalized form of product type $A \times B$. A term of type $(x : A) \times B$ is a pair (m, n) such that m is of type A and n is of type $B(m)$. The projection functions π_1 and π_2 are defined in such a way that $\pi_1(m, n) = m$ and $\pi_2(m, n) = n$. Under the Curry-Howard correspondence, Σ-type corresponds to *existential* proposition. When the variable x does not occur free in B, $(x : A) \times B$ is reduced to conjunction $A \times B$.

2. Π-type, $(x : A) \to B$, is a generalized form of function type $A \to B$. A term of type $(x : A) \to B$ is a function that takes a term a of type A and returns a term $f(a)$ of type $B(a)$. Π-type corresponds to *universal* proposition. When the variable x does not occur free in B, $(x : A) \to B$ is reduced to implication $A \to B$.

DTS is augmented with underspecified terms, written as @, which are used in the semantic representation (SR) of anaphoric expressions such as pronouns and presupposition triggers. Underspecified terms enable us to obtain the SR of a sentence in a fully compositional way. Below we will explain how one can use dependent types for the SRs of basic sentences (existential and universal sentences) and how the anaphora resolution process works in DTS.

2.2 Σ-type anaphora

As is well known, existential sentences and universal sentences have different anaphoric potentials; any theory of anaphora must capture the difference between two types of sentences. Using dependent types, we classify the class of anaphoric phenomena into two groups: Σ-type anaphora and Π-type anaphora.

An existential quantifier is said to be *externally dynamic* (Groenendijk and Stokhof, 1991) in the sense that the entity introduced by an existential quantifier is accessible to the subsequent sentences. This property can be captured by representing existential quantifiers by means of Σ-types. We call the type of anaphora in which an object (discourse referent) introduced by an existential quantifier is referred to from the subsequent discourse *Σ-type anaphora*.

our purpose here to discuss how the current proposal can extend to related phenomena discussed in the literature, such as Weak Crossover effects, Bach-Peters sentences, and functional questions, among others (cf. Jacobson, 2000). This is left for another occasion.

[2]Dependent type theory has been applied to natural language semantics, in particular, to dynamic semantics and lexical semantics (Sundholm, 1986; Ranta, 1994; Cooper, 2005; Luo, 2012) and to the study of natural language inferences in computational semantics (Chatzikyriakidis and Luo, 2014).

$$\frac{\Gamma \vdash A : \mathbf{type} \quad \Gamma, u : A \vdash B : \mathbf{type}}{\Gamma \vdash (u : A) \to B : \mathbf{type}} \,\Pi F \qquad \frac{\Gamma \vdash A : \mathbf{type} \quad \Gamma, u : A \vdash B : \mathbf{type}}{\Gamma \vdash (u : A) \times B : \mathbf{type}} \,\Sigma F$$

$$\frac{\Gamma \vdash t : (x : A) \to B \quad \Gamma \vdash u : A}{\Gamma \vdash t(u) : B[u/x]} \,\Pi E \qquad \frac{\Gamma \vdash t : (x : A) \times B}{\Gamma \vdash \pi_1(t) : A} \,\Sigma E \qquad \frac{\Gamma \vdash t : (x : A) \times B}{\Gamma \vdash \pi_2(t) : B[\pi_1(t)/x]} \,\Sigma E$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : (x : A) \to B} \,\Pi I \qquad \frac{\Gamma \vdash t : A \quad \Gamma \vdash u : B[t/x]}{\Gamma \vdash \langle t, u \rangle : (x : A) \times B} \,\Sigma I \qquad \frac{\Gamma \vdash A : \mathbf{type} \quad \Gamma \vdash A \; true}{\Gamma \vdash (@ : A) : A} \,@$$

Figure 1: Inference rules: formation rules ($\Pi F, \Sigma F$), elimination rules ($\Pi E, \Sigma E$), introduction rules ($\Pi I, \Sigma I$) and @-rule.

As an illustration, consider the case of singular E-type anaphora, which is a typical case of Σ-type anaphora. In DTS, the sentences (2a) and (3a) are given the SRs (2b) and (3b), respectively.

(2) a. A man entered.

b. $\left[\begin{array}{l} u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right] \\ \mathbf{enter}(\pi_1 u) \end{array}\right]$

(3) a. He whistled.

b. $\mathbf{whistle}(@ : \mathbf{entity})$

The underspecified term $@ : \mathbf{entity}$ in (3b) is introduced by the pronoun *he*.[3] The @-term plays the role of a gap to be filled by the antecedent of the pronoun in question. The form $@ : \Lambda$ is called *type annotation*, where Λ specifies the type of the underspecified term @.

The conjunction of two sentences is represented by a Σ-type; the mini-discourse consisting of (2a) and (3a) is given the SR shown in (4), by conjoining the two SRs (2b) and (3b) in terms of Σ-type.

(4) $\left[\begin{array}{l} v : \left[\begin{array}{l} u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right] \\ \mathbf{enter}(\pi_1 u) \end{array}\right] \\ \mathbf{whistle}(@ : \mathbf{entity}) \end{array}\right]$

We will focus on the reading of the mini-discourse where *he* refers back to *a man*. In this case, the final representation should be as shown in (5).

(5) $\left[\begin{array}{l} v : \left[\begin{array}{l} u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right] \\ \mathbf{enter}(\pi_1 u) \end{array}\right] \\ \mathbf{whistle}(\pi_1 \pi_1 v) \end{array}\right]$

The argument of the predicate **whistle**, i.e., the place of @ in (4), is to be filled by the term $\pi_1\pi_1 v$. Note that the SR (5) is equivalent to $(x : \mathbf{entity}) \times (\mathbf{man}(x) \times \mathbf{enter}(x) \times \mathbf{whistle}(x))$, which says that there is an entity that satisfies the three conditions, $\mathbf{man}(x)$, $\mathbf{enter}(x)$ and $\mathbf{whistle}(x)$.

Now the question is how to derive (5) from (4). In DTS, anaphora resolution is defined as an operation to replace the occurrences of underspecified terms with concrete proof terms. The process consists of *type checking* and *proof search*. Type checking is triggered by the *felicity condition* of a sentence or a discourse, i.e., the requirement that the SR of a sentence or a discourse be a *type* under the given context. Thus, for the mini-discourse consisting of (2a) and (3a) to be felicitous, the following judgment must hold:

[3]For simplicity, we omit the gender information and treat the pronoun *he* as an expression referring to an entity.

(6) $$\mathcal{K} \vdash \begin{bmatrix} v : \begin{bmatrix} u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix} \\ \mathbf{enter}(\pi_1 u) \end{bmatrix} \\ \mathbf{whistle}(@ : \mathbf{entity}) \end{bmatrix} : \mathbf{type}$$

The context $\mathcal{K}$ is called a *global context*, which contains lexical and world knowledge encoded as judgments. Type checking follows inference rules in dependent type theory and a rule for @-operator called @-rule. These rules are shown in Figure 1. Here we write $B[t/x]$ for the substitution of a term t for free occurrences of the variable x in the term B, with possible capture-avoiding renaming of bound variables.

The derivation tree for type checking of (6) can be given as follows.

$$\dfrac{\begin{array}{c}\overline{\qquad}\\ \vdots\\ \mathcal{K} \vdash \begin{bmatrix} u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix} \\ \mathbf{enter}(\pi_1 u) \end{bmatrix} : \mathbf{type}\end{array} \qquad \dfrac{\begin{array}{c}\overline{\qquad}\\ \vdots\\ \mathcal{K}, v : \begin{bmatrix} u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix} \\ \mathbf{enter}(\pi_1 u) \end{bmatrix} \vdash \mathbf{whistle} : \mathbf{entity} \to \mathbf{type}\end{array} \qquad \dfrac{\begin{array}{c}\overline{\qquad}\\ \vdots\\ \mathcal{K}, v : \begin{bmatrix} u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix} \\ \mathbf{enter}(\pi_1 u) \end{bmatrix} \vdash \mathbf{entity} : \mathbf{type}\end{array} \quad \mathcal{K}, v : \begin{bmatrix} u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix} \\ \mathbf{enter}(\pi_1 u) \end{bmatrix} \vdash \mathbf{entity}\ \mathit{true}}{\mathcal{K}, v : \begin{bmatrix} u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix} \\ \mathbf{enter}(\pi_1 u) \end{bmatrix} \vdash (@ : \mathbf{entity}) : \mathbf{entity}}\,{\scriptstyle @}}{\mathcal{K}, v : \begin{bmatrix} u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix} \\ \mathbf{enter}(\pi_1 u) \end{bmatrix} \vdash \mathbf{whistle}(@ : \mathbf{entity}) : \mathbf{type}}\,{\scriptstyle \Pi E}}{\mathcal{K} \vdash \begin{bmatrix} v : \begin{bmatrix} u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix} \\ \mathbf{enter}(\pi_1 u) \end{bmatrix} \\ \mathbf{whistle}(@ : \mathbf{entity}) \end{bmatrix} : \mathbf{type}}\,{\scriptstyle \Sigma F}$$

The crucial step is the one where @-rule is applied. We use a judgment of the form A *true* to mean that the type A is inhabited, that is, there exists a term of the type A. Thus, according to the @-rule, at the step in question we must show that (i) **entity** is a type, which is obviously true, and (ii) there exists a term of **entity** under the given context. The goal to be proved for (ii) is repeated here.

(7) $$\mathcal{K}, v : \begin{bmatrix} u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix} \\ \mathbf{enter}(\pi_1 u) \end{bmatrix} \vdash \mathbf{entity}\ \mathit{true}$$

This judgment launches a process of proof search. In this case, it is shown that a proof term $\pi_1\pi_1 v$ inhabits the type **entity**. Then, by substitute $\pi_1\pi_1 v$ for @ in (4), we can obtain the fully-specified representation in (5), which corresponds to the intended reading in question.[4]

The final representation in (5) shows how Σ-types capture the externally-dynamic aspects of existential sentences. The proof term v introduced by the Σ-type for the sentence (2a) is a structured object (i.e., a tuple). The point is that even if the term x itself is no longer accessible from the argument position of **whistle**, one can pick up that term by applying the sequence of projection functions to v.

Underspecified terms can have more complex types. Consider the case where (3b) is followed by the sentence (8), where the subject NP is a definite description.

(8) The man whistled.

The SR of (8) is given as follows.

(9) $$\mathbf{whistle}\left(\pi_1\left(@ : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix}\right)\right)$$

[4]DTS can account for various factors that select plausible antecedents among entities appearing in the discourse. For instance, the notion of accessibility is obtained for free from the structure of dependent types Σ and Π (cf. Bekki, 2014). Agreement on pronouns, such as gender, person, and number agreement, can be also accounted for by annotating the underspecified term with more fine-grained types, as in the case of presupposition (Tanaka et al., 2017a).

The underspecified term is introduced by the presupposition trigger *the*. The type annotation corresponds to the proposition *there is a man*, which is the existence presupposition associated with the definite description.

To resolve this presupposition, one needs to find a proof term of this proposition at the stage of proof search. This corresponds to the so-called *presupposition binding*. Another option, *presupposition accommodation*, is also available, which is just to assume the existence of a proof term and add one into the current context. In this way, DTS gives a uniform account of anaphora resolution and presupposition resolution using a proof-theoretic setting, along similar lines to the "presupposition as anaphora" paradigm in DRT (van der Sandt, 1992). One crucial difference from DRT is that in DTS, the antecedents of pronominal anaphora and presupposition are constructed by means of proof search; an empirical difference will become clear in the case of Π-type anaphora we will see below.[5] For more details on presupposition projection in DTS, including the treatment of filtering and bridging inferences, see Bekki and Mineshima (2017) and Tanaka et al. (2017a).[6]

2.3 Π-type anaphora

The *externally static* property of universal quantifiers can be captured by Π-types: a term of Π-type is not a pair of objects but a function, so there is no way to directly access to its part. One difference from the standard treatment in dynamic semantics is that a universal quantifier also introduces an object; a Π-type introduces a function as a discourse referent (Ranta, 1994). By Π-*type anaphora* we mean the type of anaphora where a functional discourse referent is used in the subsequent discourse and contributes to resolving anaphoric expressions.

The so-called *quantificational subordination* is a typical instance of Π-type anaphora, which is exhibited by (10).[7]

(10) If every boy receives a present, some boy will open it.

Here, in its most natural reading, the pronoun *it* in the consequent clause refers to the present that the boy in question received. In other words, the pronoun receives the interpretation which depends on *some boy*. The SR of (10) is given in (11).

$$(11)\quad \left(v : \left(\left(u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{boy}(x) \end{bmatrix}\right) \to \begin{bmatrix} p : \begin{bmatrix} y : \mathbf{entity} \\ \mathbf{present}(y) \end{bmatrix} \\ \mathbf{receive}(\pi_1 u, \pi_1 p) \end{bmatrix}\right)\right) \to \begin{bmatrix} w : \begin{bmatrix} z : \mathbf{entity} \\ \mathbf{boy}(z) \end{bmatrix} \\ \mathbf{open}(\pi_1 z, @ \text{: } \mathbf{entity}) \end{bmatrix}$$

According to type checking, the following judgment must hold.

$$(12)\quad \mathcal{K},\, v : \left(u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{boy}(x) \end{bmatrix}\right) \to \begin{bmatrix} p : \begin{bmatrix} y : \mathbf{entity} \\ \mathbf{present}(y) \end{bmatrix} \\ \mathbf{receive}(\pi_1 u, \pi_1 p) \end{bmatrix},\, w : \begin{bmatrix} z : \mathbf{entity} \\ \mathbf{boy}(z) \end{bmatrix} \vdash (@ : \mathbf{entity}) : \mathbf{entity}$$

Using the @-rule, then, the question becomes whether the type **entity** is inhabited under this context. In this case, by applying the function v to the term w, we can construct a term of type **entity**, i.e., $\pi_1\pi_1 v(w)$. By replacing @ with $\pi_1\pi_1 v(w)$ in (11), we obtain the fully-specified representation for (10), which captures the intended dependent interpretation. This analysis is in contrast to that in DRT, where discourse referents are limited to individuals and thus an additional machinery is needed to capture the dependency relation (cf. Krifka, 1996). For more discussion on other kinds of Π-type anaphora including plural anaphora, see Tanaka et al. (2017b).

[5]See also Yana et al. (2017) for a detailed comparison of DTS and DRT.

[6]The idea that the reference of an anaphoric expression can be constructed via inference originates from Krahmer and Piwek (1999). It should be noted here that a naive implementation of this idea leads to a problem similar to *formal link* between a pronoun and an antecedent in the context of E-type approaches to pronouns (Heim, 1990); that is, the theory over-generates in the case of (ii).

(i) A man has a wife. She is sitting next to him.

(ii) * A man is married. She is sitting next to him.

In contrast to (i), the pronoun *she* in (ii) cannot anaphorically refer to the man's wife, because there is no NP antecedent. If there is knowledge that every married man has a wife, however, the referent corresponds to the man's wife can be constructed via inference. It is beyond the scope of this paper to discuss this issue in detail.

[7]This example is taken from Ranta (1994, p. 97), attributed to Lauri Karttunen in Hintikka and Carlson (1979).

Expression	Semantic Representation	Type
a, *some*	$\lambda F \lambda G. (u : (x : \mathbf{e}) \times F(x)) \times G(\pi_1 u)$	$(\mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}$
every	$\lambda F \lambda G. (u : (x : \mathbf{e}) \times F(x)) \to G(\pi_1 u)$	$(\mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}$
the	$\pi_1(@ : (x : \mathbf{e}) \times F(x))$	$\mathbf{e}$
and	$\lambda P \lambda Q. (u : P) \times Q$	$\mathbf{t} \to \mathbf{t} \to \mathbf{t}$
if	$\lambda P \lambda Q. (u : P) \to Q$	$\mathbf{t} \to \mathbf{t} \to \mathbf{t}$
man	$\lambda x.\mathbf{man}(x)$	$\mathbf{e} \to \mathbf{t}$
mother	$\lambda y \lambda x.\mathbf{motherOf}(x, y)$	$\mathbf{e} \to \mathbf{e} \to \mathbf{t}$
enter	$\lambda x.\mathbf{enter}(x)$	$\mathbf{e} \to \mathbf{t}$
receive	$\lambda y \lambda x.\mathbf{receive}(x, y)$	$\mathbf{e} \to \mathbf{e} \to \mathbf{t}$
he, *she*, *it*	$@ : \mathbf{e}$	$\mathbf{e}$
his, *her*	$\lambda R.\pi_1(@ : (x : \mathbf{e}) \times R(x, @ : \mathbf{e}))$	$(\mathbf{e} \to \mathbf{e} \to \mathbf{t}) \to \mathbf{e}$

Figure 2: Some lexical entries.

2.4 Possessive presupposition

Before moving on to the analysis of paycheck sentences, let us explain what SR and presupposition are associated with possessive NPs. Consider the sentence (13), whose SR is given in (14).[8]

(13) His mother left.

(14) $$\mathbf{leave}\left(\pi_1\left(@_1 : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{motherOf}(x, @_2 : \mathbf{entity}) \end{bmatrix}\right)\right)$$

Here, underspecified terms that correspond to distinct proof terms are indexed with different natural numbers. In (14), $@_2$ is embedded in the annotated type of $@_1$. To prove that the SR in (14) is a type under the global context $\mathcal{K}$, the following two judgments (15) and (16) should hold for $@_1$ and $@_2$, respectively.

(15) $$\mathcal{K} \vdash \left(@_1 : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{motherOf}(x, @_2 : \mathbf{entity}) \end{bmatrix}\right) : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{motherOf}(x, @_2 : \mathbf{entity}) \end{bmatrix}$$

(16) $\mathcal{K}, x : \mathbf{entity} \vdash (@_2 : \mathbf{entity}) : \mathbf{entity}$

Applying the @-rule to (15) requires to prove the following:

(17) $$\begin{bmatrix} x : \mathbf{entity} \\ \mathbf{motherOf}(x, @_2 : \mathbf{entity}) \end{bmatrix} : \mathbf{type}$$

The embedded underspecified term $@_2$ is involved here, so that we first need to resolve $@_2$ in (16); this triggers proof search to find a term of **entity**. If such a term, say *john* of type **entity**, is obtained, the type annotated to $@_1$ becomes $(x : \mathbf{entity}) \times \mathbf{motherOf}(x, john)$. This means that the presupposition of (13) is predicted to be *John has a mother*.

The derivations of initial underspecified SRs can be formalized in a fully compositional way. Figure 2 shows some of the lexical entries, where we abbreviate type **type** as **t** and **entity** as **e**.[9] Note that we assume that in initial underspecified SRs, each occurrence of @ is assigned a mutually distinct index. We only give SRs and their types, abstracting away from particular syntactic theories. It is fairly straightforward to compositionally derive the SRs we discussed so far, using these lexical entries. We skip the detailed semantic composition steps due to space limitations.

[8]An example like *mother* is what Barker (1995) calls *lexical* possession, where the possession relation derives from the lexical meaning of the noun, in contrast to *extrinsic* possession, where the possession relation is determined depending on context. For possessive NPs with extrinsic possession like *his paycheck*, we can specify the lexical entry for possessive pronouns as $\lambda F.\pi_1(@_i : (x : \mathbf{e}) \times (F(x) \times \mathbf{own}(@_j : \mathbf{e}, x))$, which is of type $(\mathbf{e} \to \mathbf{t}) \to \mathbf{e}$. Here we simply assume that the possession relation is the 'owning' relation; instead we can also represent such an contextually determined relation by using a higher-order underspecified term @ of type $\mathbf{e} \to \mathbf{e} \to \mathbf{t}$.

[9]Note also that we write a two-place predicate $\mathbf{receive}(y)(x)$ as $\mathbf{receive}(x, y)$, and so on.

3 Paycheck sentences as Π-type anaphora

This section provides an analysis of paycheck sentences in DTS. We will show that the interpretation of paycheck pronouns can be regarded as an instance of Π-type anaphora by taking into account the presuppositions of paycheck sentences.

3.1 Simple paycheck sentences

Let us consider the following simpler paycheck sentence for illustration.

(18) Every boy loves his mother. Mary hates her.

The SRs for the first and second sentences are given as (19) and (20), respectively. Note that the paycheck pronoun *her* is represented in the same way as the standard referential pronoun. The term *mary* is of type **entity**, which is assumed to be in the global context.

(19) $$\left(u : \begin{bmatrix} x : \textbf{entity} \\ \textbf{boy}(x) \end{bmatrix}\right) \to \textbf{love}\left(\pi_1 u,\ \pi_1 \left(@_1 : \begin{bmatrix} y : \textbf{entity} \\ \textbf{motherOf}(y, @_2 : \textbf{entity}) \end{bmatrix}\right)\right)$$

(20) $\textbf{hate}(mary, @_3 : \textbf{entity})$

The type checking procedure goes as follows. First, the following judgment should hold under the felicity condition of the first sentence.

(21) $$\mathcal{K} \vdash \left(u : \begin{bmatrix} x : \textbf{entity} \\ \textbf{boy}(x) \end{bmatrix}\right) \to \textbf{love}\left(\pi_1 u,\ \pi_1 \left(@_1 : \begin{bmatrix} y : \textbf{entity} \\ \textbf{motherOf}(y, @_2 : \textbf{entity}) \end{bmatrix}\right)\right) : \textbf{type}$$

To prove this judgment, we first need a proof term that can replace $@_2$. We will focus on the reading where *his* is bound by *every boy*, thus $@_2$ is substituted with $\pi_1 u$ of type **entity**. The next step is to prove the following judgment for $@_1$.

(22) $$\mathcal{K},\ u : \begin{bmatrix} x : \textbf{entity} \\ \textbf{boy}(x) \end{bmatrix} \vdash \left(@_1 : \begin{bmatrix} y : \textbf{entity} \\ \textbf{motherOf}(y, \pi_1 u) \end{bmatrix}\right) : \begin{bmatrix} y : \textbf{entity} \\ \textbf{motherOf}(y, \pi_1 u) \end{bmatrix}$$

What is required is to find a proof that can replace $@_1$. Although the current context does not supply for such a proof term, the proof search succeeds if one can assume, by presupposition accommodation, that there is a proof term p of the proposition *everyone has a mother*.

(23) $$\mathcal{K},\ p : (x : \textbf{entity}) \to \begin{bmatrix} y : \textbf{entity} \\ \textbf{motherOf}(y, x) \end{bmatrix},\ u : \begin{bmatrix} x : \textbf{entity} \\ \textbf{boy}(x) \end{bmatrix} \vdash \begin{bmatrix} y : \textbf{entity} \\ \textbf{motherOf}(y, \pi_1 u) \end{bmatrix} \mathit{true}$$

Note that this universal presupposition is not derived from the uniqueness presupposition associated with the paycheck pronoun *his mother* but from the structure of the proof step in (22), where the underspecified term contains a free variable that is bound by an outside quantifier. Thus the same functional proof term can be added to the context in a case where the outside quantifier is existential. For instance, the present theory gives the same prediction for a case in which the first sentence in (18) is replaced by *Some boy loves his mother*, which also gives rise to the paycheck reading.[10]

By adding p into the context, the term $p(\pi_1 u)$ of type $(y : \textbf{entity}) \times \textbf{motherOf}(y, \pi_1 u)$ can be constructed. Thus, one obtains the following fully-specified representation by substitution.

(24) $$\left(u : \begin{bmatrix} x : \textbf{entity} \\ \textbf{boy}(x) \end{bmatrix}\right) \to \textbf{love}\,(\pi_1 u,\ \pi_1 p(\pi_1 u))$$

The second sentence in (18) is interpreted subsequently. Again, the following judgment should hold for the whole mini-discourse.

[10]This agrees with Heim's (1983) prediction for the presuppositions of existentially quantified sentences. Although it has been debated whether universal inference is supported by existential quantifiers (see Sudo (2012) and the references cited there), we accept this universal presupposition as one plausible way to fulfill the felicity condition, perhaps with an additional assumption on domain restrictions.

(25) $\mathcal{K},\, p : (x : \mathbf{entity}) \to \begin{bmatrix} y : \mathbf{entity} \\ \mathbf{motherOf}(y, x) \end{bmatrix} \vdash \begin{bmatrix} v : \left(u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{boy}(x) \end{bmatrix}\right) \to \mathbf{love}\,(\pi_1 u,\, \pi_1 p(\pi_1 u)) \\ \mathbf{hate}(mary, @_3 : \mathbf{entity}) \end{bmatrix} : \mathbf{type}$

The underspecified term $@_3$ corresponds to the paycheck pronoun *her*, which is resolved in the following context.

(26) $\mathcal{K},\, p : (x : \mathbf{entity}) \to \begin{bmatrix} y : \mathbf{entity} \\ \mathbf{motherOf}(y, x) \end{bmatrix},\, v : \left(u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{boy}(x) \end{bmatrix}\right) \to \mathbf{love}\,(\pi_1 u,\, \pi_1 p(\pi_1 u)) \vdash (@_3 : \mathbf{entity}) : \mathbf{entity}$

Just as we do not have an antecedent expression for the pronoun in (18), there is no term of type **entity** that the context directly supplies. At this stage, however, we already have an additional function p, which has been introduced into the context to interpret the first sentence. This function is exactly the 'paycheck' function. By applying the paycheck function p to a term $mary$ of type **entity**, we obtain term $p(mary)$: $(y : \mathbf{entity}) \times \mathbf{motherOf}(y, mary)$, from which we obtain the term $\pi_1 p(mary)$ of type **entity**. By substituting $@_3$ with this term, we obtain the fully-specified representation for (18), which is the interpretation of our interest.

(27) $\begin{bmatrix} v : \left(u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{boy}(x) \end{bmatrix}\right) \to \mathbf{love}\,(\pi_1 u,\, \pi_1 p(\pi_1 u)) \\ \mathbf{hate}(mary, \pi_1 p(mary)) \end{bmatrix}$

This representation corresponds to the reading *Mary hates her mother*.

The analysis proposed here shares with the previous approaches (Cooper, 1979; Engdahl, 1986) the view that the reference of a paycheck pronoun is an application of the 'paycheck' function to an individual. Note that our account does not require any additional lexical entry for pronouns nor any type-shifting rule for deriving the paycheck reading. The resolution of paycheck anaphora is achieved using a function as a discourse referent introduced by Π-types, in combination of the context-passing mechanism realized by Σ-types. We also argue that the presupposition of a possessive NP that contains a free variable is taken as a source of the 'paycheck' function. In contrast to typical examples of Π-type anaphora where a Π-type explicitly appears as a preceding sentence, the relevant function arises from the presuppositional inference in paycheck constructions.[11] The process of anaphora resolution is couched within the overall architecture of DTS where anaphora and presupposition resolution involve inferences, i.e., processes of constructing proof terms for underspecified terms.

3.2 Generalizing to n-place functions

In the previous section, we consider the case where a one-place 'paycheck' function takes one individual as argument. It is observed that there is a case where a n-place function is applied to n individuals (Cooper,

[11] An anonymous reviewer suggested that there is a notable empirical difference between quantificational subordination and paycheck anaphora. In the case of quantificational subordination, the pronoun must be in the scope of an operator whose restrictor is a subset of the domain of the Π-type. Thus, (iii) is ungrammatical.

(iii) * If every third-grade boy receives a present, every forth-grade boy will open it.

By contrast, the situation is more flexible in the case of paycheck sentences. The following paycheck sentences are both acceptable.

(iv) a. Every third-grade loves his mother. Every forth-grade boy hates her. (Jacobson, 2012)
b. If every butcher spends their paycheck, every baker will deposit it.

In our analysis, the difference lies between the case where a function is explicitly introduced by a Π-type and the case where a function is derived from presuppositional inferences. The former corresponds to quantificational subordination and the latter to paycheck constructions. Note that universal inferences induced by presuppositions are often flexible; thus, (iv) could give rise to the presupposition *Everyone (in the domain) has a mother* or a weaker one such as *Every third grade boy has a mother*. This view is consistent with the observation made on the semi-conditional presupposition (Geurts, 1999):

(v) If John is a scuba diver and he wants to impress his girlfriend, he'll bring his wetsuit.

This sentence gives rise to a presupposition *If John is a scuba diver, he has a wetsuit*, rather than a fully conditional one *If John is a scuba diver and he wants to impress his girlfriend, he has a wetsuit.* In both cases of (iv) and (v), the restrictor of the universal proposition, i.e., the domain of the function introduced, is not linguistically fixed but contextually determined. This is in sharp contrast to the case of Π-type anaphora in quantificational subordination.

1979; Engdahl, 1986; Jacobson, 2000). The following is taken from Jacobson (2000, p. 132).[12]

(28) The woman^1 who told Sears^2 that the money she$_1$ owned them$_2$ was in the mail was wiser than the woman^3 who told Filene's^4 that <u>it</u> had not yet been mailed.

The paycheck pronoun *it* is interpreted as *the money she$_3$ owned them$_4$*.

It is straightforward to generalize the present account to the n-place function case. The sentence of (28) is schematically represented as follows.

(29) [... X^1 ... Y^2 ... [$_{\text{PT}}$ ... PRON$_1$ ... PRON$_2$ ...]].
[... Z ... W ... it ...].

Here, X, Y, Z, and W are entity denoting expressions, Z and W being parallel to X and Y, respectively. PT is a presupposition trigger that embeds pronouns PRON$_1$ and PRON$_2$, which are bound by X^1 and Y^2, respectively. In the case of (28), PT is a definite description. Each of the anaphoric expressions, PRON$_1$, PRON$_2$, and the definite description PT, introduces underspecified terms: $@_1$, $@_2$, and $@_3$, respectively. We will then obtain the following judgment for $@_3$.

(30) $\mathcal{K}, ..., x : \textbf{entity}, ..., y : \textbf{entity}, ... \vdash @_3 : \varphi(@_1, @_2)$

Here, $x : \textbf{entity}$ and $y : \textbf{entity}$ are introduced by X^1 and Y^2, respectively, and $\varphi(@_1, @_2)$ is a type annotation of $@_3$, which embeds two underspecified terms corresponding to PRON$_1$ and PRON$_2$.

Suppose that the embedded underspecified terms have been resolved according to the intended reading, namely, $@_1 = x$ and $@_2 = y$.

(31) $\mathcal{K}, ..., x : \textbf{entity}, ..., y : \textbf{entity}, ... \vdash @_3 : \varphi(x, y)$

Then we have to find a proof term of type $\varphi(x, y)$. This is the same situation encountered for (22). Thus, one plausible way to resolve the presupposition is to add a function $p : (x : \textbf{entity}) \to (y : \textbf{entity}) \to \varphi(x, y)$ to the context. In this way, we can obtain the two-place function. The resolution of the paycheck pronoun in this case is thus an instance of Π-type anaphora where a two-place function is used to construct an antecedent proof term. Here again, the representation of paycheck pronoun is same as a standard pronoun, which searches for a term of **entity**. The required term can be constructed by applying the two-place function to two individuals Z and W. This analysis can be applied to the cases where PT is the other presupposition trigger such as possessive NPs.

4 Conclusion

In this paper, we proposed an analysis of paycheck sentences in DTS. By taking into account the presuppositions of paycheck sentences, we showed that the resolution of paycheck anaphora is formalized as an instance of the resolution of Π-type anaphora. Our analysis is based on the analysis of anaphora and presupposition in the standard framework of DTS, which has independent motivations. Also, it has an advantage in that there is no need to extend the basic system with additional machineries for handling paycheck pronouns.

References

Barker, C. (1995) *Possessive Descriptions.* CSLI Publications.

Bekki, D. (2014) "Representing Anaphora with Dependent Types", In: N. Asher and S. Soloviev (eds.): *Logical Aspects of Computational Linguistics 2014*, LNCS 8535. Springer, pp.14–29.

Bekki, D. and K. Mineshima. (2017) "Context-Passing and Underspecification in Dependent Type Semantics", In: S. Chatzikyriakidis and Z. Luo (eds.): *Modern Perspectives in Type Theoretical Semantics*, Studies in Linguistics and Philosophy. Springer, pp.11–41.

[12] Another possible interpretation of this sentence is that the pronoun *it* is coreferential with *the money she$_1$ owned them$_2$*. It is straightforward to account for this reading in terms of presupposition binding.

Charlow, S. (2017) “A modular theory of pronouns and binding”, In the Proceedings of *Proceedings of Logic and Engineering of Natural Language Semantics 14 (LENLS14).*

Chatzikyriakidis, S. and Z. Luo. (2014) “Natural Language Inference in Coq”, *Journal of Logic, Language and Information* **23**(4), pp.441–480.

Cooper, R. (1979) “The interpretation of pronouns”, *Syntax and Semantics* **10**, pp.61–92.

Cooper, R. (2005) “Records and Record Types in Semantic Theory”, *Journal of Logic and Computation* **15**(2), pp.99–112.

Engdahl, E. (1986) *Constituent Questions.* Reidel, Dordrecht.

Geurts, B. (1999) *Presuppositions and Pronouns.* Amsterdam, Elsevier.

Groenendijk, J. and M. Stokhof. (1991) “Dynamic Predicate Logic”, *Linguistics and Philosophy* **14**(1), pp.39–100.

Heim, I. (1983) “On the Projection Problem for Presuppositions”, In: M. Barlow, D. Flickinger, and M. Wescoat (eds.): *Proceedings of WCCFL 2.* Stanford, CA, Stanford University, pp.114–125.

Heim, I. (1990) “E-Type Pronouns and Donkey Anaphora”, *Linguistics and Philosophy* **13**(2), pp.137–177.

Hintikka, J. and L. Carlson. (1979) “Conditionals, Generic Quantifiers, and Other Applications of Subgames”, In: E. Saarinen (ed.): *Game-Theoretical Semantics.* Springer, pp.179–214.

Jacobson, P. (2000) “Paycheck pronouns, Bach-Peters sentences, and variable-free semantics”, *Natural Language Semantics* **8**(2), pp.77–155.

Jacobson, P. (2012) “Direct Compositionality and ‘Uninterpretability’: The Case of (Sometimes) ‘Uninterpretable’ Features on Pronouns”, *Journal of Semantics* **29**(3), pp.305–343.

Kamp, H. and U. Reyle. (1993) *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory.* Springer.

Krahmer, E. and P. Piwek. (1999) “Presupposition Projection as Proof Construction”, In: H. Bunt and R. Muskens (eds.): *Computing Meaning*, Studies in Linguistics & Philosophy. Dordrecht, Kluwer Academic Publishers, pp.281–300.

Krifka, M. (1996) “Parametrized sum individuals for plural anaphora”, *Linguistics and Philosophy* **19**(6), pp.555–598.

Luo, Z. (2012) “Formal semantics in modern type theories with coercive subtyping”, *Linguistics and Philosophy* **35**(6), pp.491–513.

Martin-Löf, P. (1984) *Intuitionistic Type Theory: Notes by Giovanni Sambin of a series of lectures given in Padua, June 1980.* Bibliopolis.

Ranta, A. (1994) *Type-Theoretical Grammar.* Oxford University Press.

Sudo, Y. (2012) “On the semantics of phi features on pronouns”, Ph.D. thesis, MIT.

Sundholm, G. (1986) “Proof Theory and Meaning”, In: D. M. Gabbay and F. Guenthner (eds.): *Handbook of Philosophical Logic*, Vol. 3. Dordrecht, Reidel, pp.471–506.

Tanaka, R., K. Mineshima, and D. Bekki. (2017a) “Factivity and Presupposition in Dependent Type Semantics”, *Journal of Language Modelling* **5**(2), pp.385–420.

Tanaka, R., K. Mineshima, and D. Bekki. (2017b) “On the Interpretation of Dependent Plural Anaphora in a Dependently-Typed Setting”, In: S. Kurahashi, Y. Ohta, S. Arai, K. Satoh, and D. Bekki (eds.): *New Frontiers in Artificial Intelligence.* Springer, pp.123–137.

van der Sandt, R. A. (1992) “Presupposition Projection as Anaphora Resolution”, *Journal of Semantics* **9**, pp.333–377.

Yana, Y., K. Mineshima, and D. Bekki. (2017) “Variable Handling in DRT and DTS”, In the Proceedings of *Proceedings of the Workshop on Logic and Algorithms in Computational Linguistics 2017 (LACompLing2017).* pp.131–159.