Context updates in head-final languages: Linear order or hierarchy?[1]

WooJin CHUNG — *New York University*

Abstract. This paper argues that extant approaches to presupposition projection that either rely on strict linear order (Schlenker, 2009) or hierarchy (Romoli and Mandelkern, 2017) cannot provide a uniform account of data drawn from head-final languages. While building on Schlenker's theory, this paper resolves the issues by restricting the calculation of local contexts to specific points in the parsing process. The consequence is that the theory makes a prediction robust to the head directionality parameter.

Keywords: presupposition projection, local context, parsing, linear order, hierarchy.

1. Introduction

Schlenker (2009, 2010, 2011a, 2011b) proposes a parsing-based account of presupposition projection that derives the local context of an expression on the basis of classical truth-conditional semantics. Schlenker argues that there is a pragmatic requirement that a presupposition must be entailed by a local context calculated according to the following definition:

(1) Local context (Schlenker 2011, incremental version)
The local context of an expression d of propositional or predicative type which occurs in a syntactic environment $a \_ b$ in a context C is the strongest proposition or property x which guarantees that for any expression d' of the same type as d, for all strings b' for which $a\ d'\ b'$ is a well-formed sentence,

$$C \vDash^{c' \to x} a\ (c' \text{ and } d')\ b' \leftrightarrow a\ d'\ b'$$

The key aspect of Schlenker's theory is that local context is calculated *incrementally*: the interpreter traverses a string of expressions from left to right. Upon encountering E, it only has access to the expressions that *linearly precede* E. Given those expressions, the interpreter calculates the strongest but innocuous restriction. This left-to-right bias is built into the formulation of local context. In (1), the interpreter is completely agnostic to what follows the expression the local context of which is to be calculated (b' in this case). Thus, it needs to take into account every possible continuation of the sequence $a\ d'$ that results in a well-formed sentence.

Schlenker claims that his theory of local contexts achieves explanatory adequacy in the sense that it predicts how presuppositions project based on syntax and classical truth-conditional semantics. On the other hand, extant dynamic approaches (Stalnaker, 1974; Heim, 1983) fail to do so because they encode such behavior in the lexical specification of words. For instance, Heim specifies 'Context Change Potentials' in the semantics of operators so that they can update the context in a specific order. However, as Schlenker points out, such a system would be

[1]I would like to thank Philippe Schlenker for guiding me through the beautiful world of presuppositions. I would also like to thank Chris Barker, Jacopo Romoli, Masha Esipova, and Robert Pasternak for the discussions we had during the development of this work. I thank the anonymous SuB 22 reviewers and the audiences for their extremely helpful comments. All remaining errors are my own.

Proceedings of Sinn und Bedeutung 22, vol. 1, ZASPiL 60, pp. 313–329. ZAS, Berlin.

too strong to be sufficiently explanatory because one can encode an arbitrary update behavior to any given operator. For instance, one can come up with a deviant conjunction *and** which updates the context in the opposite order of ordinary conjunction *and*. The dynamic approaches in principle cannot rule out this possibility.

While maintaining Schlenker's view that presupposition projection behavior is closely related to the left-to-right bias inherent in parsing, this paper points out that local contexts cannot be calculated in a strictly incremental fashion. Evidence comes from head-final languages where predicates typically follow their arguments. An alternative parsing-based solution is to apply the algorithm to syntactic trees (Romoli and Mandelkern, 2017). It will be shown that the hierarchy-based account has difficulties explaining the presupposition projection behavior of coordinated structures.

I suggest that Schlenker's algorithm should not be run word-by-word, but rather *domain-by-domain*, possibly *postponing* the computation of local context. The proposed analysis resolves the problems encountered in Schlenker's original algorithm and the hierarchy-based account, while reproducing the correct predictions.

2. Issues in the linear order-based approach

2.1. Attitude context

Let's first take a look at an English example in which a presupposition trigger is embedded under an attitude verb, and see how the incremental version of Schlenker's algorithm makes the right prediction. In (2), the attitude verb *believes* embeds the presupposition trigger *continues*.

(2) John believes that Mary smoked in high school, and he believes that she continues to smoke.

According to the incremental version of Schlenker's algorithm, the target expression and whatever follows it cannot be foreseen. It must be the *strongest* yet *innocuous* restriction that can be made regardless of what comes after the embedded clause. The point at which such calculation takes place in (2) is marked with • in (3a).

(3) a. he **believes** ~~that she **continues** to smoke~~ •
b. Corresponding equivalence:
For any expression d' of a propositional type,

$$C \vDash^{c' \to x} \text{he believes (c' and d')} \leftrightarrow \text{he believes d'}$$

Note that the matrix verb *believe* has already been encountered at the point of local context calculation and the interpreter already has it on its workspace. Thus, the context set is restricted to John's doxastic worlds, and the algorithm correctly predicts that the presupposition of (2) is 'John believes that Mary smoked'. What is crucial in this account is that the attitude verb *precedes* the embedded clause. Despite the success in accounting for the English data, we are led to question what the theory would predict for a language where an attitude verb *follows* the

embedded clause. Korean is such a language, and the default word order is SOV. An example is provided in (4).

(4) John-un [Mary-ka (cikum-to) keysokhayse tambay-lul pi-n-tako]
John-TOP [Mary-NOM (now-also) continuously cigarette-ACC smoke-PRES-COMP]
mit-nun-ta.
believe-PRES-DECL
'John believes that Mary continues to smoke.' (embedded clause > believe)

In the above example, the incremental version of Schlenker's algorithm cannot restrict the local context of the embedded clause to John's doxastic worlds. How the local context is computed is provided in (5). Here, the only information available to the interpreter is *John*. Unless the only possible sentence completion (*b'* in the posited equivalence) is *mit* 'believe', the interpreter would fail to restrict the local context to John's doxastic worlds. However, there are numerous ways to complete the sentence. One possible completion is *malha(y)* 'say' as in (6). So the algorithm predicts that the local context includes the set of worlds that are not John's doxastic worlds and the example does not presuppose that John believes that Mary used to smoke.

(5) a. John-TOP ~~[Mary-NOM **continuously** smoke-PRES-COMP]~~ • **believe**
b. Corresponding equivalence:
For any expression d' of a propositional type, and for all strings *b'* for which *John d' b'* is a well-formed sentence,

$$C \vDash^{c' \to x} \text{John (c' and d') b'} \leftrightarrow \text{John d' b'}$$

(6) John-un [Mary-ka (cikum-to) keysokhayse tambay-lul pi-n-tako]
John-TOP [Mary-NOM (now-also) continuously cigarette-ACC smoke-PRES-COMP]
malhay-ss-ta.
say-PAST-DECL
'John says that Mary continues to smoke.'

Contrary to the prediction, the example does presuppose that John believes that Mary used to smoke. We would want the local context of the embedded clause to be restricted just as much as in the English example. The issue arises because the algorithm strictly relies on linear order.

2.2. Scrambling

The naive version of Schlenker's algorithm cannot account for the Korean scrambling example in (7). The entire embedded clause linearly precedes the matrix clause, so the interpreter does not have access to the matrix subject and the attitude verb.[2] In fact, as shown in (8), the interpreter does not have access to any information at all. It is predicted that the local context

[2]It is possible that Schlenker's original algorithm can be improved by letting the embedded clause *reconstruct* before calculating its local context. However, it requires *delaying* the computation of the local context until the complete syntax structure is constructed and reconstruction takes place. As a result, it would weaken the theory's main argument that local contexts are calculated in a strictly incremental fashion.

of the embedded clause is the global context and the sentence presupposes that Mary used to smoke in the actual world.

(7) [Mary-ka keysokhayse tambay-lul pi-n-tako] John-un *t*
Mary-NOM continuously cigarette-ACC smoke-PRES-COMP John-TOP *t*
mit-nun-ta.
believe-PRES-DECL
(Lit.) 'That Mary continues to smoke, John believes.'

(8) ~~[Mary-NOM continuously cigarette-ACC smoke-PRES-COMP]~~ • John-TOP
t believe-PRES-DECL

Contrary to the prediction, the sentence presupposes that Mary used to smoke in John's beliefs.

2.3. Relative clause

Ingason (2016) raises another issue based on Japanese relative clause constructions. The Japanese examples in (9) show that the context is first updated with respect to a head noun, then with respect to its relative clause.

(9) a. Taro-ga [[**yamome**-dearu] **zyosei**-ni] atta.
Taro-NOM [[**widow**-COP] **woman**-DAT] met
'Taro met a woman who is a widow. (widow > woman)

b. # Taro-ga [[**zyosei**-dearu] **yamome**-ni] atta.
Taro-NOM [[**woman**-COP] **widow**-DAT] met
'Taro met a widow who is a woman. (woman > widow)

Example (9a) is felicitous because *zyosei* 'woman' updates the context first, then *yamome-dearu* 'who is a widow'. Since 'widow' entails 'woman', updating 'widow' after 'woman' is felicitous. In contrast, (9b) triggers the redundancy effect because the head noun *yamome* 'widow' is more restrictive than the relative clause *zyosei-dearu* 'who is a woman'. Ingason suggests that this is evidence that the order of context update mirrors syntactic hierarchy, but not linear order.

3. The hierarchy-based account

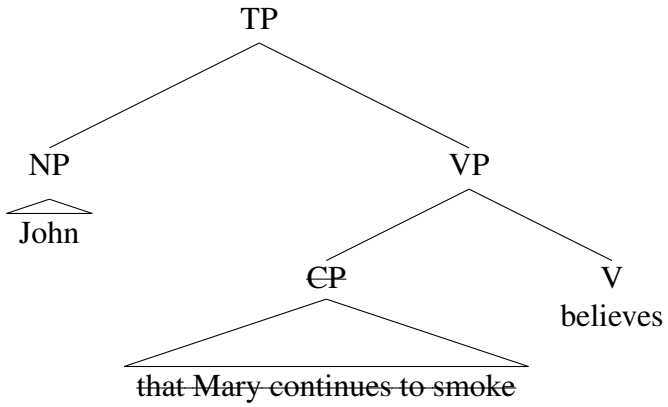
Romoli and Mandelkern (2017) reform Schlenker's algorithm in a way that the local contexts are calculated on LF: when calculating the local context of *E* within a full clause *S*, the interpreter considers *only the expressions that c-command S at LF*, instead of considering the expressions that linearly precede it. Formally, the hierarchy-based version of local context is defined as follows:

(10) Good-completion (Romoli and Mandelkern, 2017)
A *good-completion* of L at α is any well-formed LF which is identical to L except that any clause dominated or asymmetrically c-commanded by α may be replaced by new material. For any sub-tree Y, a *Y-good-completion* of L at α is any good completion of L at α such that α is replaced by a subtree beginning with $[\ Y\ [$ *and* __.

(11) Hierarchical Transparent Local Contexts (Romoli and Mandelkern, 2017)
The local context of expression E in LF L and global context C is the strongest $[\![Y]\!]$ s.t., where α is the lowest node which dominates a full clause containing E, for all good-completions D of L at α, and for all Y-good-completions D^Y of L at α, $[\![D]\!] \cap C = [\![D^Y]\!] \cap C$.

The net effect is that the expressions higher in the structure update the context first. The hierarchy-based account makes the right prediction for the Korean attitude verb example in (4), the syntax of which is provided in (12). The embedded clause is c-commanded by *John* and *mit* 'believe', thus the two items are taken into account and the interpreter can restrict the attention to John's doxastic worlds.

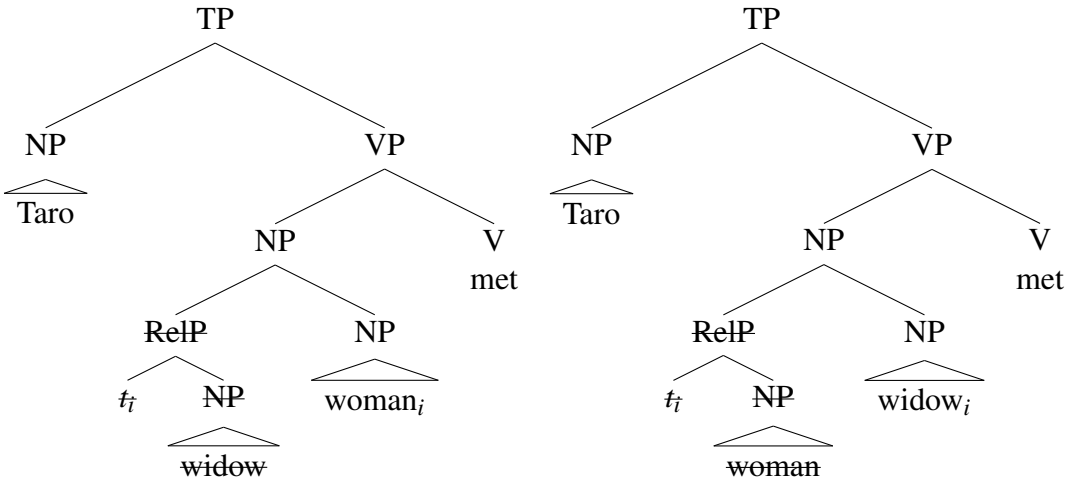
(12) (=(4))



As for the scrambling data in (7), the hierarchy-based account can assume that the scrambled embedded clause reconstructs at LF. This would yield the LF structure in (12). Since the local context is calculated on LF, the prediction is no different from the example that does not involve scrambling.

The hierarchy-based account also correctly predicts that the redundancy effect arises in (9b), but not in (9a). The syntax of (9a) and (9a) are provided in (13a) and (13b), respectively. In calculating the local context of the relative clause in (9a), only the expressions that c-command it are taken into account, hence *Taro*, *met*, and *woman*. So its local context can be restricted to the set of women that Taro met. Further updating the context with *widow* is informative, so the redundancy effect does not arise. On the other hand, the local context of the relative clause in (9b) is the set of widows that Taro met. Thus, it would be redundant to further update the context with *woman*.

(13) a. (=(9a)) b. (=(9b))



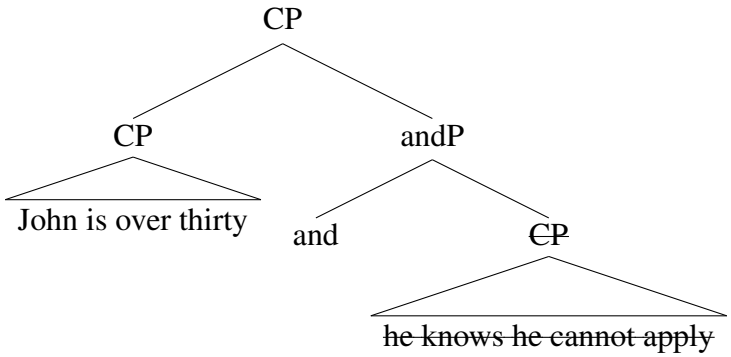
4. The problem of the hierarchy-based account: coordination

Romoli and Mandelkern (2017) cannot explain why contexts are invariably updated left-to-right in coordinated structures, despite the cross-linguistic variation in constituency. Let's first take a look at the English example in (14). Applying the incremental version of Schlenker's algorithm, the local context of the right conjunct is $C \wedge$ **john-is-over-thirty**, where C refers to the global context.

(14) John is over thirty and he knows he cannot apply.

The hierarchy-based account makes the same prediction. In calculating the local context of the right conjunct, only the items that c-command it are considered: the left conjunct and the coordinator *and*.

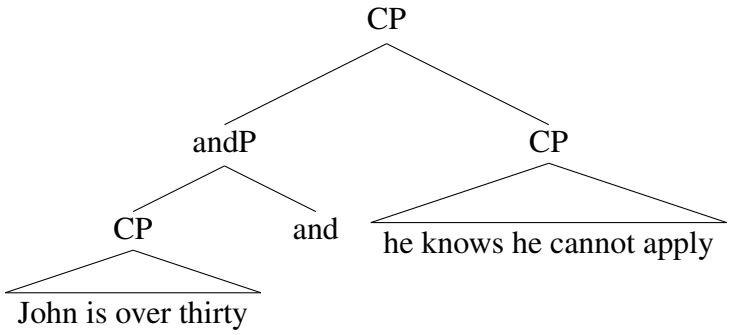
(15) The hierarchy-based account: prediction borne out



The issue arises in Korean (as well as other head-final languages), where the left conjunct and the conjunction operator form a constituent. In (16), the right conjunct c-commands the left conjunct.

(16) [John-un selun-i nem-ess-ko] caki-ka ciwenha-ci mosha-n-ta-nun
[John-TOP thirty-NOM over-perf-and] self-NOM apply-CI cannot-PRES-DECL-REL
kes-ul al-n-ta.
thing-ACC know-PRES-DECL
'John is over thirty and he knows he cannot apply.'

(17) The hierarchy-based account: prediction not borne out



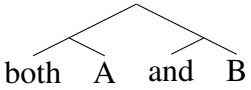
Given the structure in (17), the hierarchy-based account predicts that the right conjunct is updated before the left conjunct because the former c-commands the latter. However, just as in English, the entire sentence intuitively presupposes that 'if John is over thirty, he cannot apply'. The same problem arises in disjunction because its structure is identical to that of conjunction in many languages. The coordination data call for an algorithm that makes a robust prediction despite the variation in syntactic structure.

It is noteworthy to mention an alternative view: Chierchia (2009) argues that context update takes place in the order of semantic composition. He introduced the notion of f-command which is defined in terms of function/argument relation. Informally speaking, given two arguments of a function, the argument that first composes with the functor f-commands the other argument. Chierchia's analysis provided in (18) amounts to saying that the argument that f-commands the other updates the context first.

(18) f-command (Chierchia, 2009)
a. If A and B are co-arguments of f, A f-commands B iff the functional complex $f(A)$ containing A does not contain B. (= A is closer to f than B)
b. A provide the local context for B iff A immediately f-commands B.

For example, as for the conjunction operator *and*, its first argument gets to update the context before the second argument. This analysis makes the right prediction for the Korean conjunction example in (16), but then something more has to be said about (14): the right conjunct is the first argument of English *and*. Chierchia suggests that English has a null operator, *both*, that forms a consitutent with the left conjunct. The overt *and* is meaningless and the null operator carries the semantics of conjunction.

(19) Silent *both* as the conjunction operator



Postulating the null operator makes the right predication, however, it requires further evidence that the null operator forms a constituent with the left conjunct. To my knowledge, the overt counterpart of *both* cannot undergo movement together with the first conjunct, while stranding *and* and the right conjunct. There is no positive evidence that the two form a constituent.

Moreover, *both* can appear after the two conjuncts, as in (20a). If (20a) is in fact derived from (20b), it is reasonable to assume that *John and Mary* form a constituent, because they can be fronted while stranding *both*.

(20) a. John and Mary *both* went home.
b. *Both* John and Mary went home.

The discussion is not conclusive as the syntax and semantics of coordinated structures are controversial. In fact, Ingason (2016) makes the same point and claims that the coordination data is not a serious counterexample to the hierarchy-based account. But it is worthy of mention that the left-to-right bias naturally follows from linear order-based accounts.

5. Proposal

I maintain Schlenker's view that sentences are parsed from left to right. In addition, I assume that the interpreter constructs the syntactic structure of a given sentence during parsing (cf. Phillips 1996). Given these assumptions, I make one adjustment to Schlenker's algorithm based on considerations from the syntax-semantics interface.

I propose that the interpreter computes the local context of expressions only at certain points in the parsing process. Specifically, the equivalence in (1) is calculated only when the semantic value of the parsed expressions can be retrieved.

(21) **Adjustment: Domain-by-domin interpretation**
The interpreter parses a sentence from left to right, but the local context of an expression (either propositional or predicative) can be calculated only at points where the interpreter has access to the semantic values of the parsed expressions.

The reasoning is that the equivalence posited in (1) is semantic in nature. Entailment is a semantic notion which should operate on semantic values rather than strings. And it is commonly assumed that access to semantic values of expressions is limited to certain points in the derivation. The phase theory (Chomsky, 2008) is more or less the standard view, where the semantic information of syntactic items is shipped to the interface upon construction of either vP or CP (i.e., phases). Independently, continuation semantics (Barker and Shan, 2014; Charlow, 2014)

assumes that the semantic value of an expression can be retrieved by *evaluating* it, and only clauses are suitable targets for evaluation.

I assume along with continuation semantics that a clause is the domain of semantic evaluation. The net effect is that the interpreter needs to *postpone* the calculation of local context if the parsed expressions altogether do not constitute a clause; semantic information can only be retrieved from a clause. The analyses offered in the following section do not require the technical details of continuation semantics. It suffices to assume that the semantic value of an expression can only be fetched when the parsed expressions constitute a full clause.

6. Analysis

6.1. Attitude context

The proposed analysis forces the interpreter to postpone the computation of local context until the attitude verb has been parsed. Example (4) is repeated below as (22), and the table in (23) illustrates the parsing process.

(22) John-un [Mary-ka (cikum-to) keysokhayse tambay-lul pi-n-tako]
John-TOP [Mary-NOM (now-also) continuously cigarette-ACC smoke-PRES-COMP]
mit-nun-ta.
believe-PRES-DECL
'John believes that Mary continues to smoke.'

(23) Derivation of (22)

Step	State	Evaluate?	Target local context
1	John • [that Mary continues to smoke] believes	No	
2	John [that Mary continues to smoke] • believes	No	
3	John ~~[that Mary continues to smoke]~~ believes •	Yes	Embedded clause

The bullet points in (23) mark the positions of interest, at which the interpreter attempts to calculate the local context of the embedded clause. All expressions following the bullet points are ignored in computing the local context. The interpreter can only evaluate (i.e., retrieve the semantic value of the parsed expressions) at step 3, whereas doing so at step 1 or 2 is blocked. At step 1, the interpreter has only parsed *John*, which is not a full clause. Similarly, at step 2, the interpreter has encountered *John that Mary continues to smoke*. But again, the expressions do not form a clause. Thus, the calculation of the local context is delayed until step 3, the point at which the interpreter has access to the sentence-final *believes*.

6.2. Coordination

This section shows that the proposed analysis is robust to cross-linguistic variation in coordinated structures. The Korean conjunction example in (16) is repeated below as (24). The full derivation is provided in (25).

(24) [John-un selun-i nem-ess-ko] caki-ka ciwenha-ci mosha-n-ta-nun
[John-TOP thirty-NOM over-perf-and] self-NOM apply-CI cannot-PRES-DECL-REL
kes-ul al-n-ta.
thing-ACC know-PRES-DECL
'John is over thirty and he knows he cannot apply.'

(25) Derivation of (24)

Step	State	Evaluate?	Target local context
1	~~John is over thirty~~ • and he knows he cannot apply	Yes	Left conjunct
2	John is over thirty and • he knows he cannot apply	No	
3	John is over thirty and ~~he knows he cannot apply~~ •	Yes	Right conjunct

Just as in Schlenker's original formulation, the interpreter parses the sentence from left to right. At step 1, the interpreter has parsed the left conjunct. Since *John is over thirty* is a full clause, it can be evaluated. The general prediction is that cross-linguistic variation in coordinated structure is irrelevant to the order of context update. The left expression always updates the context before the right one.

6.3. Scrambling

The scrambling data is a challenge to any theory that relies on left-to-right bias. Since the scrambled embedded clause in (7), repeated below as (26), precedes the matrix clause, the interpreter first parses the embedded clause no matter what. On the other hand, the hierarchy-based account can assume that the embedded clause reconstructs before its local context is calculated.

(26) [Mary-ka keysokhayse tambay-lul pi-n-tako] John-un *t*
Mary-NOM continuously cigarette-ACC smoke-PRES-COMP John-TOP *t*
mit-nun-ta.
believe-PRES-DECL
(Lit.) 'That Mary continues to smoke, John believes.'

Having tied the points of semantic access to that of evaluation, it naturally follows from continuation semantics that the local context of the scrambled embedded clause is calculated after the matrix clause has been parsed. Barker (2009) develops a mechanism which handles recon-

struction effects without actually requiring to reconstruct, namely *delayed evaluation*. Barker claims that English wh-phrases can be interpreted in-situ and do not require reconstruction. Specifically, *delaying* the evaluation of a wh-phrase and evaluating the remaining expressions beforehand replicates the reconstruction effect. For example, despite the fact that the wh-phrase *who* in (27) linearly precedes the rest of the sentence, it is evaluated after *does John like t*. The technical details of delayed evaluation is offered in appendix B.

(27) Who does John like?

I extend Barker's analysis and claim that scrambled embedded clauses are also subject to delayed evaluation. In other words, the scrambled embedded clause is evaluated after the matrix clause has been processed. Since the local context of an expression can be calculated only when its semantic value can be retrieved, the interpreter has full access to the matrix clause when the local context of the scrambled embedded clause is calculated. The table in (28) depicts this process.

(28) Derivation of (26)

Step	State	Evaluate?	Target local context
1	Mary continues to smoke • John believes *t*	No	
2	Mary continues to smoke ~~John believes *t*~~ •	Yes	
3	~~Mary continues to smoke~~ John believes *t* •	Yes	Embedded clause

At step 1, the interpreter does not evaluate the scrambled embedded clause and waits for the matrix clause to be processed. At step 2, *John believes t* is evaluated but the evaluation of the embedded clause is delayed. Only at step 3 can the embedded clause be evaluated, and this is when its local context can be calculated as well. At this point, the interpreter is aware that the sentence is about John's beliefs.

I would like to emphasize that delayed evaluation is not a special mechanism invented to explain how presuppositions project in scrambling constructions. It merely offers an in-situ account of reconstruction effects. Nevertheless, the order in which the derivation unfolds provides a natural explanation of the presupposition projection behavior of such constructions.

6.4. Relative clause

The redundancy effect in (9b), repeated below as (29), is also accounted for. The derivation is provided in (30).

(29) # Taro-ga [[**zyosei**-dearu] **yamome**-ni] atta.
Taro-NOM [[**woman**-COP] **widow**-DAT] met
'Taro met a widow who is a woman.

Only after step 4 can the interpreter evaluate the parsed expressions. At step 5, all of the parsed expressions except the RelP will be taken into account. In other words, the following items are considered: *Taro*, *widow*, and *met*. This means that the local context of the RelP is the set of individuals x such that ***widow****(x)* and ***met****(x)(John)* are true. As in the hierarchy-based account, updating the local context with *that is woman* is redundant.

(30) Derivation of (29)

Step	State	Evaluate?	
1	Taro • [$_{NP}$ [$_{RelP}$ that is woman] widow] met	No	Target local context
2	Taro [$_{NP}$ [$_{RelP}$ that is woman] • widow] met	No	
3	Taro [$_{NP}$ [$_{RelP}$ that is woman] widow] • met	No	
4	Taro ~~[$_{NP}$ [$_{RelP}$ that is woman] widow]~~ met •	Yes	NP
5	Taro [$_{NP}$ ~~[$_{RelP}$ that is woman]~~ widow] met •	Yes	RelP

7. Conclusion

This paper presents a novel parsing-based account of presupposition projection which is robust to certain crosslinguistic variations in word order. While maintaining Schlenker's view that presupposition projection behavior is closely related to the left-to-right bias inherent in parsing, I hypothesize that local context is computed domain-by-domain, as opposed to word-by-word, and that clauses are such domains. The proposed analysis resolves the issues in Schlenker's original algorithm and the hierarchy-based variation.

A. Formal analysis

This section fleshes out the technical details of the proposed algorithm built on continuation semantics. The reader is referred to Barker and Shan (2014) for the interpretation of tower notations.

(31) Algorithm for computing the local context of an expression E

a. The interpreter traverses a given sentence from left to right. The syntactic structure is constructed on the way.

b. Upon parsing the expression E, check whether the sequence $A\ E$ can be evaluated (i.e., constitutes a clause), where A is the sequence of all of the expressions that precede E.

c. If the sequence $A\ E$ can be evaluated, the local context of E which occurs in a context C is the strongest restriction c such that for any proposition p or predicate P, the following equivalence holds:
(i) For proposition p:

$$C \vDash \textsc{evaluate}\left(\mathbf{A}\frac{[\ ]}{c' \textbf{ and } p}\right) \leftrightarrow \textsc{evaluate}\left(\mathbf{A}\frac{[\ ]}{p}\right)$$

(ii) For predicate P:

$$C \vDash \textsc{evaluate}\left(\mathbf{A}\frac{[\ ]}{c' \textbf{ and } P}\right) \leftrightarrow \textsc{evaluate}\left(\mathbf{A}\frac{[\ ]}{P}\right)$$

where **A** is the semantic tower of A and **and** is the generalized conjunction Op

d. If the sequence $A\ E$ cannot be evaluated, continue traversing until the interpreter can evaluate the sequence $A\ E\ B$, where B is the sequence of all expressions which follows E and was parsed by the interpreter.

e. When the parsed expressions can be evaluated, the local context of E is the strongest restriction c such that for any proposition p or predicate P, the following equivalence holds: (i) For proposition p:

$$C \vDash \textsc{evaluate}\left(\mathbf{A}\frac{[\ ]}{c' \textbf{ and } p}\mathbf{B}\right) \leftrightarrow \textsc{evaluate}\left(\mathbf{A}\frac{[\ ]}{p}\mathbf{B}\right)$$

(ii) For predicate P:

$$C \vDash \textsc{evaluate}\left(\mathbf{A}\frac{[\ ]}{c' \textbf{ and } P}\mathbf{B}\right) \leftrightarrow \textsc{evaluate}\left(\mathbf{A}\frac{[\ ]}{P}\mathbf{B}\right)$$

where **A** and **B** are the semantic tower of A and B, respectively, and **and** is the generalized conjunction Op

Sample derivations for the Korean attitude verb example (ex (22)) and the coordination example (ex (24)) are provided in (32) and (33), respectively.

(32) Derivation of (22)

a. The interpreter parses the sentence from left to right and reaches the end

$$\left(\begin{array}{ccc}\text{John} & \left(\begin{array}{cc}\text{that Mary continues to smoke} & \text{believes}\\ \frac{[\ ]}{\textbf{continues\_to\_smoke}(\mathbf{m})} & \frac{[\ ]}{\textbf{believes}}\end{array}\right) & \bullet\\ \frac{[\ ]}{\mathbf{j}} & & \end{array}\right)$$

b. Replace the embedded clause with $\frac{[\ ]}{c' \text{ and } p} \equiv \frac{[\ ]}{c' \wedge p}$

$$\left(\begin{array}{ccc}\text{John} & \left(\begin{array}{cc}\sout{\text{that Mary continues to smoke}} & \text{believes}\\ \frac{[\ ]}{c' \wedge p} & \frac{[\ ]}{\textbf{believes}}\end{array}\right) & \bullet\\ \frac{[\ ]}{\mathbf{j}} & & \end{array}\right)$$

c. Corresponding equivalence:

$$C \vDash \textsc{evaluate}\left(\frac{[\ ]}{\textbf{believes}(c' \wedge p)(\mathbf{j})}\right) \leftrightarrow \textsc{evaluate}\left(\frac{[\ ]}{\textbf{believes}(p)(\mathbf{j})}\right)$$

$$\Leftrightarrow C \vDash \forall w \in \textsc{dox}(\mathbf{j}) : c'(w) = 1 \wedge p(w) = 1 \leftrightarrow \forall w \in \textsc{dox}(\mathbf{j}) : p(w) = 1$$

(33) Derivation of (24)

a. The interpreter parses the sentence from left to right and reaches the end of the left conjunct. The expressions that follow the bullet point are ignored.

$$\left(\left(\begin{array}{c}\text{John is over thirty}\\ \frac{[\ ]}{\textbf{over-30}(\textbf{j})}\end{array}\bullet\begin{array}{c}\text{and}\\ \frac{[\ ]}{\lambda p\lambda q.p\wedge q}\end{array}\right)\begin{array}{c}\text{he knows he cannot apply}\\ \frac{[\ ]}{\textbf{knows}(\textbf{cannot-apply}(\textbf{j}))(\textbf{j})}\end{array}\right)$$

b. Compute LC of the left conjunct: replace the left conjunct with $\frac{[\ ]}{c' \text{ and } p} \equiv \frac{[\ ]}{c'\wedge p}$

$$\left(\left(\begin{array}{c}\cancel{\text{John is over thirty}}\\ \frac{[\ ]}{c'\wedge p}\end{array}\bullet\begin{array}{c}\text{and}\\ \frac{[\ ]}{\lambda p\lambda q.p\wedge q}\end{array}\right)\begin{array}{c}\text{he knows he cannot apply}\\ \frac{[\ ]}{\textbf{knows}(\textbf{cannot-apply}(\textbf{j}))(\textbf{j})}\end{array}\right)$$

c. Corresponding equivalence:

$$C \vDash \text{EVALUATE}\left(\frac{[\ ]}{c'\wedge p}\right) \leftrightarrow \text{EVALUATE}\left(\frac{[\ ]}{p}\right)$$

$$\Leftrightarrow C \vDash c'\wedge p \leftrightarrow p$$

d. The interpreter continues to parse and reaches the end of the sentence

$$\left(\left(\begin{array}{cc}\text{John is over thirty} & \text{and}\\ \frac{[\ ]}{\textbf{over-30}(\textbf{j})} & \frac{[\ ]}{\lambda p\lambda q.p\wedge q}\end{array}\right)\begin{array}{c}\text{he knows he cannot apply}\\ \frac{[\ ]}{\textbf{knows}(\textbf{cannot-apply}(\textbf{j}))(\textbf{j})}\end{array}\right)\bullet$$

e. Compute LC of the right conjunct: replace the right conjunct with $\frac{[\ ]}{c' \text{ and } p} \equiv \frac{[\ ]}{c'\wedge p}$

$$\left(\left(\begin{array}{cc}\text{John is over thirty} & \text{and}\\ \frac{[\ ]}{\textbf{over-30}(\textbf{j})} & \frac{[\ ]}{\lambda p\lambda q.p\wedge q}\end{array}\right)\begin{array}{c}\cancel{\text{he knows he cannot apply}}\\ \frac{[\ ]}{c'\wedge p}\end{array}\right)\bullet$$

$$= \left(\begin{array}{c}\text{John is over thirty and } \cancel{\text{he knows he cannot apply}}\\ \frac{[\ ]}{\textbf{over-30}(\textbf{j})\wedge(c'\wedge p)}\end{array}\right)\bullet$$

f. Corresponding equivalence:

$$C \vDash \text{EVALUATE}\left(\frac{[\ ]}{\textbf{over-30}(\textbf{j})\wedge(c'\wedge p)}\right) \leftrightarrow \text{EVALUATE}\left(\frac{[\ ]}{\textbf{over-30}(\textbf{j})\wedge p}\right)$$

$$\Leftrightarrow C \vDash \textbf{over-30}(\textbf{j})\wedge(c'\wedge p) \leftrightarrow \textbf{over-30}(\textbf{j})\wedge p$$

B. Scrambling as delayed evaluation

Delayed evaluation can be schematized as in (34). Given two expressions, the right one is first evaluated. The semantic value of the right expression is fed into the left expression, yielding the composed value of the two expressions.

(34) Schema: delayed evaluation

	Tower convention	Lambda formula
a. Initial setup	$\frac{g[\ ]}{f}\ \frac{h[\ ]}{i}$	$\lambda k.g(k(f))\quad \lambda k.h(k(i))$
b. Evaluate the right.exp	$\frac{g[\ ]}{f}\quad h(i)$	$\lambda k.g(k(f))\quad h(i)$
c. Feed the right.exp to the left.exp	$g(h(i)(f))$	$g(h(i)(f))$

A sample derivation of the sentence *Who does John like?* is provided in (35).

(35) Sample derivation of *Who does John like?*

a. Initial set up

$$\left(\frac{\begin{array}{c}\text{Who}\\ \mathbf{who}(\lambda x.[\ ])\end{array}}{x}\right)\left(\begin{array}{c}\text{John}\\ \frac{[\ ]}{\mathbf{j}}\end{array}\ \left(\frac{\begin{array}{c}\text{like}\\ [\ ]\end{array}}{\mathbf{like}}\ \frac{\begin{array}{c}t\\ \lambda y.[\ ]\end{array}}{y}\right)\right)$$

b. Reduce the right.exp

$$\left(\frac{\begin{array}{c}\text{Who}\\ \mathbf{who}(\lambda x.[\ ])\end{array}}{x}\right)\left(\frac{\begin{array}{c}\text{John like } t\\ \lambda y.[\ ]\end{array}}{\mathbf{like}(y)(\mathbf{j})}\right)$$

c. Evaluate the right.exp

$$\left(\frac{\begin{array}{c}\text{Who}\\ \mathbf{who}(\lambda x.[\ ])\end{array}}{x}\right)\left(\begin{array}{c}\text{John like } t\\ \lambda y.\mathbf{like}(y)(\mathbf{j})\end{array}\right)$$

d. Feed the right.exp to the left.exp (function application)

$$\left(\begin{array}{c}\text{Who (does) John like } t\\ \mathbf{who}(\lambda x.\mathbf{like}(x)(\mathbf{j}))\end{array}\right)$$

Scrambling constructions receives a similar treatment. The evaluation of the scrambled embedded clause (ex (26)) is delayed.

(36) Derivation of (26)

a. The interpreter parses the fronted embedded clause, but delays the evaluation

$$\left(\frac{\begin{array}{c}\text{that Mary continues to smoke}\\ [\ ]\end{array}}{\mathbf{continues\_to\_smoke(m)}}\right)\bullet\left(\begin{array}{c}\text{John}\\ \frac{[\ ]}{\mathbf{j}}\end{array}\ \left(\frac{\begin{array}{c}t\\ \lambda p.[\ ]\end{array}}{p}\ \frac{\begin{array}{c}\text{believes}\\ [\ ]\end{array}}{\mathbf{believes}}\right)\right)$$

b. The interpreter reaches the end of the sentence

$$\left(\begin{array}{c}\text{that Mary continues to smoke}\\ \frac{[\ ]}{\textbf{continues\_to\_smoke(m)}}\end{array}\right)\left(\begin{array}{c}\text{John}\\ \frac{[\ ]}{\textbf{j}}\end{array}\ \left(\begin{array}{cc}t & \text{believes}\\ \frac{\lambda p.[\ ]}{p} & \frac{[\ ]}{\textbf{believes}}\end{array}\right)\right)\bullet$$

c. Reduce the right.exp

$$\left(\begin{array}{c}\text{that Mary continues to smoke}\\ \frac{[\ ]}{\textbf{continues\_to\_smoke(m)}}\end{array}\right)\left(\begin{array}{c}\text{John } t \text{ believes}\\ \frac{\lambda p.[\ ]}{\textbf{believes}(p)\textbf{(j)}}\end{array}\right)\bullet$$

d. Evaluate the right.exp

$$\left(\begin{array}{c}\text{that Mary continues to smoke}\\ \frac{[\ ]}{\textbf{continues\_to\_smoke(m)}}\end{array}\right)\left(\begin{array}{c}\text{John } t \text{ believes}\\ \lambda p.\textbf{believes}(p)\textbf{(j)}\end{array}\right)\bullet$$

e. Replace the embedded clause with $\frac{[\ ]}{c' \text{ and } p} \equiv \frac{[\ ]}{c' \wedge p}$

$$\left(\begin{array}{c}\sout{\text{that Mary continues to smoke}}\\ \frac{[\ ]}{c' \wedge p}\end{array}\right)\left(\begin{array}{c}\text{John } t \text{ believes}\\ \lambda p.\textbf{believes}(p)\textbf{(j)}\end{array}\right)\bullet$$

f. Evaluate the entire sentence: Feed the right.exp to the left.exp

$$\left(\begin{array}{c}\sout{\text{that Mary continues to smoke}}, \text{John } t \text{ believes}\\ \textbf{believes}(c' \wedge p)\textbf{(j)}\end{array}\right)$$

$$= \left(\begin{array}{c}\sout{\text{that Mary continues to smoke}}, \text{John } t \text{ believes}\\ \forall w \in \textsc{dox}(\mathbf{j}) : c'(w) = 1 \wedge p(w) = 1\end{array}\right)$$

g. Corresponding equivalence

$$C \vDash \forall w \in \textsc{dox}(\mathbf{j}) : c'(w) = 1 \wedge p(w) = 1 \leftrightarrow \forall w \in \textsc{dox}(\mathbf{j}) : p(w) = 1$$

The equivalence derived in (36) matches that of (32). This is indeed the desired consequence.

References

Barker, C. (2009). Reconstruction as delayed evaluation. In E. W. Hinrichs and J. A. Nerbonne (Eds.), *Theory and Evidence in Semantics*, pp. 1–28. CSLI Publications.

Barker, C. and C.-c. Shan (2014). *Continuations and Natural Language*. Oxford University Press.

Charlow, S. (2014). *On the Semantics of Exceptional Scope*. Ph. D. thesis, New York University.

Chierchia, G. (2009). On the explanatory power of dynamic semantics. Handout from talk at Sinn und Bedeutung 14.

Chomsky, N. (2008). On phases. In *Current Studies in Linguistics Series*, Volume 45, pp. 133. MIT press.

Heim, I. (1983). On the Projection Problem for Presuppositions. In *Proceedings of the Second West Coast Conference on Formal Linguistics*, pp. 114–125.

Ingason, A. K. (2016). Context updates are hierarchical. *Glossa: a journal of general linguistics 1*(1), 1–9.

Phillips, C. (1996). *Order and structure*. Ph. D. thesis, Massachusetts Institute of Technology.

Romoli, J. and M. Mandelkern (2017). Hierarchical Structure and Local Contexts. In *Proceedings of Sinn und Bedeutung 21*.

Schlenker, P. (2009). Local Contexts. *Semantics and Pragmatics 2*(3), 1–78.

Schlenker, P. (2010). Presuppositions and Local Contexts. *Mind 119*(474), 377–391.

Schlenker, P. (2011a). Presupposition Projection: Two Theories of Local Contexts - Part I. *Language and Linguistics Compass 5*(12), 848–857.

Schlenker, P. (2011b). Presupposition Projection: Two Theories of Local Contexts - Part II. *Language and Linguistics Compass 5*(12), 858–879.

Stalnaker, R. C. (1974). Pragmatic Presuppositions. In M. Kunitz and P. Unger (Eds.), *Semantics and Philosophy*, pp. 197–213. New York University Press.