Appositive Projection as Implicit Context Extension in Dependent Type Semantics

Daiki Matsuoka[1][0009−0008−4378−6580](✉), Daisuke Bekki[2][0000−0002−9988−1260], and Hitomi Yanaka[1][0000−0003−0354−6116]

[1] The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
{daiki.matsuoka, hyanaka}@is.s.u-tokyo.ac.jp
[2] Ochanomizu University, 2-1-1 Ohtsuka, Bunkyo-ku, Tokyo 112-8610, Japan
bekki@is.ocha.ac.jp

Abstract. The content of an appositive relative clause is a type of conventional implicature, that is, secondary or supplementary information of a sentence. Focusing on discourse behaviors and scopal properties, we present an analysis of appositive relative clauses based on Dependent Type Semantics, a type-theoretical semantic framework. Our central idea is that appositive content implicitly extends the typing context during the process of type checking, reflecting a property of conventional implicatures, namely, that they directly update the common ground.

Keywords: Dependent Type Semantics · appositive relative clause · conventional implicature · projection · not-at-issue content

1 Introduction

Appositive relative clauses (ARCs) typically convey secondary or supplementary information apart from the *at-issue* content (i.e., the main point of the whole sentence). As such, they are highly independent of their matrix clauses. For example, the ARC in (1) is not subject to negation, considering that (1) implies the appositive content (the professor had met Kim before).

(1) The professor did not recognize Kim, whom she had met before.

This behavior is called *projection*, which is a central topic in formal semantics. [22] argued that projective meanings triggered by specific words or constructions such as ARCs form a distinct meaning class, called *conventional implicatures* (CIs) [11], and proposed a semantic system in which CIs and at-issue meanings are given separate semantic representations.

However, since the proposal by [22], it has been suggested that the content of an ARC can have various types of interactions with that of the matrix clause [1,3, 24, 29]. Above all, there can be anaphoric dependencies between an ARC and its matrix clause, as shown by (2), which is quite difficult to explain if we separate the two clauses completely.

(2) John$_1$, who nearly killed a woman$_2$ with his$_1$ car, visited her$_2$ in the hospital. (adapted from [3])

Nonetheless, the interaction is not entirely free. For instance, pronouns inside an ARC cannot be bound by non-referential quantifiers [8, 19].

(3) #No professor$_1$ recognized Kim, whom they$_1$ had met before.

Therefore, it is not obvious how we can formally account for the extent to which an ARC and its matrix clause can interact with each other.

To address this issue, we present an analysis based on *Dependent Type Semantics* (DTS) [5, 7], a type-theoretical framework of natural language semantics. The basic idea of DTS is to use types as semantic representations (SRs) and to reduce the felicity of a sentence to the well-formedness of its SR, which is checked through a process called *type checking*. This setup provides us with a unified method for capturing two classes of not-at-issue meanings — CIs and presuppositions — as well as at-issue meanings.

The remainder of this paper is structured as follows. First, we describe several properties of ARCs (Section 2) and introduce the details of DTS (Section 3). Then, we propose an extension of DTS (Section 4) and show what predictions it makes (Section 5). After briefly discussing some related work (Section 6), we conclude with future research directions (Section 7).

2 Properties of Appositive Relative Clauses

2.1 Discourse Properties

Not-at-issueness The content of an ARC is generally supplementary; it is not the central point of an utterance. Hence, ARCs contribute to a conversational context in a particular way, as illustrated in (4). In (4b), B's response with *no* targets the content of the matrix clause of (4a), which is at-issue. In contrast, (4c) shows that such a response is not possible in regard to the appositive content.

(4) a. A: Mary, who is a post-doc researcher, gave a talk at the conference.
b. B: No, she didn't. She wasn't able to attend.
c. B: #No, she isn't. She is still in her PhD course.

Thus, appositive content *directly updates* the common ground in a non-negotiable manner [3, 18], which is not the case with at-issue content.

Anti-backgroundedness ARCs basically add new information to a discourse. Thus, if an ARC repeats something that has already been established as part of the common ground, it results in redundancy. More broadly, this property distinguishes CIs from presuppositions, which are supposed to already be in the common ground. As shown in (5), an ARC containing information that has already been introduced leads to infelicity, while such repetition is not problematic with *know*, which presupposes its complement.

(5) a. #Bob has been friends with a linguistics student for a long time, and says that the student, who majors in linguistics, is very diligent.
b. Bob has been friends with a linguistics student for a long time, but doesn't know that she majors in linguistics.

2.2 Scopal Properties

Projection The content of an ARC is projective, meaning that it is not subject to entailment-canceling operators in the matrix clause. Here are examples of negation (6a) and conditional antecedent (6b).

(6) a. It is not the case that Ann, who danced, met John. ⇒ Ann danced.
b. If Ann, who danced, met John, Mary was happy. ⇒ Ann danced.

Quantifier Scope ARCs cannot scopally interact with a quantified noun phrase (NP) unless it is referential. (7) shows that non-referential quantified NPs can neither be modified by an ARC nor bind a pronoun inside an ARC.[3] By contrast, referential quantified NPs can have both types of interactions, as described in (8).

(7) a. #{Every/No} girl, who met John, danced.
b. #{Every/No} girl_1 met John, who praised her_1.

(8) a. {A/Some} girl, who met John, danced.
b. {A/Some} girl_1 met John, who praised her_1.

We remark that the indefinite NPs in (8) must be interpreted as *specific indefinites* [20, 29], which basically have only the widest-scope reading. This point is illustrated by (9), where *a professor* takes wider scope than the conditional.

(9) If a professor, who is famous, meets Ann, she will be surprised. (⇒ there is a famous professor.)

2.3 Anaphoric Dependencies

Anaphoric links can be established between an ARC and its matrix clause in both directions (10). Presupposition resolution, which can also be regarded as an anaphoric process [23], exhibits the same behavior. (11) shows examples of the additive particle *too*.

[3] Note that an ARC can be adjacent to a non-referential quantified NP [4, 9, 19].

(i) Less than half the climbers, who were French nationals, made it to the summit. (adapted from [19])

The ARC here is not semantically in the scope of *less than half the climbers* but rather targets the whole restrictor. Because we presently cannot ascertain what triggers this reading, we leave the analysis for future research.

(10) a. (Appositive → matrix) John, who met a girl$_1$, praised her$_1$.
b. (Matrix → appositive) A girl$_1$ met John, who praised her$_1$. (=(8b))

(11) a. (Appositive → matrix) John, who praised Ann, praised Mary too.
b. (Matrix → appositive) John$_1$ praised Ann, who praised him$_1$ too.

The same pattern can be observed for inter-sentential anaphora (12).

(12) a. (Appositive → matrix) John, who met a girl$_1$, smiled. She$_1$ danced.
b. (Matrix → appositive) A girl$_1$ danced. John, who met her$_1$, smiled.

3 Framework

3.1 Dependent Type Theory

The theoretical foundation of DTS is *dependent type theory* (DTT) [16, 17], a type theory with types that may depend on terms (*dependent types*). We can use DTT as a logical framework by viewing a type and its terms as a proposition and its proofs. For instance, we can regard a product type $A \times B$ as representing the conjunction $A \wedge B$ because its proof is a pair of proofs of A and B.

We can extend this correspondence to predicates using dependent types. For example, the type $\texttt{come}(x)$, representing the proposition "x comes," depends on the variable x. Here, the one-place predicate come has type $\mathsf{e} \to \mathsf{type}$, where e is the type of entities and type is the (higher-order) type of types.

Because types may depend on terms in DTT, we need to handle the well-formedness of types and typing contexts with inference rules. Hence, in addition to usual typing judgments $\Gamma \vdash M : A$ (the term M has type A under Γ), DTT has judgments of the form $\Gamma \vdash A : \mathsf{type}$ (A is a well-formed type under Γ) and $\Gamma\ \mathsf{valid}$ (Γ is a well-formed context). Figure 1 shows the inference rules for well-formed contexts. As we will see in the following, this object-level representation of well-formedness is useful for analyzing the felicity of natural language utterances.

$$\text{(ctx-emp)}\ \frac{}{\langle\rangle\ \mathsf{valid}} \qquad \text{(ctx-ex)}\ \frac{\Gamma\ \mathsf{valid} \qquad \Gamma \vdash A : \mathsf{type}}{\Gamma, x : A\ \mathsf{valid}}\ (x \notin \mathrm{dom}(\Gamma))$$

Fig. 1. Inference rules for well-formed contexts. The rule (ctx-emp) introduces the empty context $\langle\rangle$ (which is omitted when unnecessary). The rule (ctx-ex) forms a well-formed context by extending one with a well-formed type and a fresh variable.

DTT has two special types: the *dependent product type* (Σ*-type*) $(x : A) \times B$ and the *dependent function type* (Π*-type*) $(x : A) \to B$. The former corresponds to the existential quantification $\exists x \in A.B$, and the latter corresponds to the universal quantification $\forall x \in A.B$. Their inference rules are listed in Figure 2.

Central to our approach are the rules (ΣE_1) and (ΣE_2), which take the first and second elements of a pair $\langle M, N\rangle$, respectively. With these rules, we can refer to the content of a Σ-type from outside its scope by "decomposing" pairs.

$$(\Sigma F)\ \frac{\Gamma \vdash A : \mathsf{type} \qquad \Gamma, x : A \vdash B : \mathsf{type}}{\Gamma \vdash (x : A) \times B : \mathsf{type}} \quad (\Sigma I)\ \frac{\Gamma \vdash M : A \qquad \Gamma \vdash N : B[x := M]}{\Gamma \vdash \langle M, N \rangle : (x : A) \times B}$$

$$(\Sigma E_1)\ \frac{\Gamma \vdash M : (x : A) \times B}{\Gamma \vdash \pi_1 M : A} \quad (\Sigma E_2)\ \frac{\Gamma \vdash M : (x : A) \times B}{\Gamma \vdash \pi_2 M : B[x := \pi_1 M]}$$

$$(\Pi F)\ \frac{\Gamma \vdash A : \mathsf{type} \qquad \Gamma, x : A \vdash B : \mathsf{type}}{\Gamma \vdash (x : A) \to B : \mathsf{type}} \quad (\Pi I)\ \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : (x : A) \to B}$$

$$(\Pi E)\ \frac{\Gamma \vdash M : (x : A) \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : B[x := N]}$$

Fig. 2. Inference rules of the Σ-type and the Π-type. In the labels, F, I, and E each stand for *formation*, *introduction*, and *elimination* (e.g., (ΣI) is the introduction rule of the Σ-type).

For example, consider (13a) and its semantic representation (13b). We adopt the notational convention that $(x : A) \times B$ can alternatively be written as $\begin{bmatrix} x : A \\ B \end{bmatrix}$.

(13) a. A girl came.

b. $\begin{bmatrix} u : \begin{bmatrix} x : \mathtt{e} \\ \mathtt{girl}(x) \end{bmatrix} \\ \mathtt{come}(\pi_1 u) \end{bmatrix}$

Here, the set of girls is represented by the type $(x : \mathtt{e}) \times \mathtt{girl}(x)$, which consists of pairs of an entity x and a proof of its being a girl. The part $\mathtt{come}(\pi_1 u)$ means that the girl came, because $\pi_1 u$ is the first element of u, namely, the entity x. In this way, Σ-types show the same effect as if their scope were extended, which is critical for capturing the externally dynamic nature of existential quantification [12].

3.2 Underspecified Type

DTS extends DTT with a type of the form $(x \mathbin{@} A) \times B$, an *underspecified type* (@-type), which represents anaphoric or presuppositional meanings. This type is characterized by the following inference rule:

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

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

To illustrate, consider how the pronoun *she* is resolved in (14a). The condition for the second sentence to be felicitous, which we call its *felicity condition* (FC), is described as in (14b), where the SR of the first sentence is in the typing context, meaning it has already entered the common ground. Here we use the

square bracket notation $[\cdots]$ for the @-type, too. We also abbreviate $(x : \mathsf{e}) \times Px$ as P^* when P is a constant of type $\mathsf{e} \to \mathsf{type}$.

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

b. $v : \underbrace{\begin{bmatrix} u : \mathtt{girl}^* \\ \mathtt{come}(\pi_1 u) \end{bmatrix}}_{\text{A girl came.}} \vdash \underbrace{\begin{bmatrix} w \mathbin{@} \mathtt{female}^* \\ \mathtt{dance}(\pi_1 w) \end{bmatrix}}_{\text{She danced.}} : \mathsf{type}$

To derive (14b), we need to find a term of type $\mathtt{female}^*$, as required by the second premise of the rule ($@F$). By introducing the world knowledge that every girl is female and accordingly assuming a constant $\mathtt{g\text{-}to\text{-}f} : (u : \mathtt{girl}^*) \to \mathtt{female}(\pi_1 u)$, we can construct the term $M_v = \langle \pi_1 \pi_1 v, \mathtt{g\text{-}to\text{-}f}(\pi_1 v) \rangle$ of type $\mathtt{female}^*$. Substituting this term for w, we obtain $\mathtt{dance}(\pi_1 M_v)$, which reduces to $\mathtt{dance}(\pi_1 \pi_1 v)$. This result correctly predicts that *she* can be bound by *a girl*, because $\pi_1 \pi_1 v$ refers to the entity introduced by the first sentence.

3.3 Type Checking and @-elimination

To completely describe the formal system, we need to specify how we can substitute a term for the variable of an @-type during the process of type checking. In a recent version of DTS [5], the type checking function $[\![-]\!]$ returns a set of derivation trees for a typing judgment, with the clause for the @-type being as follows (see [5] for the clauses of other type constructors).[4] Note that the value of $[\![-]\!]$ is not a single derivation because there may be multiple ways to construct a term of type A.

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

The derivations of the three premises of the rule ($@F$) are internally constructed, but $\mathcal{D}_1$ and $\mathcal{D}_2$ are discarded and only $\mathcal{D}_3$ is returned (after normalization). Thus, what we obtain as a result of type checking $(x \mathbin{@} A) \times B$ is the normal form of $B[x := M]$, as if it were the type we wanted to check from the beginning. Hereafter, we refer to this procedure as @-*elimination*.

To see how @-elimination works, consider the FC (14b) again. Figure 3 describes a series of steps to derive it. In the boxes, we write the premises left to prove the FC, whose derivations are constructed one by one (each step is shown with a double arrow). The first step checks the well-formedness of $\mathtt{female}^*$, which is trivial. In the second step, we look for a concrete term of type $\mathtt{female}^*$, and one possible answer is $M_v = \langle \pi_1 \pi_1 v, \mathtt{g\text{-}to\text{-}f}(\pi_1 v) \rangle$, as we have already seen in Section 3.2. The result of this search is propagated to the third premise (p3), which then requires $\mathtt{dance}(\pi_1 M_v)$ to be well-formed. After this final premise is

[4] We define Norm as a function that returns, for any derivation of a typing judgment $\Gamma \vdash M : A$, a derivation where M is reduced to its normal form.

derived (indicated by "(done)"), what is returned as a result is $\mathsf{Norm}(\mathcal{D}_3)$, whose root is the judgment $v : [\cdots] \vdash \mathtt{dance}(\pi_1\pi_1 v) : \mathsf{type}$. We can see that the original @-type has been successfully transformed into the type that corresponds to the expected interpretation of the pronoun *she*. In this way, we can simultaneously perform type checking and eliminate @-types.

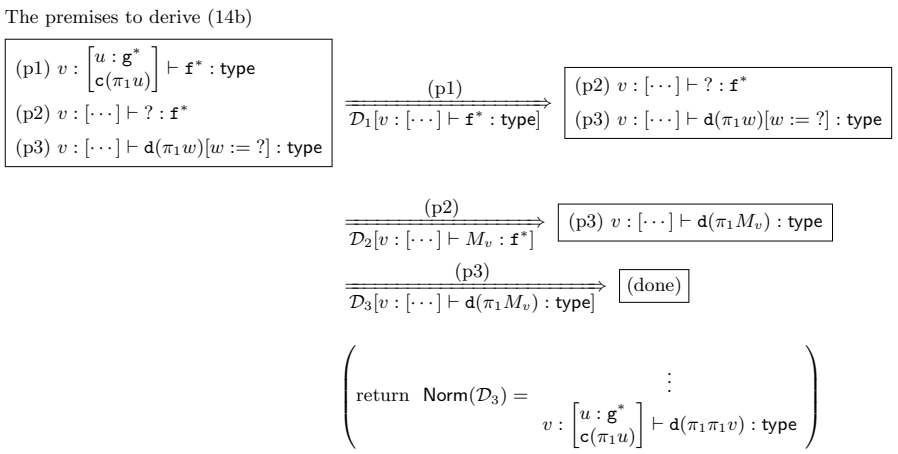

Fig. 3. Process of deriving (14b). We abbreviate the names of the predicates (e.g., $\mathtt{dance} \mapsto \mathtt{d}$). We also write $\mathcal{D}[J]$ for a derivation $\mathcal{D}$ with root J. M_v stands for the term $\langle \pi_1\pi_1 v, \mathtt{g\text{-}to\text{-}f}(\pi_1 v)\rangle$.

3.4 Two-stage Validation

Before presenting our proposal, we must clarify how DTS processes discourses, 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 will be 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.[5] In this paper, we assume two validation stages before an SR A is added to the context Γ.

(i) **Type checking**: by calculating $[\![\Gamma \vdash A : \mathsf{type}]\!]$, we check the well-formedness of A under Γ and eliminate the @-types in A (let $\Gamma \vdash A' : \mathsf{type}$ denote the resultant judgment).
(ii) **Context extension**: we check whether A' is acceptable under Γ. If so, we extend Γ with A' by the rule (ctx-ex).

[5] Although it is beyond the scope of the present paper to characterize such factors, we require at least that accepting the assertion A does not contradict the preceding discourse Γ (i.e., $\Gamma, x : A \nvdash \bot$) and does not lead to redundancy (i.e., there is no M s.t. $\Gamma \vdash M : A$).

4 Proposal

The guiding intuition for our proposal is that CIs directly update the common ground, as we observed in Section 2.1. In light of the above-mentioned two-stage validation, this direct update should be before step (ii) (context extension); otherwise CIs would be directly challengeable (just like at-issue content), which would contradict their not-at-issueness. Thus, we must handle CIs during step (i) (type checking). @-types, which represent anaphoric or presuppositional content, are not appropriate for this purpose because CIs are anti-backgrounded. Therefore, we need a mechanism by which some content is added to, not resolved by, the context during type checking.

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:

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

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

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

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. Conceptually, this context extension during type checking is *implicit* in that it leaves no room for the addressee to choose whether to accept or reject A. Such a response is possible only after step (i), namely, in step (ii) (which is, so to speak, an "explicit" context extension). This property of the $\lhd$-type captures the not-at-issueness and anti-backgroundedness of CIs.

With the $\lhd$-type, computing $[\![\Gamma \vdash A : \mathsf{type}]\!]$ may change the original context Γ as well as the type A. Hence, we revise the two-stage validation as indicated by the underlines below (Δ is empty when A contains no $\lhd$-types). Figure 4 summarizes how we handle different types of content via these two steps.

(i) **Type checking**: by calculating $[\![\Gamma \vdash A : \mathsf{type}]\!]$, we check the well-formedness of A under Γ and eliminate the @-types and $\lhd$-types in A (let $\Gamma, \Delta \vdash A' : \mathsf{type}$ denote the resultant judgment).

(ii) **Context extension**: we check whether A' is acceptable under Γ, Δ. If so, we extend Γ, Δ with A' by the rule (ctx-ex).

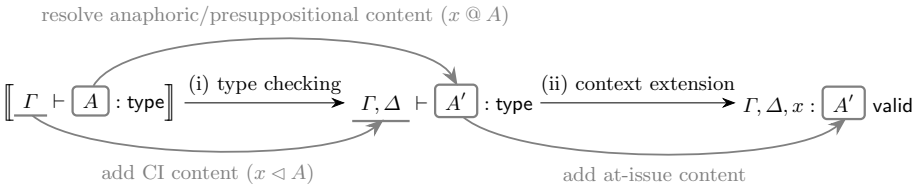

Fig. 4. Overview of how each type of content is processed in DTS

4.2 Permutation

Without modification, the introduction of the $\lhd$-type 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 at the right end of the context.

$$(\Pi F)\ \frac{\begin{array}{cc}\vdots & \vdots \\ \Gamma \vdash A : \mathsf{type} & \Gamma, x : A, y : C \vdash B' : \mathsf{type}\end{array}}{\times}$$

To address this issue, we utilize the following structural rule (*permutation*), which is admissible in DTT.

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

The side condition $x \notin \mathrm{FV}(B)$ is required by the property of DTT 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.

Now we can revise the type checking algorithm for the Π-type as follows (the same applies to the Σ-type).

$$\llbracket \Gamma \vdash (x : A) \to B : \mathsf{type} \rrbracket$$
$$= \left\{ (\Pi F)\ \frac{\begin{array}{cc}\vdots\ \mathcal{D}_1' & \vdots\ \mathcal{D}_2' \\ \Gamma, \Delta, \Theta \vdash A' : \mathsf{type} & \Gamma, \Delta, \Theta, x : A' \vdash B' : \mathsf{type}\end{array}}{\Gamma, \Delta, \Theta \vdash (x : A') \to B' : \mathsf{type}} \;\middle|\; \begin{array}{l}\mathcal{D}_1 \in \llbracket \Gamma \vdash A : \mathsf{type} \rrbracket \\ (\text{Let } \mathcal{D}_1\text{'s root be } \Gamma, \Delta \vdash A' : \mathsf{type}) \\ \mathcal{D}_2 \in \llbracket \Gamma, \Delta, x : A' \vdash B : \mathsf{type} \rrbracket \\ (\text{Let } \mathcal{D}_2\text{'s root be } \Gamma, \Delta, x : A', \Theta \vdash B' : \mathsf{type}) \\ \mathcal{D}_1' = \mathsf{Extend}(\mathcal{D}_1, \mathcal{D}_2, x) \\ \mathcal{D}_2' = \mathsf{Arrange}(\mathcal{D}_2, x)\end{array} \right\}$$

Here, $\mathsf{Extend}(\mathcal{D}_1, \mathcal{D}_2, x)$ simply extends the context of $\mathcal{D}_1$ with the variable declarations in $\mathcal{D}_2$ after x (Θ here). $\mathsf{Arrange}(\mathcal{D}, x)$ 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 $\llbracket - \rrbracket$ is empty.

5 Predictions

In this section, we demonstrate that the $\lhd$-type correctly predicts the behavior of ARCs. To begin with, we explain the theoretical setups for the analysis.

As our syntactic framework, we adopt *Combinatory Categorial Grammar* (CCG) [25], a lexicalized grammar with a transparent syntax-semantics interface. (15) is the lexical entry for the nominative relative pronoun *who*.[6]

$$(15)\quad \underbrace{\mathrm{who}_{\mathrm{nom},+\mathrm{app}}}_{\text{surface form}} := \underbrace{(NP^{\uparrow}\backslash NP)/(S\backslash NP)}_{\text{syntactic category}} : \underbrace{\lambda p.\lambda x.\lambda q.\lambda \vec{y}.\begin{bmatrix} v \lhd px \\ qx\vec{y} \end{bmatrix}}_{\text{semantic representation}}$$

$NP^{\uparrow}$ (the category of generalized quantifiers) stands for $T/(T\backslash NP)$ or $T\backslash(T/NP)$, where T is a variable ranging over categories.[7] The occurrences of T and their corresponding variables ($\vec{x}, \vec{y}, \ldots$) are properly instantiated through a derivation. Figure 5 shows a sample CCG derivation that involves an ARC.

$$\dfrac{\dfrac{\text{Ann}}{NP^{\uparrow}:\ \lambda p\vec{x}.p(\mathtt{a})\vec{x}} \;>\; \dfrac{\dfrac{\text{who}}{(NP^{\uparrow}\backslash NP)/(S\backslash NP):\ \lambda pxq\vec{y}.[\cdots]} \quad \dfrac{\text{danced}}{S\backslash NP:\ \lambda x.\mathtt{dance}(x)}}{NP^{\uparrow}\backslash NP:\ \lambda xq\vec{y}.\begin{bmatrix} v \lhd \mathtt{dance}(x) \\ qx\vec{y} \end{bmatrix}}}{NP^{\uparrow}:\ \lambda q\vec{y}.\begin{bmatrix} v \lhd \mathtt{dance}(\mathtt{a}) \\ q(\mathtt{a})\vec{y} \end{bmatrix}} \;>\; \dfrac{\dfrac{\text{met}}{(S\backslash NP)/NP:\ \lambda yx.\mathtt{meet}(x,y)} \quad \dfrac{\text{John}}{NP^{\uparrow}:\ \lambda p\vec{x}.p(\mathtt{j})\vec{x}}}{S\backslash NP:\ \lambda x.\mathtt{meet}(x,\mathtt{j})}}{S:\ \begin{bmatrix} v \lhd \mathtt{dance}(\mathtt{a}) \\ \mathtt{meet}(\mathtt{a},\mathtt{j}) \end{bmatrix}}$$

Fig. 5. CCG derivation for *Ann, who danced, met John*

Here are some details of the derivation. In the right part, the phrase *met John* is composed by instantiating the category of *John* as $(S\backslash NP)\backslash((S\backslash NP)/NP)$ (i.e., $T = S\backslash NP$). In this case, $\lambda\vec{x}.$ is realized as a single λ-abstraction ($\lambda x.$). In the part where *Ann* and *who danced* are combined, the category of *Ann* is instantiated with $T = NP^{\uparrow}$ (which means $\lambda\vec{x}. = \lambda q.\lambda\vec{y}.$). Finally, the two parts form the whole sentence after the left $NP^{\uparrow}$ is instantiated with $T = S$, in which case $\lambda\vec{y}.$ is realized as a null sequence (i.e., the SR is $\lambda q.[\cdots]$).

Next, we specify how DTS makes predictions. The felicity of an utterance is predicted through the two-stage validation: if both of the two steps succeed, then the utterance is felicitous. We define that DTS predicts inferences in the following way, based on [7] with some modifications.

(16) Let P, H be natural language sentences with P, H as their SRs. Suppose that there exist derivations $\mathcal{D} \in [\![\vdash P : \mathsf{type}]\!]$ with root $\Gamma \vdash P' : \mathsf{type}$ and

[6] We adopt the "result-left" notation here: a phrase of category $A\backslash B$ (resp. A/B) results in an A when combined with a B on its left (resp. right).

[7] Following [25], we stipulate that T must result in S (e.g., $S\backslash NP$, S/NP, $(S\backslash NP)/NP$).

$\mathcal{D}' \in [\![\Gamma, y : P' \vdash H : \mathsf{type}]\!]$ with root $\Gamma' \vdash H' : \mathsf{type}$.
Then, DTS predicts that P implies H if and only if there exists a term M s.t. $\Gamma' \vdash M : H'$ is derivable.

We assume that projection is a type of implication that survives when we embed the premise P into entailment-canceling operators [28].

5.1 Projection

First, we verify the projection behavior of ARCs. (17b) shows the FC of (17a), which is the premise of (6a). Negation $\neg A$ is defined as $(x : A) \to \bot$ in DTS, where $\bot$ is the empty type (i.e., the type representing contradiction).

(17) a. It is not the case that Ann, who danced, met John.

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

Figure 6 shows the process by which the FC is checked. The derivation of the premise (p1), shown in the bottom half of the figure, results in an extended context $u : \mathtt{dance(a)}$, and this change is propagated to the other premise (p2) due to the type checking algorithm of the rule (ΠF). After (p2) is derived, we apply the rule (ΠF) and obtain $u : \mathtt{dance(a)} \vdash (v : \mathtt{meet(a,j)}) \to \bot : \mathsf{type}$.

Next, we show that (17a) implies *Ann danced*, whose SR is $\mathtt{dance(a)}$. Let Γ be the context $u : \mathtt{dance(a)}, s : (v : \mathtt{meet(a,j)}) \to \bot$. We can easily confirm that $\mathtt{dance(a)}$ is well-formed under Γ. Moreover, $\Gamma \vdash u : \mathtt{dance(a)}$ is derivable, which satisfies the condition (16). Therefore, DTS indeed predicts that the appositive content $\mathtt{dance(a)}$ projects out of the negation.

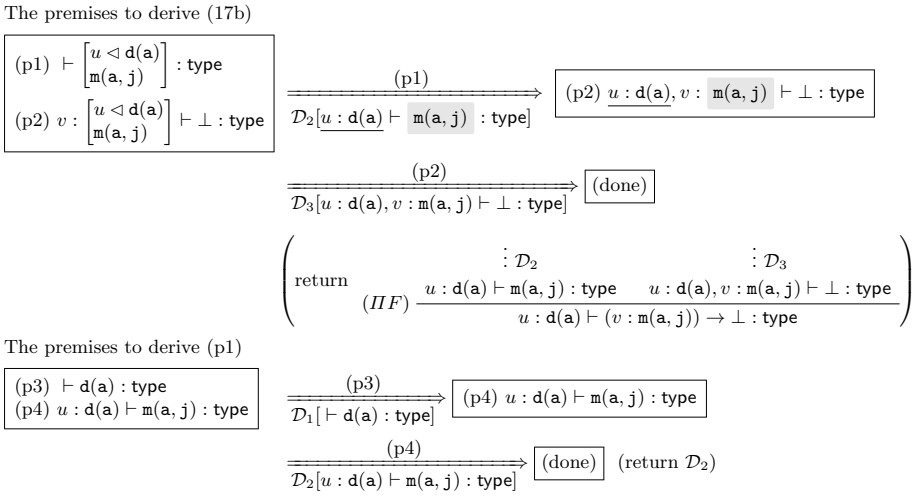

Fig. 6. Process of validating (17b). The underlined parts indicate the context extension, and the gray boxes show the types transformed during type checking.

5.2 Intra-sentential Anaphora

As a next step, we check how our system predicts anaphoric dependencies crossing the boundary between an ARC and its matrix clause. Here, we consider the "appositive → matrix" case (10a) (repeated as (18a)) and leave the analysis of the other direction until Section 5.5.

(18) a. John, who met a girl$_1$, praised her$_1$.

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

Figure 7 shows how the SR of (18a) is composed. We can see that the ARC first combines with the NP *John*, and then with the predicate *praised her*. This derivation results in an SR where the $\lhd$-type for the ARC takes scope over the @-type for the pronoun *her*.

$$\dfrac{\dfrac{\text{John}}{NP^{\uparrow} : \lambda p\vec{x}.p(\texttt{j})\vec{x}} \quad \dfrac{\text{who met a girl}}{NP^{\uparrow}/NP : \lambda xq\vec{y}. \begin{bmatrix} v \lhd \begin{bmatrix} u : \texttt{girl}^* \\ \texttt{meet}(x, \pi_1 u) \end{bmatrix} \\ q x \vec{y} \end{bmatrix}}}{NP^{\uparrow} : \lambda q\vec{y}. \begin{bmatrix} v \lhd [\cdots] \\ q(\texttt{j})\vec{y} \end{bmatrix}} > \quad \dfrac{\text{praised her}}{S\backslash NP : \lambda x. \begin{bmatrix} w \mathbin{@} \texttt{female}^* \\ \texttt{praise}(x, \pi_1 w) \end{bmatrix}}}{S : \begin{bmatrix} v \lhd [\cdots] \\ [\cdots] \end{bmatrix}} >$$

Fig. 7. CCG derivation for (18a)

Then, the FC (18b) is validated as shown in Figure 8. The crucial point is that the @-type is type checked under the context $v : [\cdots]$, which is the appositive content. Hence, we can eliminate the @-type in the same way as in (14b): $w := \langle \pi_1\pi_1 v, \texttt{g-to-f}(\pi_1 v) \rangle$. The resultant SR is $\texttt{praise}(\texttt{j}, \pi_1\pi_1 v)$, which correctly predicts the interpretation that *her* refers back to *a girl*.

$$\boxed{\begin{array}{l} \text{(p1)} \ \vdash \begin{bmatrix} u : \texttt{g}^* \\ \texttt{m}(\texttt{j}, \pi_1 u) \end{bmatrix} : \mathsf{type} \\ \text{(p2)} \ v : \begin{bmatrix} u : \texttt{g}^* \\ \texttt{m}(\texttt{j}, \pi_1 u) \end{bmatrix} \vdash \begin{bmatrix} w \mathbin{@} \texttt{f}^* \\ \texttt{p}(\texttt{j}, \pi_1 w) \end{bmatrix} : \mathsf{type} \end{array}}$$

$$\xrightarrow[\mathcal{D}_1\,[\vdash [\cdots] : \mathsf{type}]]{\text{(p1)}} \boxed{\text{(p2)} \ v : \begin{bmatrix} u : \texttt{g}^* \\ \texttt{m}(\texttt{j}, \pi_1 u) \end{bmatrix} \vdash \begin{bmatrix} w \mathbin{@} \texttt{f}^* \\ \texttt{p}(\texttt{j}, \pi_1 w) \end{bmatrix} : \mathsf{type}}$$

$$\xrightarrow[\mathcal{D}_2[v : [\cdots] \vdash \texttt{p}(\texttt{j}, \pi_1\pi_1 v) \ : \mathsf{type}]]{\text{(p2)}} \boxed{\text{(done)}} \ \text{(return } \mathcal{D}_2\text{)}$$

Fig. 8. Process of validating (18b)

5.3 Inter-sentential Anaphora

Next, we consider anaphora between sentences. As for the "appositive → matrix" direction (12a) (here repeated as (19)), the FC of the first sentence is (20a). After it is validated, the context is (implicitly) extended with $v : [\cdots]$. After the at-issue content $\mathtt{smile}(\mathtt{j})$ is accepted, we obtain an updated context (20b).

(19) John, who met a girl_1, smiled. She_1 danced.

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

b. $v : \begin{bmatrix} u : \mathtt{girl}^* \\ \mathtt{meet}(\mathtt{j}, \pi_1 u) \end{bmatrix}, v' : \mathtt{smile}(\mathtt{j})$

Note that the appositive and at-issue content are not distinguished in the typing context: the context (20b) is identical to what we would obtain if the first sentence were replaced with "*John met a girl. He smiled.*" Since the appositive content is available in the context, we can resolve the pronoun *she* in the second sentence in exactly the same way as the previous example.

The "matrix → appositive" direction (12b), which is repeated as (21), is more complicated. Supposing that the first sentence is successfully type checked and accepted, the FC of the second sentence is (22).

(21) A girl_1 danced. John, who met her_1, smiled.

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

Figure 9 shows the derivation process. In deriving (p1), we can eliminate the @-type with v,[8] and the result is propagated to the other sub-goal (p2), as indicated by the parts highlighted in gray. Finally, we obtain a context extended with $z : \mathtt{meet}(\mathtt{j}, \pi_1\pi_1 v)$ (*John met her*), as expected.

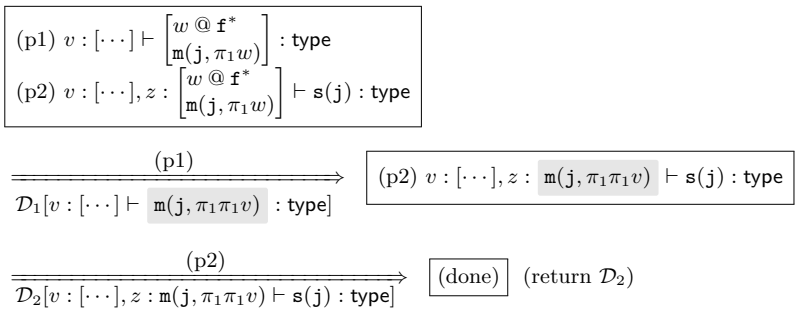

Fig. 9. Process of validating (21)

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

5.4 ARC + Non-referential Quantifier

Let us turn to the interaction with quantifiers. We first observe (23a), where an ARC modifies *every girl*.[9] Its FC is shown in (23b).

(23) a. #Every girl, who met John, danced.

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

Figure 10 describes how its infelicity is predicted. When the premise (p2) 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 the type checking to fail. With the failure of type checking, DTS predicts the infelicity of (23a).

$$\text{(p1)}\ \vdash \mathtt{g}^* : \mathsf{type}$$
$$\text{(p2)}\ u : \mathtt{g}^* \vdash \begin{bmatrix} v \lhd \mathtt{m}(\pi_1 u, \mathtt{j}) \\ \mathtt{d}(\pi_1 u) \end{bmatrix} : \mathsf{type}$$

$$(\cdots) \xRightarrow{\text{(p2)}} \mathcal{D}_2[u : \mathtt{g}^*, \underline{v : \mathtt{m}(\pi_1 u, \mathtt{j})} \vdash \boxed{\mathtt{d}(\pi_1 u)} : \mathsf{type}] \qquad \text{(perm)}\ \dfrac{\vdots\ \mathcal{D}_2 \quad u : \mathtt{g}^*, v : \mathtt{m}(\pi_1 u, \mathtt{j}) \vdash \mathtt{d}(\pi_1 u) : \mathsf{type}}{\times}$$

Fig. 10. Process showing the infelicity of (23a)

We can likewise handle the case of binding. Again using *every girl* as our example, we describe the FC of (24a) in (24b). We first eliminate the @-type with $u : \mathtt{girl}^*$ to obtain the reading where *her* is bound by *every girl*, which yields $(v \lhd \mathtt{praise}(\mathtt{j}, \pi_1 u)) \times \cdots$. Because u occurs free in the appositive content $\mathtt{praise}(\mathtt{j}, \pi_1 u)$, the type checking fails as in Figure 10.[10]

(24) a. #Every girl$_1$ met John, who praised her$_1$.

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

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 DTT.

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

[10] Note that the type checking of (24b) succeeds if w is otherwise resolved, which corresponds to cases where the pronoun *her* is not bound by *every girl* (e.g., where it refers to a female person previously introduced in the discourse).

5.5 ARC + Referential Quantifier

Before checking the case of referential quantifiers, we introduce an auxiliary assumption that a specific indefinite NP is represented by a $\lhd$-type. This is motivated by the fact that a specific indefinite projects out of entailment-canceling environments (like definites) and is generally new to the addressee [10]: both properties can be straightforwardly captured by the $\lhd$-type.[11] Concretely, we assume the following alternative lexical entry for the indefinite determiner.

$$(25)\quad \mathrm{a}_{\mathrm{spec}} := NP^{\uparrow}/N : \lambda n.\lambda p.\lambda \vec{x}.\begin{bmatrix} u \lhd (x : \mathtt{e}) \times nx \\ p(\pi_1 u)\vec{x} \end{bmatrix}$$

Then, the sentence (26a) can be translated into (26b). Because $u : \mathtt{girl}^*$, on which the appositive content $\mathtt{meet}(\pi_1 u, \mathtt{j})$ depends, is also implicitly added to the context, we need not apply the rule (perm) and thus type checking succeeds, resulting in the judgment (26c).

(26) a. A girl, who met John, danced.

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

c. $u : \mathtt{girl}^*, v : \mathtt{meet}(\pi_1 u, \mathtt{j}) \vdash \mathtt{dance}(\pi_1 u) : \mathtt{type}$

The same line of reasoning shows the felicity of the binding case (8b), which also accounts for the intra-sentential "matrix $\rightarrow$ appositive" anaphora (10b). Likewise, our system can correctly predict that a specific indefinite with an ARC projects out of conditional antecedents, which we observed in (9).

It is noteworthy that if the indefinite is interpreted as non-specific in (26b), then the SR is $(u : \mathtt{girl}^*) \times \cdots$, which cannot be successfully type checked for the same reason as (23b). This result accounts for why an indefinite NP with an ARC does not have a non-specific reading.

5.6 Additional Analysis: Clause-final ARCs

ARCs show peculiar properties when situated at the clause-final position. First, [2] observed and [26] experimentally confirmed that clause-final ARCs can be a target of the hearer's direct response, which suggests that they can be at-issue.

(27) a. Liz might be with her husband, who has prostate cancer.

b. No, he has lung cancer.

Note that (27a) implies that Liz's husband has prostate cancer, meaning that the appositive content is not subject to the modal *might*.

Second, [21, 24] showed that an ARC at the end of a subordinate clause can take narrower scope than the main clause. For example, the ARC in (28a) does not project out of the scope of *if* but yields an interpretation similar to (28b).

[11] [14] presented a similar argument, analyzing a Persian specificity marker with the system for CIs proposed by [22] (note that in the analysis by [14], the CI content is the uniqueness of the nominal to be modified).

(28) (Adapted from [24])
[Context: someone in the department made a big mistake.]

a. If tomorrow I called the Chair, who in turn called the Dean, then we would be in big trouble.

b. If tomorrow I called the Chair and the Chair in turn called the Dean, then we would be in big trouble.

To capture these facts, we introduce an additional lexical entry (29) for relative pronouns appearing in clause-final ARCs.

(29) $\text{who}_{\text{cls-fin}} := ((S\backslash(S/NP))\backslash NP)/(S\backslash NP) : \lambda p.\lambda x.\lambda q. \begin{bmatrix} u : qx \\ px \end{bmatrix}$

The SR simply conjoins the content of the matrix clause and that of the ARC with a Σ-type. The syntactic category indicates that it combines first with a relative clause predicate ($S\backslash NP$), then with an antecedent (NP), and finally with a clause missing the object (S/NP), thus forcing the ARC to be clause-final.

We can predict the projection behavior by utilizing the flexible notion of constituency in CCG. As shown in Figure 11, the clause missing the object (*John might meet*) forms a constituent of S/NP due to the functional composition rule (>**B**). We can thus keep the appositive content $\texttt{dance}(\texttt{a})$ outside the scope of the possibility operator $\Diamond$.[12] We can explain the narrow scope reading of (28a) in the same way because *I called* has category S/NP.[13]

$$\dfrac{\dfrac{\dfrac{\dfrac{\text{John}}{NP^{\uparrow} : \lambda p\vec{x}.p(\texttt{j})\vec{x}} \quad \dfrac{\text{might}}{(S\backslash NP)/(S\backslash NP) : \lambda px.\Diamond px}}{S/(S\backslash NP) : \lambda p.\Diamond p(\texttt{j})}\,{>}\mathbf{B} \quad \dfrac{\text{meet}}{(S\backslash NP)/NP : \lambda yx.\texttt{meet}(x,y)}}{S/NP : \lambda y.\Diamond\texttt{meet}(\texttt{j},y)}\,{>}\mathbf{B} \quad \dfrac{\text{Ann, who danced}}{S\backslash(S/NP) : \lambda q.\begin{bmatrix} u : q(\texttt{a}) \\ \texttt{dance}(\texttt{a}) \end{bmatrix}}}{S : \begin{bmatrix} u : \Diamond\texttt{meet}(\texttt{j},\texttt{a}) \\ \texttt{dance}(\texttt{a}) \end{bmatrix}}\,{<}$$

Fig. 11. CCG derivation for *John might meet Ann, who danced*. We have omitted the analysis of *Ann, who danced* for brevity.

[12] We leave open how to implement modal operators in DTS (see, e.g., [27] for an analysis).

[13] One crucial limitation of this analysis is that it does not correctly handle scope interactions with quantifiers. [19] observed that the following has a reading that only the interviewed climbers were French.

(i) They interviewed less than half the climbers, who were all French nationals.

Our present analysis would predict that the ARC is in the nuclear scope of the quantifier. We need a mechanism to pass the intersection of the restrictor and the nuclear scope to the ARC, which will be the subject of future work.

6 Related Work

6.1 DTS-based Approach

[6] presented an analysis of CIs with DTS, which is closely related to the present work. Their proposal was the *CI operator* $\mathsf{CI}(@ :: A)$, where the term $@ :: A$ (an underspecified term [7]) launches a proof search in the same way as an @-type. Here, a specific mechanism of their system guarantees that $@ :: A$ is never resolved by the context. In other words, the CI content A is formalized as an obligatorily accommodated presupposition. In this way, the operator correctly predicts both the projection behavior and the anti-backgroundedness of CIs.

A possible concern with this analysis is that it might not be compatible with other analyses of presuppositions with DTS. For instance, [5] recently proposed the following fallback procedure upon the failure of type checking, which is meant to reflect the distinction between *global* and *local* accommodation.

(30) If no term M with $x_1 : A_1 \dots x_n : A_n \vdash M : A$ is found in type checking $(x \mathbin{@} A) \times B$, the type checker can do either of the following:
   a. Global accommodation: add a constant symbol of type $(x_1 : A_1) \to \dots (x_n : A_n) \to M$ to the signature and re-run the type checking.
   b. Local accommodation: replace the @-type with $(x : A) \times B$ and continue the type checking.

If we adopt this definition, the CI operator incorrectly predicts that ARCs can take scope under non-referential quantifiers. (31) is a schematized SR representing cases where an ARC is bound by *every* NP.

(31) $(u : A) \to (\dots \mathsf{CI}(@ :: B(u)) \dots)$

Whether the type checker performs global or local accommodation of $B(u)$, the type checking does not fail, and this predicts that (31) is well-formed. The result is the same if we replace @-terms with @-types. Hence, this justifies introducing the CI type, a new operator independent of accommodation, in addition to the existing @-term/type.

6.2 Dynamic Approach

[3] and several related studies analyzed ARCs with dynamic semantics, based on the same idea as ours, namely, that appositive content directly updates the common ground. Their approach utilizes discourse referents (drefs) for propositions. Appositive content constrains the dref p^{cs} representing the common ground, while at-issue content targets another dref p, which later updates p^{cs}.

Although they covered various phenomena related to ARCs, it remains unclear how to account for the fact that they do not scopally interact with non-referential NPs. One possible solution is to integrate the proposal by [19] for nominal appositives, which also employs a dynamic approach. However, [3] and [19] substantially differ in the setup of dynamic semantics. For instance, [3] treats a

discourse referent as a partial function from possible worlds to entities, while [19] regards one as ambiguous between singular and plural entities. We believe further investigation is necessary to reconcile these differences.

6.3 Orphan Approach

Following [22] and many others, we posited that an ARC is locally attached to its antecedent in its surface position, with the independence of the appositive content handled at the semantic level. In contrast, some studies assume a different syntactic structure, where an ARC is viewed as an "orphan," that is, a clause that is not integrated with its antecedent but shifted to a structurally higher position.

For instance, [8] proposed that an ARC is an independent matrix clause, with the relative pronoun interpreted as an E-type pronoun. More recently, [24] extended this idea and suggested that an ARC can be attached to any propositional node dominating the antecedent.

A question that arises with this approach is why the relative pronoun of an ARC must be coindexed with the antecedent, even though it is an E-type pronoun. Consider the contrast between (32a) and (32b). The pronoun *her* allows inter-sentential anaphora, whereas the relative pronoun *whom* does not. If the relative pronoun is E-type, then (32b) is predicted to be felicitous.

(32) a. Ann_1 has been annoyed recently. The culprit is her nasty colleague_2. She_2 always teases her_1.

b. Ann_1 has been annoyed recently. #The culprit is her nasty colleague_2, whom_1 she_2 always teases.

Thus, a formal procedure needs to be implemented to ensure appropriate coindexation of relative pronouns and NPs.

7 Conclusion

We proposed an extension of DTS with the CI type $(x \lhd A) \times B$, which implicitly extends the typing context during type checking. This mechanism reflects the idea that CIs directly update the common ground. Our proposal can predict the projection behavior of appositive content and the anaphoric dependencies between ARCs and their matrix clauses. It also captures the (non-)interactions between ARCs and quantified NPs, based on the restriction on the permutation rule.

In future work, it would be interesting to consider *perspective shift* with CIs [13]. Although CIs generally express commitments made by the speaker, they can also be attributed to other attitude holders. For instance, the appositive content in (33) seems to describe the belief of the speaker's aunt.

(33) My aunt is extremely skeptical of doctors in general. She says that dentists, who are only in it for the money anyway, are not to be trusted at all. (adapted from [13])

[15] analyzed this phenomenon as a type of discourse anaphora: the contextual information determines the perspective under which a CI is interpreted. Following this idea, we could revise the lexical entry of the relative pronoun of an ARC so that it includes an @-type for an appropriate perspective. However, because the treatment of epistemic states with DTS is not well established, we leave this analysis to future work.

Another possible direction of future research is to investigate how to relate our analysis of CIs to presupposition accommodation, returning to the spirit of [6]. Although we pointed out in Section 6 that the recent definition of accommodation in DTS is not compatible with CIs, we might be able to revise it so that CIs and accommodated presuppositions are handled in a more unified manner. We need to further consider the discourse behavior of the two types of content to tackle this challenge.

Acknowledgements

We would like to thank the anonymous reviewers of LENLS20 for their comments and suggestions, which helped us improve this paper. This work was supported by JST, PRESTO grant number JPMJPR21C8, Japan, and JSPS KAKENHI Grant Number JP23H03452, Japan.

References

1. Amaral, P., Roberts, C., Smith, E.A.: Review of the logic of conventional implicatures by chris potts. Linguistics and Philosophy **30**(6), 707–749 (2007). https://doi.org/10.1007/s10988-008-9025-2
2. AnderBois, S., Brasoveanu, A., Henderson, R.: Crossing the appositive/at-issue meaning boundary. Semantics and Linguistic Theory **20**, 328–346 (2010). https://doi.org/10.3765/salt.v20i0.2551
3. Anderbois, S., Brasoveanu, A., Henderson, R.: At-issue Proposals and Appositive Impositions in Discourse. Journal of Semantics **32**(1), 93–138 (2013). https://doi.org/10.1093/jos/fft014
4. Arnold, D.: Non-restrictive relatives are not orphans. Journal of Linguistics **43**(2), 271–309 (2007). https://doi.org/10.1017/S0022226707004586
5. Bekki, D.: A proof-theoretic analysis of weak crossover. In: Yada, K., Takama, Y., Mineshima, K., Satoh, K. (eds.) New Frontiers in Artificial Intelligence: JSAI-isAI 2021 Workshops, JURISIN, LENLS18, SCIDOCA, Kansei-AI, AI-BIZ, Yokohama, Japan, November 13–15, 2021, Revised Selected Papers. pp. 228–241. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-36190-6_16
6. Bekki, D., McCready, E.: CI via DTS. In: Murata, T., Mineshima, K., Bekki, D. (eds.) New Frontiers in Artificial Intelligence: JSAI-isAI 2014 Workshops, LENLS, JURISIN, and GABA, Kanagawa, Japan, October 27–28, 2014, Revised Selected Papers. pp. 23–36. Springer, Berlin, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48119-6_3
7. Bekki, D., Mineshima, K.: Context-passing and underspecification in dependent type semantics. In: Chatzikyriakidis, S., Luo, Z. (eds.) Modern Perspectives in Type-Theoretical Semantics, pp. 11–41. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-50422-3_2

8. del Gobbo, F.: Appositives at the interface. Ph.D. thesis, University of California (2003)
9. del Gobbo, F.: On the syntax and semantics of appositive relative clauses. Parentheticals **106**, 173–201 (2007). https://doi.org/10.1075/la.106.10del
10. Geurts, B.: Specific indefinites, presupposition and scope. In: Bauerle, R., Reyle, U., Zimmermann, T. (eds.) Presuppositions and Discourse: Essays Offered to Hans Kamp, pp. 125–158. Brill, Leiden, The Netherlands (2010). https://doi.org/10.1163/9789004253162_006
11. Grice, H.P.: Logic and conversation. In: Cole, P., Morgan, J.L. (eds.) Speech Acts, Syntax and Semantics, vol. 3, pp. 41–58. Academic Press (1975). https://doi.org/https://doi.org/10.1163/9789004368811_003
12. Groenendijk, J., Stokhof, M.: Dynamic predicate logic. Linguistics and Philosophy **14**(1), 39–100 (1991). https://doi.org/10.1007/BF00628304
13. Harris, J.A., Potts, C.: Perspective-shifting with appositives and expressives. Linguistics and Philosophy **32**(6), 523–552 (2009). https://doi.org/10.1007/s10988-010-9070-5
14. Jasbi, M.: The suffix that makes persian nouns unique. In: Advances in Iranian linguistics. p. 107–118. John Benjamins (2020). https://doi.org/10.1075/cilt.351.06jas
15. Koev, T.: Two puzzles about appositives: Projection and perspective shift. In: Etxeberria, U., Fălăuş, A., Irurtzun, A., Leferman, B. (eds.) Proceedings of Sinn und Bedeutung. vol. 18, pp. 217–234 (2014)
16. Luo, Z.: Computation and Reasoning: A Type Theory for Computer Science, International Series of Monographs on Computer Science, vol. 49. Oxford University Press (1994)
17. Martin-Löf, P.: Intuitionistic type theory. Bibliopolis, Naples (1984), notes by Giovanni Sambin of a series of lectures given in Padua, June 1980
18. Murray, S.E.: Varieties of update. Semantics and Pragmatics **7**(2), 1–53 (2014). https://doi.org/10.3765/sp.7.2
19. Nouwen, R.: On appositives and dynamic binding. Research on language and computation **5**(1), 87–102 (2007). https://doi.org/10.1007/s11168-006-9019-6
20. Nouwen, R.: A note on the projection of appositives. In: McCready, E., Yabushita, K., Yoshimoto, K. (eds.) Formal Approaches to Semantics and Pragmatics: Japanese and Beyond, pp. 205–222. Springer Netherlands, Dordrecht (2014). https://doi.org/10.1007/978-94-017-8813-7_10
21. Poschmann, C.: Embedding non-restrictive relative clauses. Proceedings of Sinn und Bedeutung **22**(2), 235–252 (2018)
22. Potts, C.: The Logic of Conventional Implicatures. Oxford University Press (2004). https://doi.org/10.1093/acprof:oso/9780199273829.001.0001
23. van der Sandt, R.: Presupposition Projection as Anaphora Resolution. Journal of Semantics **9**(4), 333–377 (1992). https://doi.org/10.1093/jos/9.4.333
24. Schlenker, P.: Supplements without Bidimensionalism. Linguistic Inquiry **54**(2), 251–297 (2023). https://doi.org/10.1162/ling_a_00442
25. Steedman, M.: The Syntactic Process. MIT press (2000)
26. Syrett, K., Koev, T.: Experimental Evidence for the Truth Conditional Contribution and Shifting Information Status of Appositives. Journal of Semantics **32**(3), 525–577 (2014). https://doi.org/10.1093/jos/ffu007
27. Tanaka, R., Mineshima, K., Bekki, D.: Resolving Modal Anaphora in Dependent Type Semantics. In: Murata, T., Mineshima, K., Bekki, D. (eds.) New Frontiers in Artificial Intelligence: JSAI-isAI 2014 Workshops, LENLS, JURISIN, and GABA, Kanagawa, Japan, October 27–28, 2014, Revised Selected Papers. p.

83–98. Springer-Verlag, Berlin, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48119-6_7
28. Tonhauser, J., Beaver, D., Roberts, C., Simons, M.: Toward a taxonomy of projective content. Language **89**(1), 66–109 (2013). https://doi.org/10.1353/lan.2013.0001
29. Wang, L., Reese, B., McCready, E.: The projection problem of nominal appositives. Snippets **10**(1), 13–14 (2005)