Paychecks, Presupposition, and Dependent Types

Ribeka Tanaka, Koji Mineshima, and Daisuke Bekki

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 *it* does not refer to a particular individual but is interpreted as *his paycheck*, where *his* is anaphoric on *the man*[2]. Similarly, in (2), the second sentence is interpreted as saying *everyone put his paycheck in the bank*.

(1) The man[1] who gave his$_1$ paycheck to his$_1$ wife was wiser than the man[2] who gave it to his$_2$ mistress. (Karttunen, 1969, p. 114)

(2) John[1] gave his$_1$ paycheck to his$_1$ mistress. Everybody else[2] put it in the bank. (Cooper, 1979, p. 77)

In this paper, we account for the anaphora resolution 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 pronouns such as (1) and (2) and to show that it gives an adequate analysis of paycheck pronouns without introducing 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. No complex meaning (cf. Cooper, 1979; Engdahl, 1986) nor additional 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) and (2), the 'paycheck' function is derived from the *presupposition* of the possessive NP that contains a free variable; e.g., *his paycheck* in (1) and (2).

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 pronouns. 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.

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*.[1] 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 $\left[\begin{array}{l} x : A \\ B \end{array}\right]$.

Some remarks are in order about what each type mean.

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 introduced object is accessible to the later sentences. As we will see, Σ-types capture this property. We call the type of anaphora in which an object (discourse referent) introduced by *existential* quantifier is referred to from the subsequent discourse *Σ-type anaphora*.

As an illustration, consider the case of singular E-type anaphora, which is a typical case of Σ-type anaphora. In DTS, the sentences (3a) and (4a) are given the SRs (3b) and (4b), respectively.

(3) a. A man entered.

[1] 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.

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

(4) a. He whistled.

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

The underspecified term $@ : \mathbf{entity}$ in (4b) is introduced by the pronoun *he*.[2] 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 (3a) and (4a) is given the semantic representation shown in (5), by conjoining the two SRs (3b) and (4b) in terms of Σ-type.

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

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 (6).

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

The argument of the predicate **whistle** (namely, the place of @) is to be filled by the term $\pi_1\pi_1 v$. Note that the SR (6) 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 (6) from (5). 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: the requirement that the semantic representation of a sentence be a *type* under the given context. Thus, the following judgment must hold in the present case.

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

$\mathcal{K}$ is called the *global context*, which corresponds to lexical and world knowledge. Type checking follows the inference rules in dependent type theory and a rule for @-operator, called @-rule.

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

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 (7) can be given as follows.

$$\dfrac{\dfrac{\dfrac{}{\mathcal{K},\, v : \left[\begin{array}{l} u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right] \\ \mathbf{enter}(\pi_1 u) \end{array}\right] \vdash \mathbf{entity} : \mathbf{type}} \quad \mathcal{K},\, v : \left[\begin{array}{l} u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right] \\ \mathbf{enter}(\pi_1 u) \end{array}\right] \vdash \mathbf{entity}\, true}{\mathcal{K},\, v : \left[\begin{array}{l} u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right] \\ \mathbf{enter}(\pi_1 u) \end{array}\right] \vdash (@ : \mathbf{entity}) : \mathbf{entity}}\,@ \quad \mathcal{K},\, v : \left[\begin{array}{l} u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right] \\ \mathbf{enter}(\pi_1 u) \end{array}\right] \vdash \mathbf{whistle}(@ : \mathbf{entity})}{\mathcal{K} \vdash \left[\begin{array}{l} v : \left[\begin{array}{l} u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right] \\ \mathbf{enter}(\pi_1 u) \end{array}\right] \\ \mathbf{whistle}(@ : \mathbf{entity}) \end{array}\right] : \mathbf{type}}\,\Sigma E$$

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 (ii) is repeated here.

$$(8)\quad \mathcal{K},\, v : \left[\begin{array}{l} u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right] \\ \mathbf{enter}(\pi_1 u) \end{array}\right] \vdash \mathbf{entity}\ true$$

This judgment launches a process of proof search. In this case, we can show that a proof term $\pi_1 \pi_1 v$ inhabits the type **entity**. Then, by substitute this term for @, one obtains the fully-specified representation in (6), which corresponds to the intended reading in question.

The final representation shows how Σ-type captures the externally-dynamic aspects of existential sentences. 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 (4b) is followed by the sentence (9) with a definite description.

(9) The man whistled.

The SR of (9) is given as follows.

$$(10)\quad \mathbf{whistle}\left(\pi_1\left(@ : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right]\right)\right)$$

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 such 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).[3] For more details about an analysis of presupposition projection in DTS including the cases of filtering and bridging inferences, see Bekki and Mineshima (2017).[4]

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

[4] The idea that the reference of an anaphoric expression can be constructed via inference originates from Krahmer

2.3 Π-type anaphora

The *externally static* property of universal quantifiers can be captured by Π-types. Since a term of Π-type is a function rather than a pair, there is no way to directly access to its part. One difference from the standard treatment in dynamic semantics is that universal quantifier also introduces an object: a Π-type introduces a function as a discourse referent in the context (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. An example goes as follows[5].

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

Here the pronoun *it* in the consequent clause can refer to the present that the boy in question received. In other words, the pronoun receives the interpretation which depends on *some boy*. The semantic representation of (11) is given in (12).

$$(12)\quad \left(v : \left(\left(u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{boy}(x) \end{array}\right]\right) \to \left[\begin{array}{l} p : \left[\begin{array}{l} y : \mathbf{entity} \\ \mathbf{present}(y) \end{array}\right] \\ \mathbf{receive}(\pi_1 u, \pi_1 p) \end{array}\right]\right)\right) \to \left[\begin{array}{l} w : \left[\begin{array}{l} z : \mathbf{entity} \\ \mathbf{boy}(z) \end{array}\right] \\ \mathbf{open}(\pi_1 z, @ : \mathbf{entity}) \end{array}\right]$$

According to type checking, the following judgment should hold.

$$(13)\quad \mathcal{K}, v : \left(u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{boy}(x) \end{array}\right]\right) \to \left[\begin{array}{l} p : \left[\begin{array}{l} y : \mathbf{entity} \\ \mathbf{present}(y) \end{array}\right] \\ \mathbf{receive}(\pi_1 u, \pi_1 p) \end{array}\right], w : \left[\begin{array}{l} z : \mathbf{entity} \\ \mathbf{boy}(z) \end{array}\right] \vdash (@ : \mathbf{entity}) : \mathbf{entity}$$

Then the question is whether the type **entity** is inhabited under this given context. In this case, by applying the term w to the function v, one can construct a term of type **entity**, $\pi_1\pi_1 v(w)$. By replacing @ with the constructed term, we will obtain the fully-specified representation that captures the intended dependent interpretation.[6]

2.4 Possessive presupposition

Before moving on to the analysis of paycheck sentences, let us explain the semantic representation and presupposition associated with possessives. The sentence (14) is represented as (15)[7].

(14) His paycheck expires.

$$(15)\quad \mathbf{expire}\left(\pi_1\left(@_1 : \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}(@_2 : \mathbf{entity}, y) \end{array}\right] \end{array}\right]\right)\right)$$

Here, underspecified terms that correspond to distinct proof terms are indexed with different natural numbers. In (15), $@_2$ is embedded in the annotated type of $@_1$, and both are introduced by the possessive pronoun *his*. In order for (15) to be a type under the global context $\mathcal{K}$, the following two judgments (16) and (17) should hold for $@_1$ and $@_2$, respectively.

---

and Piwek (1999). If we naively approve inference based on wide-range knowledge, however, it leads to a problem similar to *formal link* between pronoun and antecedent (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), *she* in (ii) cannot mean the man's wife, since there is no NP antecedent. However, if there is knowledge that every married man has a wife, then the referent corresponds to the man's wife can be constructed via inference. A number of authors proposed the solutions for this problem of formal link in D-type theories, and it is possible to adopt those solutions in our proof-theoretic framework. Since this is a general issue in DTS and other related frameworks, we leave it for other occasions.

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

[6]For more discussion on other Π-type anaphora such as quantificational subordination and plural anaphora in the framework of DTS, see Tanaka et al. (2017).

[7]This case is what Barker (1995) calls *extrinsic* possession, where the possession relations is determined depending on context. Although we can represent such an contextually determined relation by using underspecified terms @, here we simply assume that the possessives in our examples express an 'owning' relation.

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	$\lambda x.\mathbf{man}(x)$	$\mathbf{e} \to \mathbf{t}$
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	$\lambda F.\pi_1(@_i : (x : \mathbf{e}) \times (F(x) \times \mathbf{own}(@_j : \mathbf{e}, x))$	$(\mathbf{e} \to \mathbf{t}) \to \mathbf{e}$

Figure 2: Some lexical entries.

$$(16)\quad \mathcal{K} \vdash \left(@_1 : \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}\,(@_2 : \mathbf{entity},\, y) \end{array} \right] \end{array} \right] \right) : \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}\,(@_2 : \mathbf{entity},\, y) \end{array} \right] \end{array} \right]$$

$$(17)\quad \mathcal{K},\, y : \mathbf{entity},\, p : \mathbf{paycheck}(y) \vdash (@_2 : \mathbf{entity}) : \mathbf{entity}$$

Recall the @-rule in §2.2: in order for (16) to hold, we first need to prove that $(y : \mathbf{entity}) \times (\mathbf{paycheck}(y) \times \mathbf{own}\,(@_2 : \mathbf{entity},\, y))$ is a type. This requirement leads us to the next judgment (17) for the embedded underspecified term $@_2$, which triggers the proof search to find a term of **entity**. After such a term, say *john* : **entity** for instance, being obtained, the annotated type of $@_1$ becomes to $(y : \mathbf{entity}) \times (\mathbf{paycheck}(y) \times \mathbf{own}\,(john,\, y))$. Then, in this case, the presupposition of the sentence is predicted to be *John owns a paycheck*.

The derivations of initial underspecified semantic representations 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**. We only give semantic representations and their types, abstracting away from particular syntactic theories. It is fairly straightforward to compositionally derive the semantic representations we discussed so far, using these lexical entries. We skip the detailed semantic composition steps due to space limitations.

3 Paycheck sentences as Π-type anaphora

This section provides an analysis of paycheck sentences in DTS. We will show that the resolution 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 sentence

Let us consider the following simpler paycheck sentence for illustration.

(18) Someone spent his paycheck. Mary kept it.

The semantic representation for the first and second sentences are given as (19) and (20), respectively. The paycheck pronoun *it* is represented in the same way as the standard referential pronoun. The term *mary* is of type **entity**, and we assume it is already defined in the global context.

$$(19)\quad \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{spend} \left(x,\, \pi_1 \left(@_1 : \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}\,(@_2 : \mathbf{entity},\, y) \end{array} \right] \end{array} \right] \right) \right) \end{array} \right]$$

$$(20)\quad \mathbf{keep}(mary, @_3 : \mathbf{entity})$$

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

$$(21)\quad \mathcal{K} \vdash \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{spend}\left(x,\, \pi_1\left(@_1 : \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}\,(@_2 : \mathbf{entity},\, y) \end{array}\right] \end{array}\right]\right)\right) \end{array}\right] : \mathbf{type}$$

To prove this judgment, we first need a proof term that can replace $@_2$. If we focus on the reading where *his* is co-indexed with *someone*, then $@_2$ can be substituted with x of **entity**. Then the next step is to prove the following judgment for $@_1$.

$$(22)\quad \mathcal{K},\, x : \mathbf{entity} \vdash \left(@_1 : \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}\,(x,\, y) \end{array}\right] \end{array}\right]\right) : \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}\,(x,\, y) \end{array}\right] \end{array}\right]$$

What is required next is to find a proof that can replace $@_1$. Although the current context does not supply for such a proof term, this proof search succeeds if, for instance, the hearer can assume that there is a proof term p of the proposition *everyone owns a paycheck*.[8]

$$(23)\quad \mathcal{K},\, p : (x : \mathbf{entity}) \to \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}(x,\, y) \end{array}\right] \end{array}\right],\, x : \mathbf{entity} \vdash \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}\,(x,\, y) \end{array}\right] \end{array}\right] \mathit{true}$$

In this case, the term px of type $(y : \mathbf{entity}) \times (\mathbf{paycheck}(y) \times \mathbf{own}\,(x,\, y))$ can be constructed. Thus, one obtains the following fully-specified representation by substitution.

$$(24)\quad \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{spend}(x,\, \pi_1 px) \end{array}\right]$$

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

$$(25)\quad \mathcal{K},\, p : (x : \mathbf{entity}) \to \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}(x,\, y) \end{array}\right] \end{array}\right] \vdash \left[\begin{array}{l} v : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{spend}(x, \pi_1 px) \end{array}\right] \\ \mathbf{keep}(\mathit{mary}, @_3 : \mathbf{entity}) \end{array}\right] : \mathbf{type}$$

The underspecified term $@_3$ corresponds to the paycheck pronoun *it*. The following judgment should hold.

$$(26)\quad \mathcal{K},\, p : (x : \mathbf{entity}) \to \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}(x,\, y) \end{array}\right] \end{array}\right],\, v : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{spend}(x, \pi_1 px) \end{array}\right] \vdash (@_3 : \mathbf{entity}) : \mathbf{entity}$$

Just as we do not have an antecedent expression for the pronoun in (18), the term $\pi_1 v$, which corresponds to *someone*, is the only 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 previously to interpret the first sentence. This function is exactly the 'paycheck' function. Thus, by applying an individual of type **entity** to this function p, one obtains a proof term of the individual owning a paycheck. Suppose that we apply p to a term $\mathit{mary} : \mathbf{entity}$ in $\mathcal{K}$ and obtain term $p(\mathit{mary}) : (y : \mathbf{entity}) \times (\mathbf{paycheck}(y) \times \mathbf{own}(\mathit{mary},\, y))$. Then, by applying a projection function, we obtain $\pi_1 p(\mathit{mary})$ of type **entity**. Since this can substitute $@_3$, the following fully-specified representation is obtained.

$$(27)\quad \mathcal{K},\, p : (x : \mathbf{entity}) \to \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{paycheck}(y) \\ \mathbf{own}(x,\, y) \end{array}\right] \end{array}\right] \vdash \left[\begin{array}{l} v : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{spend}(x, \pi_1 px) \end{array}\right] \\ \mathbf{keep}(\mathit{mary}, \pi_1 p(\mathit{mary})) \end{array}\right] : \mathbf{type}$$

[8] This corresponds to the global accommodation of the presupposition. Since *someone spent his paycheck* has a universal presupposition, this agrees with Heim's (1983) prediction.

This representation corresponds to the reading *Mary kept her paycheck*, which is the interpretation of our interest.

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 In our terms, the resolution of paycheck pronouns is taken as an instance of the resolution of Π-type anaphora. As mentioned in §1, the difference from the previous approaches is 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 possessives can be seen as a source of the 'paycheck' function. In contrast to typical examples of Π-type anaphora where a Π-type appears as a preceding sentence, it can be derived from presupposition in typical paycheck sentences. 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, 1979; Engdahl, 1986; Jacobson, 2000). The following is taken from Jacobson (2000, p. 132).

(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$_4$*.

The present account can be generalized to this n-place-function case. The construction of (28a) is schematically shown as follows.

(29) [... X^1 ... Y^2 ... [$_{PT}$ ... $PRON_1$ ... $PRON_2$ ...]].
[... Z ... W ... it ...].

X, Y, Z, and W are entity denoting expressions, where Z and W are somewhat parallel to X and Y, respectively. PT is a definite description which embeds pronouns $PRON_1$ and $PRON_2$. Each of the anaphoric expressions, $PRON_1$, $PRON_2$, and a definite article, introduce underspecified terms: say, $@_1$, $@_2$, and $@_3$. One will obtain the following judgment for $@_3$.

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

Terms x : **entity** and y : **entity** are introduced by X and Y, respectively, and φ is a type annotation of $@_3$, which embeds two underspecified terms originate from the pronouns.

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

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

Then the next step is to find a proof of type $\varphi(x, y)$. This is the same situation encountered in (22) in the previous section. 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 2-place function. Then, the same story as §3.1 applies to resolution of the paycheck pronoun. Again, we use the same lexical entry for the paycheck pronoun, which searches for term of **entity**. The required term can be constructed by applying the 2-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 an additional machineries for handling paycheck pronouns.

References

Chris Barker. Possessive Descriptions. CSLI Publications, 1995.

Daisuke Bekki. Representing anaphora with dependent types. In Nicholas Asher and Sergei Soloviev, editors, Logical Aspects of Computational Linguistics 2014, LNCS 8535, pages 14–29. Springer, 2014.

Daisuke Bekki and Koji Mineshima. Context-passing and underspecification in Dependent Type Semantics. In Stergios Chatzikyriakidis and Zhaohui Luo, editors, Modern Perspectives in Type Theoretical Semantics, Studies in Linguistics and Philosophy, pages 11–41. Springer, 2017.

Simon Charlow. A modular theory of pronouns and binding. In Proceedings of Logic and Engineering of Natural Language Semantics 14 (LENLS14), 2017.

Stergios Chatzikyriakidis and Zhaohui Luo. Natural language inference in Coq. Journal of Logic, Language and Information, 23(4):441–480, 2014.

Robin Cooper. The interpretation of pronouns. Syntax and Semantics, 10:61–92, 1979.

Robin Cooper. Records and record types in semantic theory. Journal of Logic and Computation, 15(2):99–112, 2005.

Elisabet Engdahl. Constituent Questions. Reidel, Dordrecht, 1986.

Jeroen Groenendijk and Martin Stokhof. Dynamic predicate logic. Linguistics and Philosophy, 14 (1):39–100, 1991.

Irene Heim. On the projection problem for presuppositions. In M. Barlow, D. Flickinger, and M. Wescoat, editors, Proceedings of WCCFL 2, pages 114–125. Stanford University, Stanford, CA, 1983.

Irene Heim. E-type pronouns and donkey anaphora. Linguistics and Philosophy, 13(2):137–77, 1990.

Jaakko Hintikka and Lauri Carlson. Conditionals, generic quantifiers, and other applications of subgames. In Esa Saarinen, editor, Game-Theoretical Semantics, pages 179–214. Springer, 1979.

Pauline Jacobson. Paycheck pronouns, Bach-Peters sentences, and variable-free semantics. Natural Language Semantics, 8(2):77–155, 2000.

Hans Kamp and Uwe Reyle. From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory. Springer, 1993.

Lauri Karttunen. Pronouns and variables. In R. Binnick et al., editor, Papers from the Fifth Regional Meeting of the Chicago Linguistics Society, pages 108–116, 1969.

Emiel Krahmer and Paul Piwek. Presupposition projection as proof construction. In H. Bunt and Reinhard Muskens, editors, Computing Meanings: Current Issues in Computational Semantics, Studies in Linguistics Philosophy Series. Kluwer Academic Publishers, Dordrecht, 1999.

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

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

Aarne Ranta. Type-Theoretical Grammar. Oxford University Press, 1994.

Göran Sundholm. Proof theory and meaning. In D. M. Gabbay and F. Guenthner, editors, Handbook of Philosophical Logic, volume 3, pages 471–506. Reidel, Dordrecht, 1986.

Ribeka Tanaka, Koji Mineshima, and Daisuke Bekki. On the interpretation of dependent plural anaphora in a dependently-typed setting. In Setsuya Kurahashi, Yuiko Ohta, Sachiyo Arai, Ken Satoh, and Daisuke Bekki, editors, New Frontiers in Artificial Intelligence, pages 123–137. Springer, 2017.

Rob A. van der Sandt. Presupposition projection as anaphora resolution. Journal of Semantics, 9:333–377, 1992.

Yukiko Yana, Koji Mineshima, and Daisuke Bekki. Variable handling in DRT and DTS. In Proceedings of the Workshop on Logic and Algorithms in Computational Linguistics 2017 (LACompLing2017), pages 131–159, 2017.