Appositive Projection as Implicit Context Extension in Dependent Type Semantics

Daiki Matsuoka[1] Daisuke Bekki[2] Hitomi Yanaka[1]

[1]The University of Tokyo [2]Ochanomizu University

1 Introduction

The content of an appositive relative clause (ARC) is a type of *conventional implicature* (CI), that is, secondary or supplementary (*not-at-issue*) information of an utterance [9], and it interacts with at-issue content in various interesting ways. Focusing on the projection behavior and the interaction with quantifiers, we present an analysis of ARCs based on *Dependent Type Semantics* (DTS) [4]. Our central idea is that appositive content implicitly extends the typing context during the process of type checking.

2 Data

2.1. Projection The content of an ARC projects out of entailment-canceling environments such as negation (1a) and conditional antecedent (1b).

(1) a. It is not the case that Ann, who danced, met John. $\Rightarrow$ Ann danced.
    b. If Ann, who danced, met John, Mary was happy. $\Rightarrow$ Ann danced.

Although it might thus seem that an ARC is semantically independent of its matrix clause, it has been suggested (e.g., [1]) that the two clauses should not be kept apart at the level of meaning representation because anaphoric dependencies can be established between them, as in (2):

(2) a. ARC $\rightarrow$ at-issue: John, who met a girl$_1$, smiled. She$_1$ danced.
    b. At-issue $\rightarrow$ ARC: A girl$_1$ danced. John, who met her$_1$, smiled.

2.2. Interaction with Quantifiers An ARC cannot modify a quantified NP unless the quantifier has the referential reading (see, e.g., [5]).

(3) {#Every/#No/A} girl, who met John, danced.

In parallel, pronouns inside ARCs cannot be bound by non-referential quantified NPs.

(4) {#Every/#No/A} girl$_1$ met John, who praised her$_1$.

We remark that *a girl* in the previous two examples is interpreted as a so-called *specific indefinite*, which basically has only the widest-scope reading. For instance, *a professor* in (5) takes wider scope than the conditional [8, 10].

(5) If a professor, who is famous, publishes a book, he will make a lot of money.

3 Framework

3.1. Overview DTS is a semantic framework based on dependent type theory, where a type represents the meaning of a sentence [4]. It accounts for anaphora and presupposition by reducing the felicity of a sentence to the well-formedness of its semantic representation (SR). The version of DTS that we use here [2] introduces an *underspecified type* (or an *@-type*) to represent anaphoric meaning in a compositional way.[1] The type is characterized by the following inference rule:

$$(@)\ \frac{\Gamma \vdash A : \mathsf{type} \qquad \Gamma \vdash M : A \qquad \Gamma \vdash B[x := M] : \mathsf{type}}{\Gamma \vdash (x \mathbin{@} A) \times B : \mathsf{type}} \quad \left(\xrightarrow{\text{@-elimination}} \Gamma \vdash B[x := M] : \mathsf{type} \right)$$

Intuitively, x is a placeholder for a concrete term M of type A. The term is searched for when we type check the @-type, after which we replace x with M and obtain an @-free type $B[x := M]$.

[1]Due to space limitations, we omit the process of semantic composition throughout this abstract.

For illustration, consider how the pronoun *she* is resolved in (6a). (6b) shows the *felicity condition* (FC) of the second sentence.[2] Note that the SR for the first sentence is in the typing context, meaning it has already entered the common ground.

(6) a. A girl came. She danced.

b. $s : \begin{bmatrix} u : \texttt{girl}^* \\ \texttt{come}(\pi_1 u) \end{bmatrix} \vdash \begin{bmatrix} v \mathbin{@} \texttt{female}^* \\ \texttt{dance}(\pi_1 v) \end{bmatrix} : \mathsf{type}$

To derive (6b), we need to find a term of $\texttt{female}^*$, which is required by the second premise of the rule (@). By introducing the world knowledge that every girl is female and accordingly assuming a constant $\texttt{g-to-f} : (u : \texttt{girl}^*) \to \texttt{female}(\pi_1 u)$, we can construct the term $\langle \pi_1\pi_1 s, \texttt{g-to-f}(\pi_1 s)\rangle$ of type $\texttt{female}^*$. Substituting this term for v, we obtain $\texttt{dance}(\pi_1\pi_1 s)$. This result correctly predicts that *she* can be bound by *a girl*, because $\pi_1\pi_1 s$ refers to the entity introduced by the first sentence.

3.2. @-elimination What is yet to be specified is how we can replace the variable of a @-type in tandem with the process of type checking. In a recent version of DTS [2], the type checking function $[\![-]\!]$ returns a set of derivation trees for a typing judgment, the clause for the @-type being as follows (see [2] for the clauses for other type constructors)[3]. Note that the value of $[\![-]\!]$ is not a single derivation because there can be multiple ways to construct a term of type A.

$$[\![\Gamma \vdash (x \mathbin{@} A) \times B : \mathsf{type}]\!] = \left\{ \mathcal{D}_3 \;\middle|\; \begin{array}{l} \mathcal{D}_1 \in [\![\Gamma \vdash A : \mathsf{type}]\!] \ \text{(Let } \mathcal{D}_1\text{'s root be } \Gamma \vdash A' : \mathsf{type}) \\ \mathcal{D}_2 \in [\![\Gamma \vdash M : A']\!] \ \text{for some term } M. \\ \mathcal{D}_3 \in [\![\Gamma \vdash \mathsf{nf}(B[x := M]) : \mathsf{type}]\!] \ \text{(Let } \mathcal{D}_2\text{'s root be } \Gamma \vdash B' : \mathsf{type}) \end{array} \right\}$$

The derivations of the three premises of the rule (@) are internally constructed, but $\mathcal{D}_1$ and $\mathcal{D}_2$ are discarded; only $\mathcal{D}_3$ is returned. That is, what we obtain as a result of type checking $(x \mathbin{@} A) \times B$ is an @-free type B', as if it were the type we wanted to check from the beginning. We can thus simultaneously perform type checking and eliminate @-types.

3.3. Two-stage Validation Before presenting our proposal, we must clarify how discourses are processed in DTS, which is crucial in distinguishing between at-issue and not-at-issue content. Importantly, type checking confirms only the FC of a sentence; it does not consider whether it is accepted by the addressee. In other words, whether the SR is added to the typing context is determined after it is type checked, based on some pragmatic factors.[4] In this paper, we assume two validation stages before an SR A is added to the context Γ.

(i) By calculating $[\![\Gamma \vdash A : \mathsf{type}]\!]$, we check the well-formedness of A under Γ and eliminate the @-types in A (suppose we obtain $\Gamma \vdash A' : \mathsf{type}$ as a result).

(ii) We check whether A' is acceptable under Γ. If it is true, we extend Γ with $x : A'$ ($x \notin \mathrm{FV}(\Gamma)$).

4 Proposal

4.1. CI Type We propose extending DTS with a new type $(x \lhd A) \times B$ (a *CI type*), characterized by the following inference rule:

$$(\lhd)\ \frac{\Gamma \vdash A : \mathsf{type} \qquad \Gamma, x : A \vdash B : \mathsf{type}}{\Gamma \vdash (x \lhd A) \times B : \mathsf{type}} \quad \left(\xrightarrow{\lhd\text{-elimination}} \Gamma, x : A \vdash B : \mathsf{type} \right)$$

As well as the @-type, elimination of the $\lhd$-type is defined in a clause for $[\![-]\!]$.

$$[\![\Gamma \vdash (x \lhd A) \times B : \mathsf{type}]\!] = \left\{ \mathcal{D}_2 \;\middle|\; \begin{array}{l} \mathcal{D}_1 \in [\![\Gamma \vdash A : \mathsf{type}]\!] \ \text{(Let } \mathcal{D}_1\text{'s root be } \Gamma, \Delta \vdash A' : \mathsf{type}) \\ \mathcal{D}_2 \in [\![\Gamma, \Delta, x : A' \vdash B : \mathsf{type}]\!] \end{array} \right\}$$

[2]We interchangeably use two notations for Σ-like operators such as $(x : A) \times B$ and $\begin{bmatrix} x : A \\ B \end{bmatrix}$. We also abbreviate $(x : \mathsf{e}) \times Px$ as P^*, where e is the type of entities and P is of type $\mathsf{e} \to \mathsf{type}$.

[3]$\mathsf{nf}(M)$ is the normal form of M (if any).

[4]Although it is beyond the scope of the present paper to characterize such factors, we at least require that the acceptance of the assertion should not contradict the preceding discourse (i.e., $\Gamma, x : A \nvdash \bot$).

Unlike the @-type, this type does not require a term of A for its well-formedness. Instead, it extends the context Γ with $x : A$ when it is eliminated after the two premises are derived. In other words, well-formedness of the $\lhd$-type requires the context to behave as if $x : A$ had already been introduced before B's well-formedness is checked.

Conceptually, the context extension launched by $(x \lhd A) \times B$ is *implicit* in that it leaves no room for the addressee to choose whether to accept or reject A. In fact, $x : A$ is added to the context while the felicity of the whole SR is being checked; A is not subject to the second validation stage. This behavior reflects the idea by [1] that appositive content is an *imposition* on the common ground, which is normally hard to respond to or negotiate on.

4.2. Permutation With the $\lhd$-type, computing $[\![\Gamma \vdash A : \mathsf{type}]\!]$ may change the original context (Γ) as well as the type (A). Hence, the above-mentioned two-stage validation needs to be revised as indicated by underlines below. Note that Δ is empty if A contains no $\lhd$-types.

(i) By calculating $[\![\Gamma \vdash A : \mathsf{type}]\!]$, we check the well-formedness of A under Γ and eliminate the @-types $\underline{\text{and } \lhd\text{-types}}$ in A (suppose we obtain $\Gamma\underline{, \Delta} \vdash A' : \mathsf{type}$ as a result).

(ii) We check whether A' is acceptable under $\Gamma\underline{, \Delta}$. If it is true, we extend $\Gamma\underline{, \Delta}$ with $x : A'$ $(x \notin \mathrm{FV}(\Gamma\underline{, \Delta}))$.

However, this revision clashes with other parts of the theory. Suppose that while type checking $(x : A) \to B$, we verified $\Gamma, x : A \vdash B : \mathsf{type}$, resulting in an extended context $\Gamma, x : A, y : C$ due to a $\lhd$-type inside B. We cannot apply the rule (ΠF) in such cases because $x : A$ is not on the right end of the context. With this motivation, we introduce the following structural rule (*permutation*), which is admissible in the dependent type theory on which DTS is based.

$$(\mathrm{perm})\ \frac{\Gamma, x : A, y : B, \Delta \vdash M : C}{\Gamma, y : B, x : A, \Delta \vdash M : C}\ (x \notin \mathrm{FV}(B))$$

The side condition $x \notin \mathrm{FV}(B)$ is required by the property of dependent type theory that a type can depend on terms: if $x : A$ occurs free in B, exchanging the two premises would result in an ill-formed context. We will see that this (independently motivated) restriction is important in explaining the interaction between an ARC and a quantified NP.

Using permutation, we can formally define the type checking algorithm for the Π-type as follows (the same applies to the Σ-type). Here, $\mathsf{Arrange}$ is a partial function that applies the rule (perm) to $\mathcal{D}$ and returns the derivation (if any) such that its root has the variable x at the right end of the context. If there is no such derivation, type checking fails because the result of $[\![-]\!]$ is empty.

$$[\![\Gamma \vdash (x : A) \to B : \mathsf{type}]\!] = \left\{ \begin{array}{c|l} (\Pi F)\,\dfrac{(\mathrm{wk})\,\dfrac{\vdots}{\Gamma, \Delta, \Theta \vdash A' : \mathsf{type}} \left((\mathrm{wk})\,\dfrac{\begin{matrix}\mathcal{D}_1 \\ \Gamma, \Delta \vdash A' : \mathsf{type}\end{matrix}}{} \right) \quad \begin{matrix}\mathcal{D}_3 \\ \Gamma, \Delta, \Theta, x : A' \vdash B' : \mathsf{type}\end{matrix}}{\Gamma, \Delta, \Theta \vdash (x : A') \to B' : \mathsf{type}} & \begin{array}{l} \mathcal{D}_1 \in [\![\Gamma \vdash A : \mathsf{type}]\!] \\ (\text{Let } \mathcal{D}_1\text{'s root be } \Gamma, \Delta \vdash A' : \mathsf{type}) \\ \mathcal{D}_2 \in [\![\Gamma, \Delta, x : A' \vdash B : \mathsf{type}]\!] \\ (\text{Let } \mathcal{D}_2\text{'s root be } \Gamma, \Delta, x : A', \Theta \vdash B' : \mathsf{type}) \\ \mathcal{D}_3 = \mathsf{Arrange}(\mathcal{D}_2, x) \end{array} \end{array} \right\}$$

5 Verification

5.1. Projection We demonstrate that the $\lhd$-type correctly predicts the behavior of ARCs. First, we verify their projectivity. (7) shows the FC of (1a).[5]

(7) $\vdash \left(v : \begin{bmatrix} u \lhd \mathtt{dance}(\mathtt{a}) \\ \mathtt{meet}(\mathtt{a}, \mathtt{j}) \end{bmatrix} \right) \to \bot : \mathsf{type}$

Figure 1 shows the process by which the FC is validated. Deriving the sub-goal (g1) results in an extended context $u : \mathtt{dance}(\mathtt{a})$, and this change is propagated to the other sub-goal (g2). After (g2) is derived, we apply the rule (ΠF) and obtain $u : \mathtt{dance}(\mathtt{a}) \vdash (v : \mathtt{meet}(\mathtt{a}, \mathtt{j})) \to \bot : \mathsf{type}$. The appositive content $\mathtt{dance}(\mathtt{a})$ has thus successfully projected out of the negation.[6]

[5]In DTS, negation $\neg A$ is defined as $(x : A) \to \bot$, where $\bot$ is the empty type [4].

[6]This claim is justified by the fact that $\mathtt{dance}(\mathtt{a})$ is inhabited under the updated context $u : \mathtt{dance}(\mathtt{a}), s : (v : \mathtt{meet}(\mathtt{a}, \mathtt{j})) \to \bot$. To be precise, we need a revised definition of an inference, which we will describe in the full paper.

The sub-goals to derive (7)

$$\boxed{\begin{array}{l}(\mathrm{g1})\ \vdash \begin{bmatrix} u \lhd \mathtt{d(a)} \\ \mathtt{m(a,j)}\end{bmatrix} : \mathsf{type} \\ (\mathrm{g2})\ v : \begin{bmatrix} u \lhd \mathtt{d(a)} \\ \mathtt{m(a,j)}\end{bmatrix} \vdash \bot : \mathsf{type}\end{array}} \xRightarrow[\mathcal{D}_1[\underline{u : \mathtt{d(a)}} \vdash \mathtt{m(a,j)} : \mathsf{type}]]{\text{Derive (g1)}} \boxed{(\mathrm{g2})\ \underline{u:\mathtt{d(a)}}, v : \mathtt{m(a,j)} \vdash \bot : \mathsf{type}}$$

$$\xRightarrow[\mathcal{D}_2[\cdots \vdash \bot : \mathsf{type}]]{\text{Derive (g2)}} \boxed{\text{(done)}} \left(\text{return}\quad (\Pi F)\ \dfrac{\overset{\mathcal{D}_1}{u:\mathtt{d(a)} \vdash \mathtt{m(a,j)}:\mathsf{type}} \qquad \overset{\mathcal{D}_2}{u:\mathtt{d(a)}, v:\mathtt{m(a,j)} \vdash \bot : \mathsf{type}}}{u:\mathtt{d(a)} \vdash (v:\mathtt{m(a,j)}) \to \bot : \mathsf{type}}\right)$$

The sub-goals to derive (g1)

$$\boxed{\begin{array}{l}(\mathrm{g3})\ \vdash \mathtt{d(a)} : \mathsf{type} \\ (\mathrm{g4})\ u : \mathtt{d(a)} \vdash \mathtt{m(a,j)} : \mathsf{type}\end{array}} \xRightarrow[\mathcal{D}_3[\vdash \mathtt{d(a)} : \mathsf{type}]]{\text{Derive (g3)}} \boxed{(\mathrm{g4})\ u : \mathtt{d(a)} \vdash \mathtt{m(a,j)} : \mathsf{type}} \xRightarrow[\mathcal{D}_4[u : \mathtt{d(a)} \vdash \mathtt{m(a,j)} : \mathsf{type}]]{\text{Derive (g4)}} \boxed{\text{(done)}}\ (\text{return } \mathcal{D}_4)$$

Figure 1: The process of validating the FC of (1a). We abbreviate the names of the predicates (e.g., $\mathtt{dance} \mapsto \mathtt{d}$). We also write $\mathcal{D}[J]$ for a derivation $\mathcal{D}$ with root J. The underlined parts indicate the context extension, and the gray boxes show the types transformed during type checking.

5.2. Anaphoric Dependencies Next, we check how anaphora crossing the boundary between at-issue and ARC content can be resolved. (8a) is the FC of the first sentence of (2a), whose validation extends the context with $v : [\cdots]$. Assuming the at-issue content $\mathtt{smile(j)}$ is accepted, the FC of the second sentence is (8b).[7] The @-type can be resolved with v in a way similar to (6b).

(8) a. $\vdash \begin{bmatrix} v \lhd \begin{bmatrix} u : \mathtt{girl}^* \\ \mathtt{meet}(\mathtt{j}, \pi_1 u)\end{bmatrix} \\ \mathtt{smile(j)} \end{bmatrix} : \mathsf{type} \xrightarrow{\text{type checking}} v : \begin{bmatrix} u : \mathtt{girl}^* \\ \mathtt{meet}(\mathtt{j}, \pi_1 u)\end{bmatrix} \vdash \mathtt{smile(j)} : \mathsf{type}$

b. $v : [\cdots], s : \mathtt{smile(j)} \vdash \begin{bmatrix} w \mathbin{@} \mathtt{female}^* \\ \mathtt{dance}(\pi_1 u)\end{bmatrix} : \mathsf{type} \xrightarrow{\text{type checking}} v : [\cdots], s : \cdots \vdash \mathtt{dance}(\pi_1\pi_1 v) : \mathsf{type}$

The other direction, shown in (2b), is more complicated. Provided that the first sentence is successfully type checked and accepted, the FC of the second sentence is as follows.

(9) $v : \begin{bmatrix} u : \mathtt{girl}^* \\ \mathtt{dance}(\pi_1 u)\end{bmatrix} \vdash \begin{bmatrix} z \lhd \begin{bmatrix} w \mathbin{@} \mathtt{female}^* \\ \mathtt{meet}(\mathtt{j}, \pi_1 w)\end{bmatrix} \\ \mathtt{smile(j)} \end{bmatrix} : \mathsf{type}$

Figure 2 shows the derivation process. In deriving the first sub-goal, we can resolve the @-type as in (8b),[8] and the result is reflected in the other sub-goal. Finally, we obtain a context extended with $z : \mathtt{meet}(\mathtt{j}, \pi_1\pi_1 v)$ (*John met her*), as expected.

The sub-goals to derive (9)

$$\boxed{\begin{array}{l}(\mathrm{g1})\ v : [\cdots] \vdash \begin{bmatrix} w \mathbin{@} \mathtt{f}^* \\ \mathtt{m}(\mathtt{j}, \pi_1 w)\end{bmatrix} : \mathsf{type} \\ (\mathrm{g2})\ v : [\cdots], z : \begin{bmatrix} w \mathbin{@} \mathtt{f}^* \\ \mathtt{m}(\mathtt{j}, \pi_1 w)\end{bmatrix} \vdash \mathtt{s(j)} : \mathsf{type}\end{array}} \xRightarrow[\mathcal{D}_1[v : [\cdots] \vdash \mathtt{m}(\mathtt{j}, \pi_1\pi_1 v) : \mathsf{type}]]{\text{Derive (g1)}} \boxed{(\mathrm{g2})\ v : [\cdots], z : \mathtt{m}(\mathtt{j}, \pi_1\pi_1 v) \vdash \mathtt{s(j)} : \mathsf{type}}$$

$$\xRightarrow[\mathcal{D}_2[v : [\cdots], z : \mathtt{m}(\mathtt{j}, \pi_1\pi_1 v) \vdash \mathtt{s(j)} : \mathsf{type}]]{\text{Derive (g2)}} \boxed{\text{(done)}}\ (\text{return } \mathcal{D}_2)$$

Figure 2: The process of validating the FC of (2b).

5.3. ARC + Non-referential Quantifier Let us turn to the interaction with quantifiers. Taking *every* as an example,[9] we show the FC of (3) in (10).

(10) $\vdash (u : \mathtt{girl}^*) \to \begin{bmatrix} v \lhd \mathtt{meet}(\pi_1 u, \mathtt{j}) \\ \mathtt{dance}(\pi_1 u)\end{bmatrix} : \mathsf{type}$

[7]Note that the context in (8b) is the same as when the first sentence is replaced with "John met a girl. He smiled." The difference lies in how $v : [\cdots]$ (*John met a girl*) is added to the context.

[8]If we could not resolve the @-type here, the whole SR would not be well-typed. Hence, our theory predicts that the infelicity of the appositive content leads to the infelicity of the whole sentence.

[9]The prediction is the same for "No girl ...," which is translated using a Π-type ($(u : \mathtt{girl}^*) \to \neg(\cdots)$).

The sub-goals to derive (10)

$$\boxed{\begin{array}{l}(\text{g1})\ \vdash \mathsf{g}^* : \mathsf{type} \\ (\text{g2})\ u : \mathsf{g}^* \vdash \begin{bmatrix} v \lhd \mathtt{m}(\pi_1 u, \mathtt{j}) \\ \mathtt{d}(\pi_1 u) \end{bmatrix} : \mathsf{type}\end{array}} \ (\cdots) \ \xRightarrow{\text{Derive (g2)}} \ \mathcal{D}_2[u : \mathsf{g}^*, \underline{v : \mathtt{m}(\pi_1 u, \mathtt{j})} \vdash \boxed{\mathtt{d}(\pi_1 u)} : \mathsf{type}] \quad (\text{perm})\ \dfrac{\overset{\mathcal{D}_2}{u : \mathsf{g}^*, v : \mathtt{m}(\pi_1 u, \mathtt{j}) \vdash \mathtt{d}(\pi_1 u) : \mathsf{type}}}{\times}$$

Figure 3: The process showing the infelicity of (3). The step for (g1) is omitted for brevity.

Figure 3 describes how its infelicity is predicted. When the sub-goal (g2) is derived, the context is extended with $v : \mathtt{meet}(\pi_1 u, \mathtt{j})$, in which u occurs free. This prevents the application of the rule (perm), so $u : \mathtt{girl}^*$ cannot be moved to the right end of the context, causing type checking to fail.

We can handle the case of binding in a similar way. Again using *every girl* as our example, we describe the FC of (4) in (11). We first need to resolve the @-type corresponding to *her* under the context $u : \mathtt{girl}^*$. We can replace w using u and obtain $(w \lhd \mathtt{praise}(\mathtt{j}, \pi_1 u)) \times \cdots$. Since u occurs free in the appositive content $\mathtt{praise}(\mathtt{j}, \pi_1 u)$, type checking fails as in (10).

$$(11) \quad \vdash (u : \mathtt{girl}^*) \to \begin{bmatrix} v \mathbin{@} \mathtt{female}^* \\ \begin{bmatrix} w \lhd \mathtt{praise}(\mathtt{j}, \pi_1 v) \\ \mathtt{meet}(\pi_1 v, \mathtt{j}) \end{bmatrix} \end{bmatrix} : \mathsf{type}$$

In summary, the appositive content $x \lhd A$ cannot project out if it depends on a variable introduced by Π or Σ. Importantly, this restriction derives from the side condition of the permutation rule, which is inherent in dependent type theory.

5.4. ARC + Specific Indefinite Finally, we check the case of specific indefinites. We assume that the determiner *a* with the specific reading is translated with a $\lhd$-type.[10] Then, the type checking of the SR for "A girl, who met John, danced" proceeds as described in (12). Since $u : \mathtt{girl}^*$, on which the appositive content $\mathtt{meet}(\pi_1 u, \mathtt{j})$ depends, is also implicitly added to the context, we need not apply (perm) and thus we can complete type checking.

$$(12) \quad \vdash \begin{bmatrix} u \lhd \mathtt{girl}^* \\ \begin{bmatrix} v \lhd \mathtt{meet}(\pi_1 u, \mathtt{j}) \\ \mathtt{dance}(\pi_1 u) \end{bmatrix} \end{bmatrix} : \mathsf{type} \quad \xrightarrow{\text{type checking}} \quad u : \mathtt{girl}^*, v : \mathtt{meet}(\pi_1 u, \mathtt{j}) \vdash \mathtt{dance}(\pi_1 u) : \mathsf{type}$$

The same line of reasoning shows that a specific indefinite with an ARC projects out of a conditional antecedent, which accounts for the wide-scope reading of (5).

6 Discussion and Conclusion

We proposed an extension of DTS with a new type that implicitly extends the context during type checking. This mechanism not only predicts the projection behavior of ARCs but also captures their interaction with quantifiers, based on the restriction on the permutation rule.

Closely related to this work is [3], which also analyzed CIs with DTS. Informally, the proposal treated a CI as a presupposition that is obligatorily accommodated and does not contribute to the at-issue content. We can point out that the system does not straightforwardly account for the infelicity of an ARC under the scope of non-referential quantifiers, because in such cases it would accommodate the appositive content (as a presupposition) and incorrectly predict that the sentence is felicitous. However, since that study used a different version of DTS, we need an in-depth comparison, which we leave for future work.

We also have not discussed cases where appositives do not project. For instance, appositive content inside an attitude operator is sometimes attributed to the attitude holder rather than the speaker [6], and nominal appositives can take narrow scope with respect to intensional operators [10]. We need to further consider the treatment of extended contexts to account for such challenging phenomena.

[10] Viewing the specificity of an indefinite as a conventional implicature is uncommon but not unnatural because it projects out of entailment-canceling environments (not at-issue) and is generally new to the addressee (not presuppositional). [7] presented a similar argument, analyzing a Persian specificity marker with the system proposed by [9] (note that in his analysis, the CI content is the uniqueness of the nominal to be modified).

Acknowledgements

We thank the two anonymous reviewers for their helpful comments and suggestions, which improved this paper. This work was supported by JST, PRESTO grant number JPMJPR21C8, Japan.

References

[1] S. AnderBois, A. Brasoveanu, and R. Henderson. At-issue proposals and appositive impositions in discourse. *Journal of Semantics*, 32(1):93–138, 2015.

[2] D. Bekki. A proof-theoretic analysis of weak crossover. In K. Yada, Y. Takama, K. Mineshima, and K. Satoh, editors, *New Frontiers in Artificial Intelligence*, pages 228–241, Cham. Springer Nature Switzerland, 2023.

[3] D. Bekki and E. McCready. *Ci via dts.* In *New Frontiers in Artificial Intelligence: JSAI-isAI 2014 Workshops, LENLS, JURISIN, and GABA, Kanagawa, Japan, October 27–28, 2014, Revised Selected Papers.* 2015, pages 23–36.

[4] D. Bekki and K. Mineshima. *Context-Passing and Underspecification in Dependent Type Semantics.* In *Modern Perspectives in Type-Theoretical Semantics.* S. Chatzikyriakidis and Z. Luo, editors. Springer, 2017, pages 11–41.

[5] F. del Gobbo. *Appositives at the interface.* PhD thesis, University of California, 2003.

[6] J. A. Harris and C. Potts. Perspective-shifting with appositives and expressives. *Linguistics and Philosophy*, 32:523–552, 2009.

[7] M. Jasbi. The suffix that makes persian nouns unique. In *Advances in Iranian linguistics*, pages 107–118. John Benjamins, 2020.

[8] R. Nouwen. A note on the projection of appositives. *Formal approaches to semantics and pragmatics: Japanese and beyond*:205–222, 2014.

[9] C. Potts. *The Logic of Conventional Implicatures.* Oxford University Press, 2005.

[10] L. Wang, B. Reese, and E. McCready. The projection problem of nominal appositives. *Snippets*, 10(1):13–14, 2005.