Long-distance movement, binding, and scope in a continuation grammar

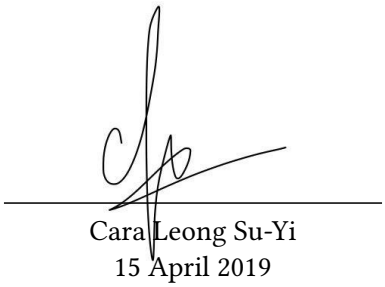
Cara Leong Su-Yi

An Honours Thesis submitted in part fulfilment
of the requirements for the degree of
Bachelor of Arts with Honours in English Language

Department of English Language and Literature
Faculty of Arts and Social Sciences
National University of Singapore
Singapore

15 April 2019

This Honours Thesis represents my own work and due acknowledgement is given whenever information is derived from other sources. No part of this Honours Thesis has been or is being concurrently submitted for any other qualification at any other university.



Cara Leong Su-Yi
15 April 2019

ACKNOWLEDGMENTS

This thesis would not exist without my advisor, Dr Michael Yoshitaka Erlewine. All the good ideas herein are mostly due to him; all the bad ones that remain were made less terrible by him. I am tremendously grateful for his discernment, enthusiasm, and fortitude in the face of nonsensical notation.

This work has benefited from the insight of my linguist-friends. Thanks to Sharmala Solomon and Lauren Koh for accepting the presupposition that unicorns exist, and for my knives. Anne Ng dug out years-old essays in service of this paper. Thanks also to Keely New for grammaticality judgments, proofreading, and documenting the crucial moments after I poured an entire glass of chocolate milk onto my laptop three days before this thesis was due.

I probably haven't said this enough, but I owe almost everything to the friends who have seen me through university. Anne and Shien suffered through the worst of it — I'm ever so grateful. *Truly, please.* Ellen was there from day one. I miss Nicole and Kimi Raikkonen. Si En has never made a bad suggestion, and introduces me to new ways of thinking every time. Royston once told me my thesis was "interesting within the domain", which I will take as a compliment any day. Ruizhi walked alongside on the writing journey mostly unironically. Leon planted 2020 dreams. Yuchuan bought me a kaya waffle a week. Darren housed my refrigerated goods and midnight worries. Thank you all for propping me up.

Finally, all gratitude to my family — I feel impossibly privileged to be yours. Thanks for providing snacks and not asking how my thesis was going. Words aren't enough.

And to You, an open proposition if ever I knew one — still, always, thank You.

CONTENTS

LIST OF DEFINITIONS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

B&S	Barker and Shan 2014
DP	determiner phrase
QP	quantifier phrase
QR	quantifier raising
S	sentence
TCC	Tensed Clause Condition

ABSTRACT

In this thesis, I describe and evaluate the continuation-based account of quantifier scope, variable binding and movement presented in Barker and Shan 2014. I show that under this account, it is possible to independently enforce the clause-boundedness of quantifier scope while accounting for movement, treat long-distance movement without variable binding, and deal with variable binding in local movement configurations. I also point out that the uniform treatment of scope, movement and binding through continuations makes fundamentally flawed predictions when faced with the combination of clause-bound quantifiers, long-distance movement, and variable binding. Particularly, although the continuation framework claims to account for crossover effects, I show that such an account is lost in long-distance movement configurations. I suggest that this problem has its roots in operations fundamental to the framework, and thus poses a serious challenge to the validity of the theory.

CHAPTER 1
INTRODUCTION

This thesis concerns itself with the treatment of quantifier scope, movement and variable binding in the semantic framework of Barker and Shan 2014. On this theory, the interplay between linear and scopal relationships is key; such an interaction harkens back to the seminal problem in semantics of quantifier scope ambiguity. May (1977) notes that a single form such as (1) can have two different readings:

(1) Some man loves every woman.

On one reading, a single man loves all women. On this "surface scope" reading, the existential quantifier phrase (QP) *some man* is said to take wide scope over the universal QP *every woman* ($\exists > \forall$). On the second reading, in which the existential takes narrow scope ($\forall > \exists$), a potentially different man loves each woman. The availability of this "inverse scope" reading does not seem apparent based on surface word order.

A common approach to of quantifier scope ambiguity posits the *covert movement* of quantifiers, or *quantifier raising* (QR), as an explanation for the availability of inverse scope readings. On this theory, objects are covertly moved to allow them to take scope, such that (1) is covertly analyzed as (2):

(2) **Logical Form (LF)**: every woman, some man loves.

In contrast, so called "in-situ" theories of scope-taking allow for both surface and inverse scope readings to be obtained without appealing to covert movement. In particular, Partee (1987) introduced the notion of type-shifting to allow for in-situ analyses of inverse scope, while Hendriks (1993)'s Flexible Types analysis suggests multiple denotations for verbs and quantifiers.

Against this background, Barker and Shan (2014) — henceforth B&S — present a grammar that provides an in-situ analysis of quantifier scope ambiguity (see also Barker 2002; Barker and Shan 2008; Kiselyov and Shan 2014; Shan 2007; Shan and Barker 2006). The chief intuition in their framework, which I call the continuation framework, is that some expressions take their *continuations*, or the immediate context surrounding them, as arguments. Working under this hypothesis, B&S build a fragment that uses continuations to capture the scope-taking behavior of quantifiers. They then extend the use of continuations to in-situ analyses of movement and variable binding.

In this thesis, I consider issues surrounding such a uniform treatment of scope, binding and movement in the continuation framework, and ultimately show that the continuation framework runs into serious problems when attempting to account for the combination of clause-bounded scope, long-distance movement, and variable binding. Chapter 2 introduces the continuation framework and shows how it uses type-shifting to provide an in-situ account of inverse scope and movement. In Chapter 3, I problematize unconstrained scope-taking behavior in the continuation framework and implement a constraint that I term the Tensed Clause Condition to account for the clause-boundedness of quantifiers. In Chapter 4, I turn to B&S's account of variable binding and its interaction with movement. I point out a flaw in the way B&S characterize scopal and linear relationships, and suggest why this flaw may be fatal to the framework. Chapter 5 concludes.

CHAPTER 2
A CONTINUATION GRAMMAR

In this chapter, I summarize the notational system and operations developed by B&S. I show how the continuation framework provides in-situ accounts of inverse scope and the extraction of expressions from their canonical positions. I then briefly discuss the value of working in the continuation framework.

2.1 Continuations

A *delimited continuation* of an expression is "a portion of the context surrounding [that] expression" (Barker and Shan 2014:1) which represents "the computational future of an expression, i.e. what is about to happen to it" (Shan and Barker 2006:95). For example, the continuation of *everyone* in *John gave everyone a pen* is what remains of the clause after *everyone* is taken out of it: "John gave [ ] a pen"; the continuation of *gave everyone a pen* in the same sentence is "John [ ]". The continuation of an expression a of semantic type α embedded within a context b of semantic type β is thus a function from $\alpha \longrightarrow \beta$ (Shan and Barker 2006). Reifying the former continuation gives a function with the informal denotation λx . **gave** x **pen John**[1], i.e. a function from individuals to truth values. Happily, such a denotation corresponds with the nuclear scope of *everyone*, making continuations a useful way of describing and implementing in-situ quantifier scope-taking.

2.2 Semantic types and syntactic categories

The continuation framework is operationalized in the form of a combinatory categorial grammar in the style of Jacobson 1999 and Steedman and Baldridge 2011. B&S propose a

1. I refer to expressions using *italics* and represent denotations in **boldface**. Following B&S, I use the notation of predicate logic to represent denotations. In denotations, values associate from left to right, so an expression written **gave Mary pen John** is equivalent to **((gave Mary) pen) John**

grammar with basic syntactic categories *DP* and *S* whose corresponding semantic types are individuals (type *e*) and truth values (type *t*) respectively. Complex syntactic categories are of the form $A \backslash B$, B / A, $A \mathbin{\backslash\!\!\backslash} B$, $B \mathbin{/\!\!/} A$[2], indicating expressions with the semantic type $\langle \alpha, \beta \rangle$, where *A* and *B*[3] are syntactic categories with semantic types α and β respectively.

In the standard mode of composition, a functor of syntactic category $A \backslash B$ and semantic type $\langle \alpha, \beta \rangle$ takes an argument located immediately to its left of category *A* and type α to produce a result of type *B*. For instance, in (3), *cries*, of category $DP \backslash S$ and semantic type $\langle e, t \rangle$, takes as its argument the *DP John* with semantic type *e* to form the expression *John cries* of category *S* and semantic type *t*:

(3)
$$\begin{pmatrix} DP & DP \backslash S \\ \text{John} & \text{cries} \\ \textbf{John} & \textbf{cries} \end{pmatrix} = \begin{matrix} S \\ \text{John cries} \\ \textbf{cries John} \end{matrix}$$

B&S represent expressions as a linear combination of towers. Towers consist of a stacked triple: the top section of a tower corresponds to the syntactic category of the expression; the middle section indicates the orthographic/phonological form of the expression; the bottom section of a tower indicates an expression's denotation. Composition goes through if the syntactic categories of adjacent expressions are compatible — in this case, if one expression is a functor that takes in the other as an argument.

In more complex configurations, the syntactic categories of adjacent expressions may not immediately match. For instance, in (4), the initial syntactic categories of *John* and *loves* are incompatible. However, after combining with a direct object *Mary*, the expression *loves Mary* is compatible with *John*, and combines to derive a complete sentence.

---

2. Barker and Shan (2006, 2014) also introduce categories corresponding to variable binding and *wh*-questions. See Chapter 4 for discussion on binding. I do not discuss *wh*-questions here.

3. Here and elsewhere, I use capital letters to represent arbitrary syntactic categories.

(4)
$$\left(\begin{array}{ccc} DP & \left(\begin{array}{ccc} (DP\backslash S)/DP & & DP \end{array}\right) \\ \text{John} & \text{loves} & \text{Mary} \\ \textbf{John} & \textbf{loves} & \textbf{Mary} \end{array}\right) = \left(\begin{array}{cc} DP & DP\backslash S \\ \text{John} & \text{loves Mary} \\ \textbf{John} & \textbf{loves Mary} \end{array}\right)$$

$$S$$
$$= \text{John loves Mary}$$
$$\textbf{loves Mary John}$$

B&S extend their grammar by introducing two new complex category-forming markers, $\diagdown\!\!\diagdown$ and $\diagup\!\!\diagup$, that "govern scope-taking where an in-situ quantifier is pronounced and where it takes scope" (Barker and Shan 2008:8). Broadly, an expression of category $A \diagdown\!\!\diagdown B$ can be thought of as an expression that would be a B if it surrounded an A, while an expression of category $B \diagup\!\!\diagup A$ can be thought of as an expression that would be a B if it were surrounded by an A.

Specifically, while the $\backslash$ operator indicates adjacency, $\diagdown\!\!\diagdown$ marks containment. Consider two items x, y with syntactic categories $A\backslash B$ and $A\diagdown\!\!\diagdown B$ respectively. x and y have the same semantic type $\langle\alpha, \beta\rangle$ and both take an argument of category A, but differ syntactically and conceptually in where their arguments are located. The relationship between a functor of category $A\backslash B$ and its argument A is that of *adjacency*: a functor of category $A\backslash B$ looks for an argument of category A that is immediately adjacent to it on its left (*mutatis mutandis* with a functor of category A / B and an argument on its right). In contrast, the relationship between a functor of category $A \diagdown\!\!\diagdown B$ and its argument is one of *containment*; a functor of category $A \diagdown\!\!\diagdown B$ looks for an argument of category A somewhere within its scope in order to form a B.

Schematically, the relationship between functors and their arguments is represented in Figure 1. In Figure 1a, a functor of category $A\backslash B$ is represented by the white trapezoid, and takes as an argument an item of category A that is adjacent to it on the left. In Figure 1b, a functor of category $A \diagdown\!\!\diagdown B$ expects an argument of category B that is located in a specific place within it.

If $\diagdown\!\!\diagdown$ represents containment, then $\diagup\!\!\diagup$ is its opposite — an $A \diagup\!\!\diagup B$ is an expression that would be an A if it were surrounded by a B. In many cases, the category of B is likely to be that

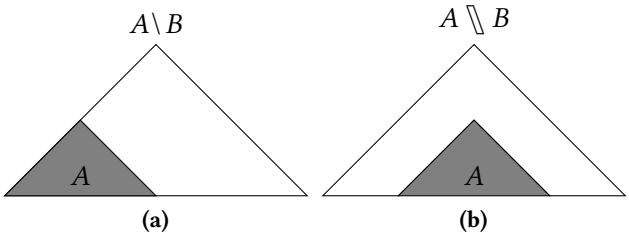


Figure 1: Schemata of relationship between functors and arguments

of a continuation, since continuations look for expressions to surround.

These new operators find a natural application in categorizing quantificational DPs, such as *everyone* in (5). In particular, the operators account for the fact that quantifier phrases can take scope over their own continuations.

(5) John said everyone ate cake.

The nuclear scope that *everyone* takes is its continuation, *John said [ ] ate cake*. In this continuation, and elsewhere in this paper, [ ] represents a hole in the denotation into which content can be inserted. Armed with a way to describe expressions that contain missing subparts within them, we can now ascribe such a string the category of a sentence missing a DP, $DP \backslash\!\!\backslash S$. We also know that when *everyone* combines with its continuation, it forms a complete sentence of category S. We can then describe the syntactic category of *everyone*: it needs to combine with a $DP \backslash\!\!\backslash S$ that surrounds it, and over which it takes scope, to return an S. In other words, it is of category $S /\!\!/ (DP \backslash\!\!\backslash S)$, and has the semantic value $\lambda\kappa \,.\, \forall x \,.\, \kappa\, x$, where κ is a relation over entities of type $\langle e, t\rangle$. Such a denotation corresponds neatly with a quantificational phrase's traditional semantic type $\langle\langle e, t\rangle, t\rangle$.

2.3 Tower notation

So far, we have seen single-layered towers that mark the syntactic category, orthographic form and semantic value of an expression. B&S introduce *tower notation* as a way to neatly

represent and compose scopal expressions by allowing syntactic categories and semantic values to be multi-layered. These multi-layered representations are equivalent to their linear representations. However, using the tower notation visually separates the scopal aspects of a scope-taking expression, which live on the *upper layers* of the syntactic and semantic tower, from the expression's non-scopal value, located in the *bottom layer* of the towers. Here and below, A, B, C, D, E refer to any valid syntactic categories.

Definition 1: Tower notation.

$$\begin{array}{ccc} C \mathbin{/\!\!/} (A \mathbin{\backslash\!\!\backslash} B) & \dfrac{C \mid B}{A} & \left.\vphantom{\dfrac{a}{b}}\right] \text{ syntactic category} \\ \text{expression} & = \text{expression} & \left.\vphantom{\dfrac{a}{b}}\right] \text{ phonetic form} \\ \lambda\kappa \,.\, f[\kappa a] & \dfrac{f[\,]}{a} & \left.\vphantom{\dfrac{a}{b}}\right] \text{ semantic value} \end{array}$$

Given an expression of syntactic category $\dfrac{C \mid B}{A}$, we can say that the expression takes in a continuation of type $A \mathbin{\backslash\!\!\backslash} B$ (given the semantic value κ in the definition above) to scope over. Internally, the expression acts like an A with respect to the continuation, plugging its value a into the continuation to give an expression of category B. Then, the expression applies operations on B, semantically represented by applying scopal effects f over κa. Doing so produces an expression of category C.

As an example, the universal QP *everyone* is represented in tower notation in (6):

(6)
$$\begin{array}{ccc} S \mathbin{/\!\!/} (DP \mathbin{\backslash\!\!\backslash} S) & & \dfrac{S \mid S}{DP} \\ \text{everyone} & = & \text{everyone} \\ \lambda P \,.\, \forall x \,.\, Px & & \dfrac{\forall x \,.\, [\,]}{x} \end{array}$$

Here, *everyone* acts like a DP, takes scope over an S and returns an S. Its local value is x, while its scopal value $\forall x$ will take scope over its continuation.

Using tower notation simplifies how we compose multi-level expressions, by indicating how we can build scopal (upper-level) and non-scopal (bottom-level) values simultaneously, as in the schema below:

Definition 2: Combination schema in the continuation framework. For any categories A, B, C, D, E:

$$\begin{array}{ccccc} \dfrac{A \mid B}{C/D} & \dfrac{B \mid E}{D} & & \dfrac{A \mid E}{C} \\ \text{left-exp} & \text{right-exp} & = & \text{left-exp right-exp} \\ \dfrac{f[\,]}{h} & \dfrac{g[\,]}{x} & & \dfrac{f[[g[\,]]}{h(x)} \end{array}$$

$$\begin{array}{ccccc} \dfrac{A \mid B}{D} & \dfrac{B \mid E}{C \setminus D} & & \dfrac{A \mid E}{D} \\ \text{left-exp} & \text{right-exp} & = & \text{left-exp right-exp} \\ \dfrac{f[\,]}{x} & \dfrac{g[\,]}{h} & & \dfrac{f[g[\,]]}{h(x)} \end{array}$$

Semantically, composing the two expressions amounts to combining the non-scopal elements in the bottom layer under functional application, while the scope-relevant parts of the expression combine from left to right — the scopal effects of a linearly preceding expression thus scope over the scopal effects on the same layer of expressions that succeed it.

In order for any two expressions to combine, two conditions must be satisfied. Firstly, in the upper, scopal tier, the right-hand category of the left expression and left-hand category of the right expression must match. Intuitively, a match means that the left expression takes in a completed expression of the same category as the right-hand expression produces. Secondly, in the bottom layer, one expression must take the other as an argument — this requirement straightforwardly allows functional application to occur. Combining the two expressions is then equivalent to the left expression taking scope over the entire

filled right expression of category B, which itself is filled by way of composing a functor of category $C \backslash D$ with an argument of category D. Composition is thus couched in the continuation framework as inherently scopal — any time expressions compose, the scope-taking side effects of each expression interleave from top to bottom, left to right. The combination operation defined in Definition 2 is visually represented in Figure 2:

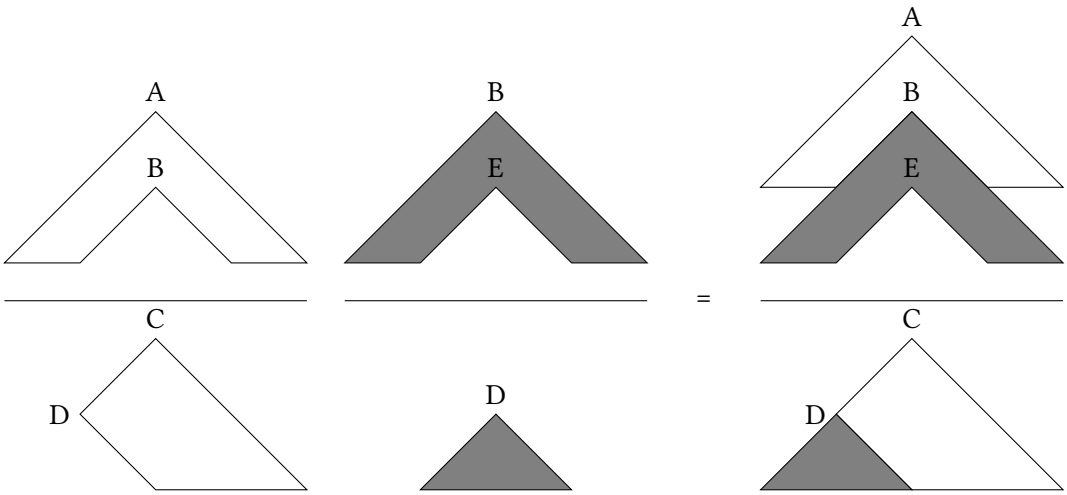


Figure 2: Schema for composing towers (Barker and Shan 2014)

2.4 Type-shifting in towers

In order to compose inherently scope-taking expressions such as QPs in the continuation framework with non-scopal expressions, B&S propose the LIFT and LOWER type-shifters, which deal with scopal behavior in towers. Type-shifters apply "freely and without constraint" (Barker and Shan 2014:10).

2.4.1 LIFT

B&S propose the LIFT type-shifter, a generalization of Partee (1987)'s LIFT type-shifter, which turns non-scope-taking expressions into trivially scope-taking expressions in order for them to compose with scope-takers.

Definition 3: LIFT type shifter. In general, for all categories A and B, for all semantic values x:

$$\begin{array}{ccc} A & & \dfrac{B \mid B}{A} \\ \text{expression} & \overset{\text{LIFT}}{\Longrightarrow} & \text{expression} \\ x & & \dfrac{[\,]}{x} \end{array}$$

The LIFT-ed expression takes as an argument a continuation of type $A \backslash\!\backslash B$ over which it 'takes scope', and returns a result of type B by applying its original value to the continuation. Since the semantics of the expression should not change, although the LIFT-ed expression takes scope over its context [ ], it does not make any changes and simply "plugs itself" (Kiselyov and Shan 2014:113) into its context. Correspondingly, LIFT-ing an expression is semantically equivalent to generating an identity function into which the expression is fed: for an expression of category A whose semantic value is x, the semantic value of the LIFT-ed A is $\dfrac{[\,]}{x} \equiv \lambda\kappa\ .\ \kappa x$, where κ has the semantic type $\langle\alpha, \beta\rangle$.

LIFT can be used, for instance, on a proper noun of category DP to turn it into a generalized quantifier of category $S \mathbin{/\!\!/} (DP \backslash\!\backslash S)$. I call the syntactic category of the lifted layer (B in the definition above) the *target* of LIFT.

(7)
$$\begin{array}{ccccc} & & \dfrac{S \mid S}{} & & \\ DP & & \dfrac{DP}{} & & S \mathbin{/\!\!/} (DP \backslash\!\backslash S) \\ \text{John} & \overset{\text{LIFT}}{\Longrightarrow} & \text{John} & = & \text{John} \\ \textbf{John} & & \dfrac{[\,]}{\textbf{John}} & & \lambda P\ .\ P\,\textbf{John} \end{array}$$

LIFT-ing is not limited to noun phrases. In (8), this is illustrated through the use of LIFT on a verb phrase, where the expression *cried* is LIFT-ed in order to combine with the scope-taker *everybody*.

(8)

$$\frac{\frac{S \mid S}{DP}}{\frac{\forall x\ .\ [\,]}{x}}\ \begin{array}{c}\text{everybody}\\ \text{cried}\\ \textbf{cried}\end{array}\ DP\backslash S \xRightarrow{\text{LIFT}} \frac{\frac{S \mid S}{DP}}{\frac{\forall x\ .\ [\,]}{x}}\ \begin{array}{c}\text{everybody}\\ \text{cried}\\ [\,]\end{array}\ \frac{\frac{S \mid S}{DP\backslash S}}{\textbf{cried}}$$

$$\frac{S \mid S}{S}$$

$$= \text{everybody cried}\ \frac{\forall x\ .\ [\,]}{\textbf{cried}\ x}$$

[4]

In (8), LIFT-ing *cried* with the target category S gives it an empty context that it 'takes scope' over. While the expression receives its continuation of category $(DP\backslash S) \setminus\!\!\setminus S$, it does not apply any scopal effects and instead only feeds its local value, *cried*, into the bottom layer of computation. While this move may seem vacuous, applying LIFT makes it possible to merge *cried* with *everyone*, which started out as a two-layered tower.

LIFT is also able to lift towers to an arbitrary number of levels. The category of the input A to the LIFT type-shifter is unconstrained, meaning that even complex categories such as $B /\!\!/ (A \setminus\!\!\setminus B)$ are suitable to be LIFT-ed as well, by setting the entire category $\frac{B \mid B}{A}$ as the bottom level of the tower. I demonstrate this in (9):

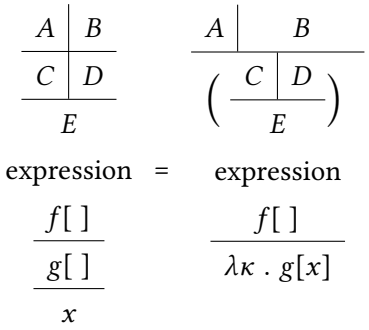
(9) For any syntactic category B,

$$\frac{\frac{S \mid S}{DP}}{\frac{\forall x\ .\ [\,]}{x}}\ \text{everyone} \xRightarrow{\text{LIFT}} \frac{\frac{\frac{B \mid B}{S \mid S}}{DP}}{\frac{[\,]}{\frac{\forall x\ .\ [\,]}{x}}}\ \text{everyone}$$

4. In derivations, I use a gray-colored font to mark the sections of each computation that have not changed since the previous step.

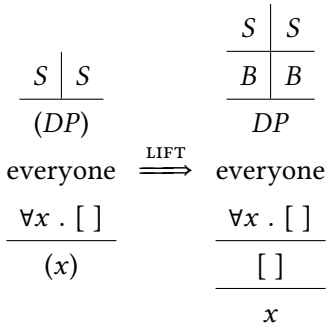
Applying LIFT to a two-level tower of category $A /\!\!/ (C \setminus\!\!\setminus B)$ generates a three-level tower of type $D /\!\!/ (A /\!\!/ (C \setminus\!\!\setminus B) \setminus\!\!\setminus D)$ for any syntactic category A, B, C, D. It may be worthwhile to note here that a multi-level tower can be thought of as a tower whose base level is itself a tower, rather than an atomic category, as represented in (10):

(10)



When dealing with multi-layered towers, we may also want to insert empty scopes *between* layers of the tower, raising the scopal effects of an expression up to a higher level. Doing so amounts to applying LIFT to an internal category, rather than to the entire expression:

(11) For any category B,



Observe that after applying internal LIFT, the scopal effects of the expression are located on the top level, while after applying external LIFT the scopal effects of the expression are on the middle level.

2.4.2 LOWER

Using LIFT, we are able to compose non-scope-taking and scope-taking items together in (8). However, once composed, the final expression in (8) is of category $S \mathbin{/\!\!/} (S \mathbin{\backslash\!\!\backslash} S)$ with semantic type $\langle\langle t, t\rangle t\rangle$ instead of a plain S of semantic type t. Correspondingly, the value of the final expression in (11) still expects to be fed a continuation of category $S \mathbin{\backslash\!\!\backslash} S$ (semantic type $\langle t, t\rangle$). In order to derive a final semantic value for a complete sentence, then, B&S define a type-shifter LOWER:

Definition 4: LOWER type-shifter.

$$\begin{array}{ccccc} \dfrac{A \mid S}{S} & & A \mathbin{/\!\!/} (S \mathbin{\backslash\!\!\backslash} S) & \xRightarrow{\text{LOWER}} & A \\ \text{expression} & = & \text{expression} & & \text{expression} \\ \dfrac{f[\,]}{x} & & \lambda\kappa\,.\,f(\kappa x) & & f[x] \end{array}$$

Semantically, the LOWER type-shifter feeds the identity function $\lambda y\,.\,y$ to the expression, 'plugging' the gap that existed there. Unlike beta reduction, plugging a hole can result in variable capture. For example, if $g[\,] = \lambda x\,.\,(x[\,])y$, then $g[x] = \lambda x\,.\,(x[x])y = \lambda x\,.\,(xx)y$, where the value x that is plugged in is bound by the lambda (Barker and Shan 2014:16).

Crucially, LOWER-ing an expression has the function of "clos[ing] off the scope domain of [a] quantifier" (Barker and Shan 2014:20). When we execute a LOWER, we discharge the scopal powers of the towers: f applies over no more and no less than the material that it is LOWER-ed onto; after LOWER, its gap has been filled and it is unable to take scope over any more material. This intuition will come in handy later.

Unlike LIFT, which can be used on expressions of any syntactic category, B&S specify that LOWER only applies to expressions that take in continuations of type $S \mathbin{\backslash\!\!\backslash} S$, meaning that only sentences with sentence-shaped gaps can be LOWER-ed. When the top-right and bottom categories are both S, we know that the material on the bottom layer of the tower has the same category as the gap [ ] that it intends to fill. We can then fill the gap with

the bottom-layer content, 'plugging' the hole.

Derivations are considered complete when the expression has been LOWER-ed to a single-layer tower; once an expression is completely lowered and all gaps [ ] have been filled, the expression no longer expects to scope over additional material to its right.

2.5 Multi-layered towers and inverse scope

LIFT and LOWER give us the ability to generate in-situ analyses of both the surface scope and inverse scope readings of sentences with multiple quantifiers, such as (12).

(12) Someone likes everyone.

a. A specific person likes everyone. surface scope

b. For all people, there is a person who likes them. inverse scope

To derive the surface scope reading (12a), we first LIFT the non-scopal expression *likes* so that it forms a two-level tower that can compose with the inherently scope-taking expressions *someone* and *everyone*. Then, we compose the towers and LOWER the result. This results in the reading where a single person likes all people, shown in (13).

(13)

$\frac{S \mid S}{DP}$	$\frac{S \mid S}{(DP \backslash S) / DP}$	$\frac{S \mid S}{DP}$		$\frac{S \mid S}{S}$
someone	likes	everyone	=	someone likes everyone
$\frac{\exists x \,.\, [\,]}{x}$	$\frac{[\,]}{\textbf{likes}}$	$\frac{\forall y \,.\, [\,]}{y}$		$\frac{\exists x \,.\, \forall y \,.\, [\,]}{\textbf{likes}\ y\ x}$

$$\xRightarrow{\text{LOWER}} \begin{array}{c} S \\ \text{someone likes everyone} \\ \exists x \,.\, \forall y \,.\, \textbf{likes}\ y\ x \end{array}$$

Deriving the inverse scope reading (12b) requires something different: we first LIFT *everyone* internally so that the universal quantifier ends up on the third level of the tower.

Doing so amounts to giving the universal quantifier scope over a continuation of type $(S \mathbin{/\!\!/} (DP \mathbin{\backslash\!\!\backslash} S)) \mathbin{\backslash\!\!\backslash} S$ instead of $DP \mathbin{\backslash\!\!\backslash} S$ — in other words, *everyone* takes scope over a continuation that itself takes scope over a continuation. In the semantic tower, we can see this represented by the fact that the universal quantifier will be evaluated first and take widest scope, since it is located on the top layer of the eventual tower. Its scope, reading left to right and top to bottom, includes the existential.

Applying external LIFT to *someone* and *everyone* to match the three-layered tower for *everyone* and combining the expressions before LOWER-ing the tower twice yields the reading where for each person, someone likes them, as shown in (14).

(14)

$\frac{\frac{S \mid S}{S \mid S}}{DP}$	$\frac{S \mid S}{S \mid S}$	$\frac{\frac{S \mid S}{S \mid S}}{DP}$		$\frac{\frac{S \mid S}{S \mid S}}{S}$
someone	likes	everyone	=	someone likes everyone
$\frac{[\,]}{\exists x\,.\,[\,]}$	$\frac{[\,]}{[\,]}$	$\frac{\forall y\,.\,[\,]}{[\,]}$		$\frac{\forall y\,.\,[\,]}{\exists x\,.\,[\,]}$
x	**likes**	y		**likes** $y\,x$

$$\xRightarrow{\text{LOWER}} \text{someone likes everyone} \quad \frac{\frac{S \mid S}{S}}{\frac{\forall y\,.\,[\,]}{\exists x\,.\,\textbf{likes}\ y\,x}}$$

$$\xRightarrow{\text{LOWER}} \text{someone likes everyone} \quad \frac{S}{\forall y\,.\,\exists x\,.\,\textbf{likes}\ y\,x}$$

Having the power to arbitrarily LIFT expressions without constraints on when to apply corresponding LOWERS overgenerates readings, as we will see in Section 3.1.

2.6 Gaps, movement and FRONT

Evaluation of sentences so far has occurred from left to right in linear order. However, in instances of topicalization such as (15), expressions appear to contribute semantic content to a position, represented by ___, different from the position in which they are pronounced.

(15) Everyone, John likes ___.
⇒ *John likes everyone*

In the continuation framework, B&S account for non-linear evaluation of expressions using semantic reconstruction. While there is no 'movement' — i.e. no structure corresponding to a difference between logical form (LF) and phonetic form (PF) — in the continuation framework, B&S propose a silent expression called a 'gap' and a FRONT type-shifter such that the local semantic value of the moved item is eventually evaluated as if it were in the gap position.

2.6.1 Gaps

In general, we can think about the structure of a clause that undergoes movement as a full clause over which we have abstracted a constituent. If such a description seems to resemble that of a continuation, that's because clauses with gaps are exactly of the category $A \backslash\!\backslash S$, where a gap of category $A \mathbin{/\!/} A$ fills in for the extracted expression. The semantic value of a gap is the identity function. Incidentally, B&S propose the identity function as the semantic value of a pronoun, as we will see in Chapter 4. Let's put the gap into action by trying to derive the gapped clause in (15b).

(16)
$$\frac{\frac{DP \backslash\!\!\backslash S \mid DP \backslash\!\!\backslash S}{DP}\ \text{John}\ [\,]}{\textbf{John}} \quad \frac{\frac{DP \backslash\!\!\backslash S \mid DP \backslash\!\!\backslash S}{(DP \backslash S)/DP}\ \text{likes}\ [\,]}{\textbf{likes}} \quad \frac{\frac{DP \backslash\!\!\backslash S \mid S}{DP}\ \underline{\quad}\ \lambda x\,.\,[\,]}{x} \quad = \quad \frac{\frac{DP \backslash\!\!\backslash S \mid S}{S}\ \text{John likes}\ \underline{\quad}\ \lambda x\,.\,[\,]}{\textbf{likes}\ x\ \textbf{John}}$$

$$\xRightarrow{\text{LOWER}} \quad \begin{array}{c} DP \backslash\!\!\backslash S \\ \text{John likes}\ \underline{\quad} \\ \lambda x\,.\,\textbf{likes}\ x\ \textbf{John} \end{array}$$

We first generate a gap of the type $(DP \backslash\!\!\backslash S) \mathbin{/\!\!/} (DP \backslash\!\!\backslash S)$ with semantic value $\lambda\kappa\,.\,\lambda x\,.\,\kappa x$. We then LIFT both *John* and *likes*, targetting the category $DP \backslash\!\!\backslash S$, and combine the towers. Since the top-right and bottom categories of the resulting tower are compatible, LOWER-ing the tower forms an expression of type $DP \backslash\!\!\backslash S$ — just the type of a continuation.

2.6.2 FRONT

We now turn to integrating an ex-situ expression with a gapped clause. B&S present the FRONT type-shifter for this purpose. FRONT is applied to an expression that is 'moved' out of its canonical position — in (16b), *everyone*. We mark the fact that an expression is fronted using a subscript F on the top-left syntactic category of the expression. Marking an expression as FRONT-ed does not affect its category or type until we type-shift the syntactic categories as follows:

Definition 5: FRONT type-shifter.

$$\frac{\frac{A_F \mid B}{C}\ \text{expression}\ [\,]}{x} \quad = \quad A \mathbin{/\!\!/} (B \backslash\!\!\backslash C)\ \text{expression}\ \lambda\kappa\,.\,f[\kappa x] \quad \xRightarrow{\text{FRONT}} \quad A/(B \backslash\!\!\backslash C)\ \text{expression}\ \lambda\kappa\,.\,f[\kappa x]$$

Notice that no change is made to the semantics when an expression is FRONT-ed. The change in syntactic category simply represents a difference in *where* an expression expects

to take an argument; the FRONT-ed version of an expression expects its argument to the right as opposed to surrounding it, which corresponds to the fact that it has been extracted to the left out of its expected context.

We can use FRONT to combine *everyone* with the gapped clause derived in (16):

(17)

$\frac{S_F \mid S}{DP}$	$DP \setminus\!\!\setminus S$	$\xRightarrow{\text{FRONT}}$	$S/(DP \setminus\!\!\setminus S)$	$DP \setminus\!\!\setminus S$
everyone	John likes ___		everyone	John likes ___
$\frac{\forall y\ .\ [\ ]}{y}$	$\lambda x\ .\ \mathbf{likes}\ x\ \mathbf{John}$		$\lambda\kappa\ .\ \forall y\ .\ \kappa[y]$	$\lambda x\ .\ \mathbf{likes}\ x\ \mathbf{John}$

$$= \begin{array}{c} S \\ \text{everyone, John likes } \_\_\_ \\ (\lambda\kappa\ .\ \forall y\ .\ \kappa[y])(\lambda x\ .\ \mathbf{likes}\ x\ \mathbf{John}) \end{array}$$

$$= \begin{array}{c} S \\ \text{everyone, John likes } \_\_\_ \\ \forall y\ .\ (\lambda x\ .\ \mathbf{likes}\ x\ \mathbf{John}))(y) \end{array}$$

$$= \begin{array}{c} S \\ \text{everyone, John likes } \_\_\_ \\ \forall y\ .\ \mathbf{likes}\ y\ \mathbf{John} \end{array}$$

In (17), we combine *everyone* with a fully-formed gapped clause and apply functional application to derive the expected reading. Notably, while the use of FRONT does not change the semantics of the fronted expression, it does change the order in which we evaluate the sentence. Instead of evaluating *everyone* first, as we would if we were deriving the 'base' sentence *John likes everyone*, we instead first evaluate the gap. The FRONT type-shifter builds in delayed evaluation of the fronted phrase by requiring the fronted phrase to combine with the meaning of a gapped clause, which is only possible after the gapped clause itself is fully evaluated.

2.7 The value of a continuation-based framework

The continuation framework makes predictions about phenomena affecting evaluation order and quantifier scope and binding. Specifically, Barker and Shan (2014) provide analyses of crossover (see Chapter 4.1), superiority, negative polarity licensing and donkey anaphora as claims to the utility of their fragment.

Moreover, B&S's continuation framework should also be located within a broader ecosystem of work applying computer science concepts of side effects and monads to linguistics. B&S's continuation framework conceives of scope-taking as a side effect that an expression carries on a separate layer from its regular semantic value, and integrates with monadic frameworks that extend to other linguistic phenomena such as exceptional scope (Charlow 2014), conventional implicature (Giorgolo and Asudeh 2012), focus, presupposition, intension and interrogatives (Shan 2001).

CHAPTER 3
LONG-DISTANCE MOVEMENT

3.1 The clause-bound scope of quantifiers

For many speakers of English[5], the scope of universal quantifier phrases is clause-bound — quantifier phrases are unable to take scope outside of their minimal tensed clause (Cecchetto 2004; Johnson 2000; Szabolcsi 1997, among others). For instance, the embedded universal quantifier *every book* is unable to take scope over *a different student* out of the finite clause in (18) (marked with angled brackets). In contrast, the non-finite clause in (19) does not bound the quantifier; it is thus able to take sentential scope.

(18) A (different) student said ⟨that I had read every book⟩. (Johnson 2000:196)

a. *For every book, a (different) student said that I had read it.

b. A student (different from the one we were talking about) said that for every book, I read it.

(19) A (different) student wanted to read every book.

a. A specific student (who is different from the one we were previously talking about) wanted to read each book.

b. For each book, a (different) student wanted to read it.

Although specific constraints on the boundaries of quantifier scope are debated (Farkas and Giannakidou 1996), the broad consensus remains that quantifier phrases, and in particular universal quantifiers, are clause-bound. As a framework that aims to account for

5. Some speakers allow quantifiers to take wide scope outside of clause boundaries; see Wurmbrand 2018.

the scopal power of quantifiers, then, it seems vital that the continuation framework be able to account for such a systematic constraint on quantifier scope. However, the framework presented by B&S fails to do so. As it stands, the continuation framework does not block universal quantifiers from taking scope outside the embedded clauses. For instance, the sentence in (20) should only have the reading (20a).

(20) Someone said ⟨everyone likes John⟩.

a. There is a specific person who said that everyone likes John.

b. #For each person, there is someone who said that that person likes John.

However, the derivation in (21) shows that the reading where the universal takes wide scope is predicted to be available by B&S's grammar, contrary to fact.

(21)

$\frac{\frac{S \mid S}{S \mid S}}{DP}$	$\frac{S \mid S}{S \mid S}$	$\frac{\frac{S \mid S}{S \mid S}}{DP}$	$\frac{S \mid S}{S \mid S}$	$\frac{\frac{S \mid S}{S \mid S}}{DP}$
$(DP \setminus S)/S$	DP	$(DP \setminus S)/DP$	DP	
someone	said	everyone	likes	John
$\frac{[\,]}{\exists x \,.\, [\,]}$	$\frac{[\,]}{[\,]}$	$\frac{\forall y \,.\, [\,]}{[\,]}$	$\frac{[\,]}{[\,]}$	$\frac{[\,]}{[\,]}$
x	**said**	y	**likes**	**John**

$$= \text{someone said everyone likes John} \quad \frac{\frac{\forall y \,.\, [\,]}{\exists x \,.\, [\,]}}{\mathbf{said}\ (\mathbf{likes}\ \mathbf{John}\ y)\ x} \quad \frac{\frac{S \mid S}{S \mid S}}{S} \quad \frac{S \mid S}{S}$$

$$\xRightarrow{\text{LOWER}} \text{someone said everyone likes John} \qquad \xRightarrow{\text{LOWER}} \text{someone said everyone likes John}$$

$$\frac{\forall y \,.\, [\,]}{\exists x \,.\, \mathbf{said}\ (\mathbf{likes}\ \mathbf{John}\ y)\ x} \qquad \frac{S}{\forall y \,.\, \exists x \,.\, \mathbf{said}\ (\mathbf{likes}\ \mathbf{John}\ y)\ x}$$

In (21), I generate a set of three-level towers by applying internal LIFT on *everyone*, and external LIFT on the other lexical entries. Composing the lexical entries and LOWER-ing the resulting tower gives the unavailable reading (20b) where the universal takes wide scope. Thus, if left unconstrained, the framework overgenerates allowable readings.

In this section, following Barker and Shan (2008) and Charlow (2014), I set out a condition which I call the *Tensed Clause Condition* (TCC) as a way in the continuation framework to capture the facts about scope islands. I then highlight a quirk of clauses that are gapped that conflicts with the TCC and makes them unable to compose under the categorial grammar presented by B&S. I suggest a solution to this problem by drawing on evidence for successive cyclic movement and reconsidering the syntactic category of a LOWER-ed continuation.

3.2 Limiting the scope of quantifiers

3.2.1 The Tensed Clause Condition

In order to limit the scope of quantifiers, we can stipulate that we must fully evaluate an expression at the edge of every tensed clause, fully LOWER-ing it into a single-layered tower before proceeding with composition. This idea is not new; Charlow (2014:65) proposes that "evaluation is *obligatory* at sentence boundaries" in order to prevent a quantifier from indefinitely scoping over content to its right, while Barker and Shan (2008:27) propose using LOWER to close off the scope of quantifiers at the edge of a minimal clause. However, although B&S allude to the availability of LOWER to constrain scope islands, they do not operationalize this claim in the framework presented in Barker and Shan 2014.

As discussed in Chapter 2.4.2, executing LOWER exhausts the scope of all quantifier phrases on the second level of an expression, disallowing those quantifier phrases from further taking scope over additional material. Stipulating that an expression must be fully LOWER-ed occur at the edge of every tensed clause would thus discharge the full scope of all quantifier phrases, and capture the fact that universals cannot scope out of their minimal

tensed clause. Such a stipulation is equivalent to the stipulation in the LF paradigm that a constituent cannot QR out of a finite clause (Barker 2002; Heim and Kratzer 1998:215).

I will call this stipulation the *Tensed Clause Condition* (TCC). Below is a first pass at operationalizing what would occur at the edge of a tensed clause:

Definition 6: Tensed Clause Condition (first version). When we have derived a tensed clause, apply LOWER as many times as necessary such that all continuations are filled and denotations occur on the bottom level of the tower.

Let's try to derive the unattested reading (20b) again to show that when we forcibly LOWER the expression at the edge of the embedded clause, the universal is unable to scope out of the embedded clause. We begin with the same type-shifted denotations for the lexical items in (21), but forcibly LOWER at the boundary of the embedded clause in keeping with the TCC. Doing so blocks the universal quantifier from scoping out of the embedded clause, and thus correctly predicts the unavailability of (20b).

(22)

$\frac{S \mid S}{DP}$	$\frac{S \mid S}{(DP\backslash S)/S}$	($\frac{S \mid S}{DP}$	$\frac{S \mid S}{(DP\backslash S)/DP}$	$\frac{S \mid S}{DP}$)
someone	said	everyone	likes	John
$\frac{\exists x\,.\,[\,]}{x}$	$\frac{[\,]}{\textbf{said}}$	$\frac{\forall y\,.\,[\,]}{y}$	$\frac{[\,]}{\textbf{likes}}$	$\frac{[\,]}{\textbf{John}}$

$\frac{S \mid S}{DP}$	$\frac{S \mid S}{(DP\backslash S)/S}$	$\frac{S \mid S}{S}$
= someone	said	everyone likes John
$\frac{\exists x\,.\,[\,]}{x}$	$\frac{[\,]}{\textbf{said}}$	$\frac{\forall y\,.\,[\,]}{\textbf{likes John}\ y}$

$$\xRightarrow{\textsc{lower}} \quad \frac{S \mid S}{DP} \quad \frac{S \mid S}{(DP \backslash S)/S} \quad S$$

someone said everyone likes John

$\exists x \,.\, [\,]$ $[\,]$ $\forall y \,.\, \textbf{likes John}\ y$

x **said**

$$\xRightarrow{\textsc{lift}} \quad \frac{S \mid S}{DP} \quad \frac{S \mid S}{(DP \backslash S)/S} \quad \frac{S \mid S}{S}$$

someone said everyone likes John

$\exists x \,.\, [\,]$ $[\,]$ $[\,]$

x **said** $\forall y \,.\, \textbf{likes John}\ y$

$$\frac{S \mid S}{S}$$

$$= \text{someone said everyone likes John} \quad \frac{\exists x \,.\, [\,]}{\forall y \,.\, \textbf{likes John}\ y}$$

$$\xRightarrow{\textsc{lower}} \quad S$$

someone sid everyone likes John

$\exists x \,.\, \forall y \,.\, \textbf{said}\ (\textbf{likes John}\ y)\ x$

In (22), the universal is first internally LIFT-ed as in (21) onto the top layer of the tower, where it ostensibly takes scope over the existential quantifier on the second level of the tower. However, before composing *everyone likes John* with *said*, we force the intermediate tower *everyone likes John* to comply with the tensed clause condition. Firstly, we apply LOWER to fully evaluate the expression, representing the resultant denotation in a single-level tower. Doing so exhausts the universal quantifier's scope — in tower notation, this is represented by placing the quantifier on the bottom layer of the tower. Composition then proceeds by LIFT-ing the trivial tower back into a two-layer tower, and gives the expected reading (20a) — where a single person claims that all people like John — rather than the blocked reading (20b), where it is possible for multiple people to attest that someone likes John.

3.3 Long-distance movement fails

I have proposed that the TCC captures the appropriate constraint on quantifier scope displacement across tensed clauses. Unsurprisingly, the TCC remains just as necessary when dealing with sentences that contain displaced expressions, as in (23):

(23) John, someone said ⟨everyone likes ___⟩.

(23) is an example of long-distance movement where an expression is extracted from an embedded tensed clause, moved across a higher clause, and pronounced in a derived position. In (23), the scope island formed by the embedded clause remains in effect — the sentence only has the reading where the universal quantifier takes narrow scope under the existential. However, the continuation framework cannot presently account for the clause-boundedness of the universal quantifier in (23).

Barker and Shan (2014) provision a perspective and matching syntactic formalism for viewing the gapped clause as a continuation, as we saw in Section 2.6.2. They also propose the FRONT type-shifter (Definition 5), which composes a fronted sequence with a LOWER-ed expression. While their theory does not explicitly deal with examples of long-distance movement, they show that short-distance movement can be resolved using gaps and FRONT. More precisely, fronted material can be composed with a LOWER-ed simple sentence, for example in (24).

(24)

	$DP \setminus\!\!\setminus S \mid DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S \mid DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S \mid S$
$S_F \mid S$	$S \mid S$	$S \mid S$	$S \mid S$
DP	DP	$(DP \backslash S)/DP$	DP
John	everyone	likes	___
[ ]	[ ]	[ ]	λz . [ ]
John	$\forall y$. [ ]	[ ]	[ ]
	y	**likes**	z

$$
\begin{array}{rcc}
 & & \dfrac{DP \setminus\!\!\setminus S \mid S}{\dfrac{S \mid S}{S}} \\
 & \dfrac{S_F \mid S}{DP} & \\
= & \text{John} & \text{everyone likes } \_\_\_ \\
 & [\,] & \lambda z\,.\,[\,] \\
 & \textbf{John} & \forall y\,.\,[\,] \\
 & & \textbf{likes}\ z\ y
\end{array}
$$

$$
\begin{array}{rcc}
 & \dfrac{S_F \mid S}{DP} & DP \setminus\!\!\setminus S \\
\xRightarrow{\text{LOWER,LOWER}} & \text{John} & \text{everyone likes } \_\_\_ \\
 & [\,] & \lambda z\,.\,\forall y\,.\,\textbf{likes}\ z\ y \\
 & \textbf{John} &
\end{array}
$$

$$
\begin{array}{rcc}
 & S/(DP \setminus\!\!\setminus S) & DP \setminus\!\!\setminus S \\
\xRightarrow{\text{FRONT}} & \text{John} & \text{everyone likes } \_\_\_ \\
 & \lambda\kappa\,.\,\kappa\,\textbf{John} & \lambda z\,.\,\forall y\,.\,\textbf{likes}\ z\ y
\end{array}
$$

$$
\begin{array}{rc}
 & S \\
= & \text{John, everyone likes } \_\_\_ \\
 & \forall y\,.\,\textbf{likes John}\ y
\end{array}
$$

In a simple sentence such as (24), the only scope island is the matrix clause. LOWER thus occurs only once at the edge of the matrix clause, both to satisfy the tensed clause condition and to form a syntactic category that combines immediately with the fronted material.

Unfortunately, while B&S show that FRONT and gaps can account for local movement, it is not generally true that clauses containing gaps can combine immediately with their FRONT-ed expressions. Indeed, in a long-distance movement configuration like (23) — in other words, when an extracted expression's gap is embedded within multiple tensed clauses — the FRONT type-shifter is unable to smooth out how a gapped clause can compose in a derivation. This general configuration is not accounted for by B&S.

The problem arises as follows: in a long-distance movement configuration like (23), FRONT-ed material is extracted out of an embedded clause instead of the local clause. The embedded clause is itself a clausal argument to a functor — in the case of (23), the embedded clause is an argument to *said*. In this configuration, LOWER-ing occurs as expected at the

edge of the embedded clause in order to satisfy the TCC. However, unlike in the case of movement out of a local clause, the LOWER-ed material in an embedded clause does not immediately combine with a FRONT-ed expression. Instead, it must compose with a functor expecting a non-gapped argument via the regular mode of composition. Thus, a category mismatch emerges: while the functor expects an argument of syntactic category S, the embedded clause has syntactic category $DP \mathbin{\backslash\!\!\backslash} S$. I illustrate this mismatch in (25):

(25) John, someone said ⟨everyone likes ___⟩.

$S_F \mid S$	$S \mid S$		
DP	DP	$(DP \backslash S)/S$	$DP \mathbin{\backslash\!\!\backslash} S$
John	someone	said	everyone likes ___
[ ]	$\exists x$. [ ]	**said**	λz . $\forall y$. **likes** z y
John	x		

In (25), I generate a clause with a gap corresponding to the extracted object *John*, as in (24). I then LOWER the clause to comply with the TCC. However, whereas in (24) the lowered clause immediately combines with a FRONT-ed *wh*-phrase that expects a gapped clause to its right, in (25) the gapped clause combines with the lexical item *said*, which expects a non-gapped clause of category S. The category mismatch prevents us from proceeding with composition. The need to abide by the TCC and LOWER a gapped clause seems thus to be incompatible with the current mode of composition.

Of course, we could choose not to LOWER the embedded clause at the boundary of the tensed clause. However, doing so incorrectly predicts in a similar fashion to (21) that the universal can take wide scope out of its scope island. Thus, the TCC must hold in the continuation framework to limit the scope of universals.

3.4 A solution

So far, I have shown that the tensed clause condition must hold in the continuation framework to account for scope islands. I have also shown that without additional machinery, forcibly LOWER-ing expressions leads to the formation of an expression that is not well-defined in the continuation framework. In this section, I propose a solution that accounts for long-distance movement.

My solution comes in three parts. Firstly, I point out a notational quirk in the continuation framework and use it to show how to circumvent the category mismatch stemming from LOWER-ing a gapped clause. Then, drawing from contemporary syntactic theory, I propose that intermediate gaps exist to the left of gapped clauses which serve to propagate the scope of gaps. This allows for the derivation of structures involving extraction out of embedded clauses.

3.4.1 Adjacency and containment

Recall from Section 2.2 that the sole difference between an expression of syntactic category $A \backslash B$ and one of category $A \mathbin{\backslash\!\!\backslash} B$ is that of adjacency vs. containment — whereas the former takes in an argument to its left, the latter takes in an argument from within it. Semantically, both a continuation $A \mathbin{\backslash\!\!\backslash} B$ and a functor $A \backslash B$ have denotations of semantic type $\langle \alpha, \beta \rangle$, where α, β are the semantic types of A and B respectively.

When an expression is overtly moved from its continuation, its relationship with its continuation is no longer one of containment. For instance, in the sentence *A book, John sent [ ] to Mary*, the expression *a book* is no longer contained within its continuation; rather, it is to the left of its continuation. Thus, the continuation of an extracted expression is better represented by the category $A \backslash B$ if the expression is located to the left of its continuation, and B / A if it is located to the right, as illustrated in Figure 3.

Notably, this same intuition underlies B&S's FRONT type-shifter. By proposing that fronting an expression is concomitantly marked by a shift in type from the syntactic category

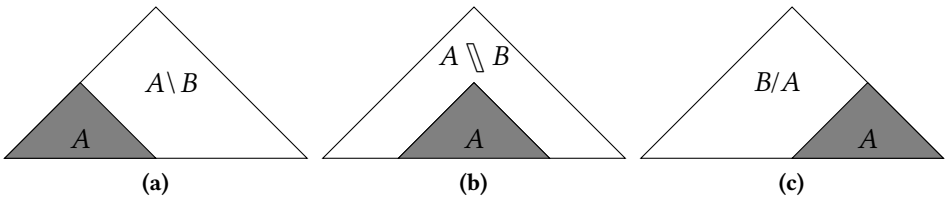


Figure 3: Schemata of relationship between functors and arguments

$A \mathbin{/\!\!/} (C \mathbin{\backslash\!\!\backslash} B)$ to $A/(C \mathbin{\backslash\!\!\backslash} B)$, B&S note as well that a fronted expression no longer expects to be surrounded by a continuation of category $C \mathbin{\backslash\!\!\backslash} B$; instead, it expects to take in the continuation to its right.

A corollary of this intuition is that we can now identify expressions of category $A \mathbin{\backslash\!\!\backslash} B$ that live on the bottom layer of a tower as being equivalently of category $A \backslash B$. Consider what occurs when a tower is lowered — the continuation, now LOWER-ed to the bottom layer, can no longer surround any other expressions; instead, it takes in arguments to its left or right. Just as with FRONT, the denotation of an expression does not change. With this intuition, let us update our definition of the TCC:

Definition 7: Tensed Clause Condition (second version). If the expression is a tensed clause, apply LOWER as many times as possible, until all quantifiers have exhausted their scope. If the syntactic category of the resulting expression is $A \mathbin{\backslash\!\!\backslash} B$, optionally type-shift the syntactic category to $A \backslash B$ with no change to the denotation.

3.4.2 An intermediate gap

Although we have made it possible for a fully LOWER-ed gapped clause to compose on the bottom layer, the gapped clause now expects to combine with an argument of category *DP*, which it is unable to do. I show this in (26); *everyone likes*, which is now of category $DP \backslash S$ instead of $DP \mathbin{\backslash\!\!\backslash} S$, expects a *DP* to its left. However, there is no overt *DP* to its left.

(26)

$\frac{S_F \mid S}{DP}$	$\frac{S \mid S}{DP}$	$(DP \backslash S)/S$	$DP \backslash S$
John	someone	said	everyone likes ___
[ ]	$\exists x$. [ ]	**said**	λz . $\forall y$**likes** z y
John	x		

This problem could be solved by positing the existence of an unpronounced *DP* in the position immediately to the left of the LOWER-ed clause. Such a *DP* would act as an argument to the gapped clause, making composition possible again.

Motivation for the existence of such an invisible particle does exist — evidence for successive cyclic movement suggests that an A'-extracted expression is moved not in one fell swoop to its final position, but instead first to intermediate positions corresponding to the edge of each tensed clause. For instance, *wh*-copy behavior in child language points to the pronunciation of the intermediate position of a moved *wh*-phrase:

(27) a. What do you think what Cookie Monster eats ___? (age 5;0)

b. Who do you think who Grover wants to hug ___? (age 4;9)

(Crain and Lillo-Martin 1999:238)

As another argument for successive cyclic movement, consider (28). While (28a) is grammatical, (28b) is ungrammatical due to Binding Principle A (Chomsky 1981), which requires that reflexives be bound locally. The fact that (28c) is grammatical then suggests that the pronoun is not interpreted in its surface position, since it cannot be bound there, nor in its base position (28b), where the intended binding is also ungrammatical. Instead, the grammaticality of (28c) suggests that there exists an intermediate position for the *wh*-phrase that linearly follows its binder, but occurs before the start of the embedded tensed clause *that Mary bought* ___; it is in this position that the reflexive in (28c) is bound.

(28) a. Mary$_j$ bought a picture of herself$_j$. (Richards, cited in Erlewine 2016)

b. *John$_i$ said [that Mary$_j$ bought a picture of himself$_i$].

c. [Which picture of himself$_i$] did John$_i$ say ___[that Mary$_j$ bought ___]?

Based on such evidence, it seems plausible then to posit that there might be an unpronounced intermediate trace corresponding to the position of the intermediate *wh*-phrase.

Having theoretically grounded my proposal for an intermediate gap, I now integrate an intermediate position gap into the continuation framework. In Chapter 2.6, I introduced the expression B&S call a 'gap', which has the syntactic category $A /\!\!/ A$ for some category A, and whose denotation is the identity function. I repurpose the same denotation and category of gap here, and propose that in addition to the gap located in the original position of the extracted expression, an additional intermediate trace exists that adjoins the edge of a tensed clause when movement occurs.

Let us now consider how adding an intermediate gap and implementing type-shifting upon LOWER-ing makes our long-distance movement configuration composable, as illustrated in (29):

(29)

$DP \setminus\!\!\setminus S$ | $DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S$ | $DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S$ | S	S | S
S | S	S | S	S | S	S | S
DP	$(DP \setminus S)/S$	DP	$DP \setminus S$
someone	said	___	everyone likes ___
[ ]	[ ]	λx . [ ]	[ ]
$\exists w$. [ ]	[ ]	[ ]	[ ]
w	**said**	x	λz . $\forall y$. **likes** z y

$$\begin{array}{c|c} DP \setminus\!\!\setminus S & DP \setminus\!\!\setminus S \\ \hline S & S \\ \hline \multicolumn{2}{c}{DP} \end{array} \quad \begin{array}{c|c} DP \setminus\!\!\setminus S & DP \setminus\!\!\setminus S \\ \hline S & S \\ \hline \multicolumn{2}{c}{(DP \setminus S)/S} \end{array} \quad \begin{array}{c|c} DP \setminus\!\!\setminus S & S \\ \hline S & S \\ \hline \multicolumn{2}{c}{S} \end{array}$$

$$= \begin{array}{c} \text{someone} \\ [\,] \\ \hline \exists w \,.\, [\,] \\ \hline w \end{array} \quad \begin{array}{c} \text{said} \\ [\,] \\ \hline [\,] \\ \hline \textbf{said} \end{array} \quad \begin{array}{c} \_\_\_ \text{ everyone likes } \_\_\_ \\ \lambda x \,.\, [\,] \\ \hline [\,] \\ \hline \forall y \,.\, \textbf{likes}\ x\ y \end{array}$$

$$\begin{array}{c|c} DP \setminus\!\!\setminus S & S \\ \hline S & S \\ \hline \multicolumn{2}{c}{S} \end{array}$$

$$= \text{someone said } \_\_\_ \text{ everyone likes } \_\_\_$$

$$\begin{array}{c} \lambda x \,.\, [\,] \\ \hline \exists w \,.\, [\,] \\ \hline \textbf{said}\ (\forall y \,.\, \textbf{likes}\ x\ y)\ w \end{array}$$

$$\xRightarrow{\text{LOWER}\times 2} \begin{array}{c} DP \setminus\!\!\setminus S \\ \hline \text{someone said } \_\_\_ \text{ everyone likes } \_\_\_ \\ \hline \lambda x \,.\, \exists w \,.\, \textbf{said}\ (\forall y \,.\, \textbf{likes}\ x\ y)\ w \end{array}$$

I begin by generating towers for the local clause, intermediate trace and embedded clause. To the proposed trace, I assign the syntactic category $(DP \setminus\!\!\setminus S) /\!\!/ (DP \setminus\!\!\setminus S)$. This syntactic category corresponds to a two-level tower where the lambda binder is initially on the second level of the tower. Noting that it will combine with the existential quantifier later, I choose to apply internal LIFT to the intermediate trace so that it will be located instead on the third layer from the bottom, and be able to take scope over the entire expression. This move in the semantic tower is mirrored in the way that the syntactic category $DP \setminus\!\!\setminus S$ is lifted to the third layer of the syntactic tower; conceptually, such a move is akin to implying that the gap will take scope over the existential quantifier. Without lifting the level of the lambda binder, we would trap the lambda operator within the scope of the existential, causing it to be unbindable later. I then apply external LIFT to the other expressions, including the gapped clause derived earlier in (24), so that they will compose smoothly. After composing the towers and LOWER-ing the result, I arrive at an expression of category $DP \setminus\!\!\setminus S$. Note that I do not type-shift the final expression from $DP \setminus\!\!\setminus S$ to

$DP \backslash S$, in preparation for combining the evaluated expression with a FRONT-ed expression. Doing so is acceptable, since type-shifters apply freely and optionally. The final expression composes well with the topicalized expression *John* to derive the expected reading:

(30)

$$\frac{S_F \mid S}{DP} \qquad DP \backslash S$$

John — someone said ___ everyone likes ___

[ ] — $\lambda x \,.\, \exists w \,.\, \textbf{said}\,(\forall y \,.\, \textbf{likes}\; x\; y)\; w$

John

$$\overset{\text{FRONT}}{\Longrightarrow} \quad \underset{\lambda \kappa \,.\, \kappa\, \textbf{John}}{S/(DP \mathbin{\backslash\!\!\backslash} S)} \qquad \underset{\lambda x \,.\, \exists w \,.\, \textbf{said}\,(\forall y \,.\, \textbf{likes}\; x\; y)\; w}{DP \mathbin{\backslash\!\!\backslash} S}$$

John — someone said ___ everyone likes ___

$$= \underset{\exists w \,.\, \textbf{said}\,(\forall y \,.\, \textbf{likes John}\; y)\; w}{S}$$

John, someone said ___ everyone likes ___

We have thus seen that the despite having to LOWER the embedded clause completely to fulfill the TCC, making two independently-attested enhancements to the continuation framework can allow long-distance movement configurations to compose. Through the addition of intermediate traces at the edges of tensed clause boundaries and a type-shifter that optionally turns a continuation into a predicate, I have shown that long-distance movement can be accounted for in the continuation framework.

3.5 Predictions

Adding an intermediate trace and proposing that lowered continuations behave equivalently to predicates has so far made it possible to account for long-distance movement in the continuation framework. In this section, I show that enhancing B&S's continuation framework also makes predictions about the availability of long-distance movement from arbitrarily deeply-embedded clauses, and accurately accounts for scope ambiguities that result from extracting quantifier phrases.

3.5.1 Embedding

In previous sections, we have seen how sentences like (31a) can be derived by LOWER-ing the gapped clause once and combining it with a FRONT-ed expression. I have modified the continuation framework such that (31b) can correctly be derived by forcibly LOWER-ing the expression at the edges of both tensed clauses. Using the newly-defined TCC, I also predict that expressions can be extracted from arbitrarily deeply-embedded positions in a sentence. Thus, (31c), and even constructions with deeper embeddings, are predicted to be acceptable.

(31) a. John, ⟨everyone likes ___⟩.

b. John, ⟨someone said ⟨everyone likes ___⟩⟩.

c. John, ⟨Mary thought ⟨someone said ⟨everyone likes ___⟩⟩⟩.

To show that (31c) is derived as expected under the enhanced continuation framework, I first take the result of (29) — the expression *someone said everyone likes* ___ with the syntactic category $DP \setminus\!\!\setminus S$ — and apply the optional type-shifter that turns a $DP \setminus\!\!\setminus S$ into $DP \setminus S$. I use this type-shifted expression as an argument in the matrix clause of (31c), and attach an intermediate trace of category $(DP \setminus\!\!\setminus S) /\!\!/ (DP \setminus\!\!\setminus S)$ to the left of the embedded clause. I show this configuration in (32):

(32)

$DP \setminus\!\!\setminus S \mid DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S \mid DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S \mid S$	
DP	$(DP \setminus S)/S$	DP	$DP \setminus S$
Mary	thought	___	sm. said ___ ev. likes ___
[ ]	[ ]	λz . [ ]	λx . $\exists w$. **said** ($\forall y$. **likes** x y) w
Mary	**thought**	z	

LIFT ⟹

$DP \setminus\!\!\setminus S \mid DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S \mid DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S \mid S$	$S \mid S$
DP	$(DP \setminus S)/S$	DP	$DP \setminus S$
Mary	thought	___	sm. said ___ ev. likes ___
[ ]	[ ]	λz . [ ]	[ ]
Mary	**thought**	z	λx . $\exists w$. **said** ($\forall y$. **likes** x y) w

$$\frac{DP \setminus\!\!\setminus S \mid S}{S}$$

$$= \text{Mary thought } \_\_\_ \text{ someone said } \_\_\_ \text{ everyone likes } \_\_\_$$

$$\frac{\lambda z \,.\, [\,]}{\exists w \,.\, \textbf{said}\,(\forall y \,.\, \textbf{likes}\; z\; y)\; w}$$

$$\overset{\text{LOWER}}{\Longrightarrow} \text{Mary thought } \_\_\_ \text{ someone said } \_\_\_ \text{ everyone likes } \_\_\_$$

$$DP \setminus\!\!\setminus S$$

$$\lambda z \,.\, \exists w \,.\, \textbf{said}\,(\forall y \,.\, \textbf{likes}\; z\; y)\; w$$

Just as in (29), combining the LOWER-ed gapped clause in (32) with an intermediate trace yields an expression that can ultimately be LOWER-ed into an expression of category $DP \setminus\!\!\setminus S$. This expression can then be combined with a FRONT-ed expression, as I show in (33):

(33)

$$\frac{\dfrac{S_F \mid S}{DP}}{\dfrac{\text{John}}{\dfrac{[\,]}{\textbf{John}}}} \qquad \begin{array}{c} DP \setminus\!\!\setminus S \\ \text{Mary thought } \_\_\_ \text{ someone said } \_\_\_ \text{ everyone likes } \_\_\_ \\ \lambda z \,.\, \exists w \,.\, \textbf{said}\,(\forall y \,.\, \textbf{likes}\; z\; y)\; w \end{array}$$

$$\overset{\text{FRONT}}{\Longrightarrow} \frac{\dfrac{S/(DP \setminus\!\!\setminus S)}{\text{John}}}{\dfrac{[\,]}{\textbf{John}}} \qquad \begin{array}{c} DP \setminus\!\!\setminus S \\ \text{Mary thought } \_\_\_ \text{ someone said } \_\_\_ \text{ everyone likes } \_\_\_ \\ \lambda z \,.\, \exists w \,.\, \textbf{said}\,(\forall y \,.\, \textbf{likes}\; z\; y)\; w \end{array}$$

$$S$$

$$= \text{John, Mary thought } \_\_\_ \text{ someone said } \_\_\_ \text{ everyone likes } \_\_\_$$

$$\exists w \,.\, \textbf{said}\,(\forall y \,.\, \textbf{likes}\;\textbf{John}\; y)\; w$$

Abstractly speaking, it is possible to embed a gapped clause at an arbitrarily deep level because each intermediate trace fills a deeper gap, generating a larger gapped clause with the same syntactic category $DP \setminus\!\!\setminus S$ as the gapped clause embedded inside it. Since the first gapped clause has syntactic category $DP \setminus\!\!\setminus S$ and all intermediate gapped clauses are also $DP \setminus\!\!\setminus S$, the final gapped clause is likewise of syntactic category $DP \setminus\!\!\setminus S$, and composes successfully with the extracted expression. Thus, the enhanced continuation framework predicts that long-distance movement is as available as local movement, i.e. we can move a DP from a clause that is deeply-embedded just as well as from a simple sentence.

3.5.2 Reconstructing the narrow scope reading

We have seen so far that scope islands constrain the scope that in-situ quantifiers can take. In this section, I account for the fact that extracting a quantifier phrase out of a scope island allows it to take wide scope over the entire sentence, but does not lose the narrow scope reading. (34) exemplifies this phenomenon — it has the interpretation in (34a), obtained by allowing *everyone* to take wide scope over the existential despite its base position being within a scope island; it also has the interpretation in (34b) such that the quantifier behaves as if it were interpreted inside the scope island.

(34) Everyone, someone thinks ⟨that Mary met ___⟩. (Bošković 2004:618)

a. For each person x, a (potentially different) person thinks that Mary met x.

b. There is a specific person who thinks that Mary met everyone.

The surface scope reading (34a) obtains straightforwardly from the procedure outlined in the previous section. First, I compose and LOWER the embedded clause *Mary met* ___, in (35). Here, I give the gap the usual syntactic category $(DP \setminus\!\!\setminus S) /\!\!/ (DP \setminus\!\!\setminus S)$.

(35)

$DP \setminus\!\!\setminus S \mid DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S \mid DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S \mid S$		$DP \setminus\!\!\setminus S \mid S$
DP	$(DP \setminus S)/DP$	DP		S
Mary	met	___	=	Mary met ___
[ ]	[ ]	λw . [ ]		λw . [ ]
Mary	**met**	w		**met** w **Mary**

$$\xRightarrow{\text{LOWER}}$$

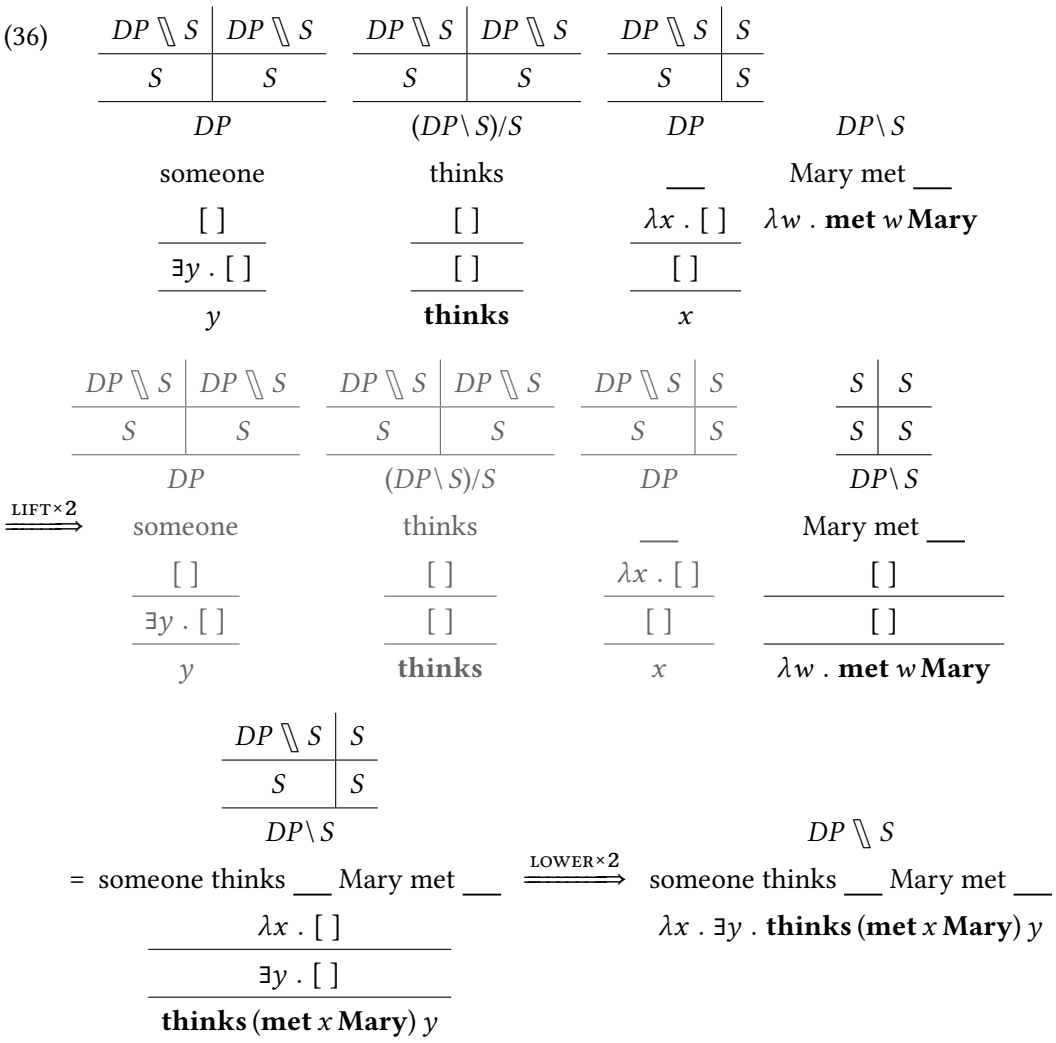
$DP \setminus S$

Mary met ___

λw . **met** w **Mary**

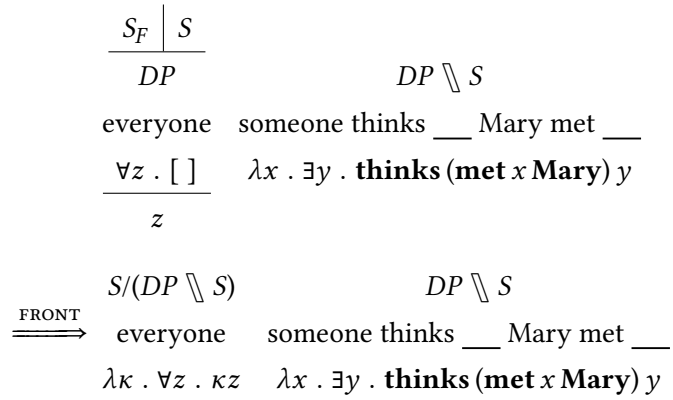
Then, I compose the embedded clause with the matrix clause, applying internal LIFT on the intermediate trace so that it scopes over the entire expression, and applying external LIFT on the other expressions to match. Composing the expressions and LOWER-ing yields an expression of category $DP \setminus\!\!\setminus S$, which I do not type-shift; I show this in (36).

(36)



Finally, I combine the result of (36) with the fronted expression *everyone*. The fronted expression takes scope over the gapped clause, and yields the surface scope reading where for all people, potentially different people think that Mary met them.

(37)



$$S$$
$$= \text{ everyone, someone thinks } \_\_\_ \text{ Mary met } \_\_\_$$
$$\forall z \,.\, \exists y \,.\, \textbf{thinks}\,(\textbf{met}\,z\,\textbf{Mary})\,y$$

We have seen how to obtain the surface scope reading (34a). More interesting is how we might be able to scope the universal underneath the existential to obtain the reading in (34b). Since the continuation framework is based on left-to-right evaluation, the universal quantifier *everyone* seems at first glance like it might be forced to take wide scope over the gapped clause.

A solution to this problem lies in the polymorphic nature of gaps. We can leverage polymorphism to build a higher-typed intermediate trace that allows the quantifier to semantically reconstruct both the item and its scope effects into its original position. Since the expression moved in the case of topicalization is a quantifier, I choose the category $(QP \setminus\!\!\setminus S) /\!\!/ (QP \setminus\!\!\setminus S)$ instead of $(DP \setminus\!\!\setminus S) /\!\!/ (DP \setminus\!\!\setminus S)$ for the trace, where $QP = \dfrac{S \mid S}{DP}$. The semantic value of the trace remains the identity function.

Having a quantifier-like object in the position of the intermediate trace allows the trace to scope over the gapped clause, but notably *under* any quantifiers to its left. This fact will be vital in obtaining the reading where *everyone* takes narrow scope.

To the embedded clause obtained in (35), I introduce an intermediate trace of category $(QP \setminus\!\!\setminus S) /\!\!/ (QP \setminus\!\!\setminus S)$ before combining the gapped clause with the matrix clause. Note that I LIFT the gapped clause twice here, so that it combines with the three-layered intermediate trace. I also apply external LIFT to the quantifier phrase *someone* so that the intermediate trace scopes over it.

(38)

$\dfrac{\dfrac{QP \setminus\!\!\setminus S \mid QP \setminus\!\!\setminus S}{S \mid S}}{DP}$	$\dfrac{\dfrac{QP \setminus\!\!\setminus S \mid QP \setminus\!\!\setminus S}{S \mid S}}{(DP \setminus S)/S}$	$\dfrac{\dfrac{QP \setminus\!\!\setminus S \mid S}{S \mid S}}{DP}$	$\dfrac{\dfrac{S \mid S}{S \mid S}}{DP \setminus S}$
someone	thinks	___	Mary met ___
$\dfrac{[\,]}{\exists y \,.\, [\,]}$	$\dfrac{[\,]}{[\,]}$	$\dfrac{\lambda P \,.\, [\,]}{P(\lambda z \,.\, [\,])}$	$\dfrac{[\,]}{[\,]}$
y	**thinks**	z	$\lambda w \,.\, \textbf{met}\,w\,\textbf{Mary}$

$$\dfrac{\dfrac{QP \setminus\!\!\setminus S \mid S}{S \mid S}}{S}$$

$$= \text{someone thinks } \_\_\_ \text{ Mary met } \_\_\_ \quad \Longrightarrow^{\textsc{lower}} \quad \text{someone thinks } \_\_\_ \text{ Mary met } \_\_\_$$

$$\dfrac{\dfrac{\lambda P \,.\, [\,]}{\exists y \,.\, P(\lambda z \,.\, [\,])}}{\textbf{thinks}\,(\textbf{met}\,z\,\textbf{Mary})\,y} \qquad\qquad \begin{array}{c} QP \setminus\!\!\setminus S \\ \lambda P \,.\, \exists y \,.\, P(\lambda z \,.\, \textbf{thinks}\,(\textbf{met}\,z\,\textbf{Mary})\,y) \end{array}$$

After LOWER, notice that we have an expression whose syntactic category is that of a sentence missing a quantifier, not just a sentence missing a *DP*. Secondly, note also that because combination on the upper levels occurs from left to right, the existential takes scope over the expected quantifier P — we can see this in the semantic value of the final tower. It is this relationship that allows the quantifier to semantically reconstruct into a position where it takes scope over the matrix clause but not additional material, despite linearly preceding the entire gapped clause. I combine the gapped clause with a fronted quantifier to illustrate such a reconstruction in (39). To the quantifier phrase, I apply external LIFT. FRONT-ing the LIFT-ed expression then produces an expression that wants a $QP \setminus\!\!\setminus S$ on its right — just the category of our gapped clause.

(39)

$$\dfrac{\dfrac{S \mid S}{DP}}{x} \qquad \begin{array}{c} QP \setminus\!\!\setminus S \\ \text{someone thinks } \_\_\_ \text{ Mary met } \_\_\_ \\ \lambda P \,.\, \exists y \,.\, P(\lambda z \,.\, \textbf{thinks}\,(\textbf{met}\,z\,\textbf{Mary})\,y) \end{array}$$

$$\begin{array}{c} \dfrac{\dfrac{\dfrac{S_F \mid S}{S \mid S}}{DP}}{x} \\ \Longrightarrow^{\textsc{lift}} \end{array} \qquad \begin{array}{c} QP \setminus\!\!\setminus S \\ \text{someone thinks } \_\_\_ \text{ Mary met } \_\_\_ \\ \lambda P \,.\, \exists y \,.\, P(\lambda z \,.\, \textbf{thinks}\,(\textbf{met}\,z\,\textbf{Mary})\,y) \end{array}$$

$$
\begin{array}{rcc}
 & \dfrac{S_F \mid S}{QP} & QP \setminus\!\!\setminus S \\
= & \text{everyone} & \text{someone thinks \_\_\_ Mary met \_\_\_} \\
 & \dfrac{[\,]}{\lambda\kappa\,.\,\forall x\,.\,\kappa x} & \lambda P\,.\,\exists y\,.\,P(\lambda z\,.\,\textbf{thinks}\,(\textbf{met}\,z\,\textbf{Mary})\,y)
\end{array}
$$

$$
\begin{array}{rcc}
 & S/(QP \setminus\!\!\setminus S) & QP \setminus\!\!\setminus S \\
\overset{\text{FRONT}}{\Longrightarrow} & \text{everyone} & \text{someone thinks \_\_\_ Mary met \_\_\_} \\
 & \lambda\gamma\,.\,\gamma(\lambda\kappa\,.\,\forall x\,.\,\kappa x) & \lambda P\,.\,\exists y\,.\,P(\lambda z\,.\,\textbf{thinks}\,(\textbf{met}\,z\,\textbf{Mary})\,y)
\end{array}
$$

$$
\begin{array}{rc}
 & S \\
= & \text{everyone, someone thinks \_\_\_ Mary met \_\_\_} \\
 & (\lambda P\,.\,\exists y\,.\,P(\lambda z\,.\,\textbf{thinks}\,(\textbf{met}\,z\,\textbf{Mary})\,y))(\lambda\kappa\,.\,\forall x\,.\,\kappa x)
\end{array}
$$

$$
\begin{array}{rc}
 & S \\
= & \text{everyone, someone thinks \_\_\_ Mary met \_\_\_} \\
 & \exists y\,.\,(\lambda\kappa\,.\,\forall x\,.\,\kappa x)(\lambda z\,.\,\textbf{thinks}\,(\textbf{met}\,z\,\textbf{Mary})\,y)
\end{array}
$$

$$
\begin{array}{rc}
 & S \\
= & \text{everyone, someone thinks \_\_\_ Mary met \_\_\_} \\
 & \exists y\,.\,\forall x\,.\,(\lambda z\,.\,\textbf{thinks}\,(\textbf{met}\,z\,\textbf{Mary})\,y)x
\end{array}
$$

$$
\begin{array}{rc}
 & S \\
= & \text{everyone, someone thinks \_\_\_ Mary met \_\_\_} \\
 & \exists y\,.\,\forall x\,.\,\textbf{thinks}\,(\textbf{met}\,x\,\textbf{Mary})\,y
\end{array}
$$

Composing the two towers and going through beta reduction leads to the desired reading, with the universal taking narrow scope over just the gapped clause.

3.6 Summary

In this chapter, motivated by well-known restrictions on quantifier scope-taking, I introduced the Tensed Clause Condition, which enforces LOWER-ing expressions at the edge of every tensed clause. I showed how this requirement made composing clauses involving long-distance movement problematic in B&S's continuation framework. I treated the problem by proposing an intermediate trace and treating fully LOWER-ed continuations equivalently to unsaturated predicates. Finally, I showed how so enhancing the continua-

tion framework makes desirable predictions about topicalization out of deeply-embedded clauses and achieving the narrow scope reading from clauses with extracted quantifiers.

CHAPTER 4
BINDING AND CROSSOVER

I argued in the previous chapter that an enhanced account of the continuation framework can account for long-distance movement. In this chapter, I discuss the capacity for variable binding in the continuation framework. I consider the $\triangleright$ operator that B&S introduce for pronouns and how it interacts with movement. After introducing how to create an expression with pronouns and how to bind pronouns, I go on to show how an additional type-shifter must be introduced in order to account for binding pronouns across clause boundaries. I then attempt to integrate B&S's theory of pronominal binding into the account of movement I proposed in Chapter 3. I locate two issues related to crossover configurations, show that they pose serious threats to the predictive power of the continuation framework, and finally suggest that features fundamental to the continuation framework make these issues difficult to solve.

4.1 Binding variables in the continuation framework

4.1.1 Adding a new syntactic category

B&S propose a new complex syntactic category $A \triangleright B$ that represents an expression of type B except for the fact that it contains a pronoun of type A that needs to be bound. The semantic type of an expression of syntactic category $A \triangleright B$ is $\langle \alpha, \beta \rangle$, where α, β are the semantic types of A and B respectively.

With this new category, B&S define pronouns as expressions that "function locally as a DP, take scope over [a propositional expression], and turn that [expression] into an open proposition" (Barker and Shan 2014:25). This view follows Dowty (2007), who suggests that anaphors take scope over the expressions containing them. B&S's definition in tower notation for pronouns is given below:

Definition 8: Pronoun. For some syntactic category A:

$$\frac{DP \triangleright A \mid A}{DP}$$
$$\text{pronoun}$$
$$\frac{\lambda x\ .\ [\ ]}{x}$$

On B&S's theory, sentences with free pronouns are of a different syntactic category from sentences without pronouns, corresponding to differently-typed semantic values. A sentence like *John cried* represents a full expression that can immediately be evaluated, as shown in (40). In contrast, expressions with free pronouns, such as *he cried* in (41), contain an unbound lambda operator and represent open propositions that "[do] not express a complete thought until the value of the pronoun has been specified, whether by binding or by the pragmatic context" (Barker and Shan 2014:25).

(40)

DP	$DP \backslash S$		S
John	cried	=	John cried
John	**cried**		**cried John**

(41)

$$\frac{DP \triangleright S \mid S}{DP} \quad \frac{S \mid S}{DP \backslash S} \quad = \quad \frac{DP \triangleright S \mid S}{S}$$

he	cried	=	he cried
$\frac{\lambda x\ .\ [\ ]}{x}$	$\frac{[\ ]}{\textbf{cried}}$		$\frac{\lambda x\ .\ [\ ]}{\textbf{cried}\ x}$

$$\xRightarrow{\text{LOWER}} \begin{array}{c} DP \triangleright S \\ \text{he cried} \\ \lambda x\ .\ \textbf{cried}\ x \end{array}$$

In (41), a pronoun scoping over a syntactic category S is constructed. It combines with an intransitive verb *cried* to form a sentence that contains an unbound pronoun. Applying LOWER to the derivation yields an expression with the syntactic category $DP \triangleright S$, or a sentence that contains a pronoun; this category corresponds to a semantic type of $\langle e, t \rangle$. In comparison, (40) is of semantic type t.

In order to construct a bound reading of a pronoun, B&S propose a type-shifter called BIND that "enables an arbitrary DP to bind a downstream pronoun" (Barker and Shan 2014:26):

Definition 9: BIND type-shifter. For some syntactic categories A and B:

$$\begin{array}{ccccc} \dfrac{A \mid B}{DP} & & \dfrac{A \mid DP \triangleright B}{DP} & & A /\!\!/ (DP \backslash\!\!\backslash (DP \triangleright B)) \\ \text{expression} & \overset{\text{BIND}}{\Longrightarrow} & \text{expression} & = & \text{expression} \\ \dfrac{f[\,]}{x} & & \dfrac{f([\,]x)}{x} & & \lambda\kappa\,.\, f([\kappa x]x) \end{array}$$

Syntactically, a BIND-shifted expression expects as input an expression of its original category B except with an unbound pronoun. Semantically, applying BIND applies an additional copy of x to the function that will eventually plug the hole in f.

(42) illustrates how we can put BIND to use. Here, following B&S, I let the semantics of the possessive be vacuous and *mother* be a relation from individuals to their mothers. First, I LIFT *John* into a two-level tower. Then, I apply BIND to it so that it expects to bind a DP to its right. I apply LIFT on *mother* and *loves*, choosing the type $DP \triangleright S$ as the target category, so that the pronoun propagates upstream on the upper level of the towers.

(42) John$_i$'s mother loves him$_i$.

$$\begin{array}{cccccc} & \dfrac{S \mid S}{DP} & \dfrac{DP \triangleright S \mid DP \triangleright S}{DP \backslash DP} & \dfrac{DP \triangleright S \mid DP \triangleright S}{(DP \backslash S)/DP} & \dfrac{DP \triangleright S \mid S}{DP} \\ & \text{John's} & \text{mother} & \text{loves} & \text{him} \\ & [\,] & [\,] & [\,] & \lambda x\,.\,[\,] \\ & \textbf{John} & \textbf{mother} & \textbf{loves} & x \\ \overset{\text{BIND}}{\Longrightarrow} & \dfrac{S \mid DP \triangleright S}{DP} & \dfrac{DP \triangleright S \mid DP \triangleright S}{DP \backslash DP} & \dfrac{DP \triangleright S \mid DP \triangleright S}{(DP \backslash S)/DP} & \dfrac{DP \triangleright S \mid S}{DP} \\ & \text{John's} & \text{mother} & \text{loves} & \text{him} \\ & [\,]\textbf{John} & [\,] & [\,] & \lambda x\,.\,[\,] \\ & \textbf{John} & \textbf{mother} & \textbf{loves} & x \end{array}$$

$$\frac{S \mid DP \triangleright S}{DP} \qquad \frac{DP \triangleright S \mid S}{DP \backslash S}$$

$$= \frac{\text{John's mother}}{\frac{[\,]\ \textbf{John}}{\textbf{mother John}}} \qquad \frac{\text{loves him}}{\frac{\lambda x\ .\ [\,]}{\textbf{loves}\ x}}$$

$$\frac{S \mid S}{S}$$

$$= \frac{\text{John's mother loves him}}{\frac{(\lambda x\ .\ [\,])\ \textbf{John}}{\textbf{loves}\ x\ (\textbf{mother John})}}$$

$$\xRightarrow{\textsc{lower}} \frac{S}{\text{John's mother loves him}}$$
$$(\lambda x\ .\ \textbf{loves}\ x\ (\textbf{mother John}))\ \textbf{John}$$

$$= \frac{S}{\text{John's mother loves him}}$$
$$\textbf{loves John (mother John)}$$

Composing the towers yields a single tower of syntactic category $S /\!\!/ (S \setminus\!\!\setminus S)$, where the pronoun has been bound. The value **John** that binds the pronoun linearly precedes the pronoun. Applying LOWER and reducing the lambda expression then gives the desired reading where John's mother loves John.

Notice that in (42), the possessor can bind a pronoun without covert movement, as binding occurs from left to right, and information about binding is propagated by "a chain of matching $DP \triangleright S$'s connecting the possessor with the pronoun [that] is not disrupted by the major constituent boundary between the subject and verb phrase" (Barker and Shan 2014:27). Barker (2002) and Barker and Shan (2014) point to the linear nature of variable binding as a feature that allows the continuation framework to dispense with hierarchical relationships such as c-command. The framework's reliance on linear order for variable binding will prove to be important in future sections, and is worth keeping in mind.

4.1.2 Binding and the Tensed Clause Condition

Recall from Chapter 3.1 that quantifiers are blocked from taking scope out of scope islands. Scope island restrictions continue to apply in configurations involving pronouns; (43) does not have the reading where the universal scopes out of its minimal tensed clause over the existential.

(43) Someone$_i$ said ⟨he$_i$ liked everyone⟩.

I argued in Chapter 3.2 that the TCC forces expressions to lower at the edge of a tensed clause. Forcibly LOWER-ing the entire expression in (43) would similarly stop the universal from taking wide scope unexpectedly. However, the addition of the syntactic category $A \triangleright B$ into our toolbox poses a problem for effectively composing expressions containing pronouns across clause boundaries. Conducting LOWER proceeds smoothly, as shown in (44). However, the lowered syntactic tower is an expression of category $DP \triangleright S$, which then cannot compose with a functor expecting an S argument.

(44)

$\frac{\frac{DP \triangleright S \mid S}{S \mid S}}{DP}$	$\frac{\frac{S \mid S}{S \mid S}}{(DP \backslash S)/DP}$	$\frac{S \mid S}{S \mid S}$		$\frac{\frac{DP \triangleright S \mid S}{S \mid S}}{S}$
he	liked	everyone	=	he liked everyone
$\frac{\lambda x \,.\, [\,]}{[\,]}$	$\frac{[\,]}{[\,]}$	$\frac{[\,]}{\forall y \,.\, [\,]}$		$\frac{\lambda x \,.\, [\,]}{\forall y [\,]}$
x	**liked**	y		**liked** $y\, x$

$$\xRightarrow{\text{LOWER,LOWER}} \begin{array}{c} DP \triangleright S \\ \text{he liked everyone} \\ \lambda x \,.\, \forall y \,.\, \textbf{liked}\, y\, x \end{array}$$

While the final result of (44) is valid as a complete open proposition, in the absence of further specification, a sentence like *he liked everyone* cannot be embedded as is into (43), since functors seeking sentential arguments would be looking for an expression of category S rather than $DP \triangleright S$. Moreover, although its containing expression may be forcibly

LOWER-ed, a free pronoun should not be forced to reside on the bottom level of an expression. Rather, adopting the view that free pronouns take scope over their containing expressions implies that the scope of a pronoun should be able to propagate upstream until it can either be bound, or fully LOWER-ed to produce a complete open proposition. For instance, despite being located inside the clause marked ③, the deeply embedded pronoun in (45) would be expected to take scope over the widest embedded clause ①.

(45) Mary$_i$ said ⟨①that John thought ⟨②that Bill believed ⟨③that Tom liked her$_i$⟩⟩⟩.

In order for a pronoun to take scope over expressions on its left, it must reside on an upper level of the tower. As such, I propose a type-shifter that lifts pronouns out of the bottom level. I call this type-shifter PROLIFT, and label it P in derivations. Type-shifting also solves the problem of undefined combination on the bottom layer by returning a syntactic category of $DP \triangleright A$ to an upper level, where it can be bound by any DP that has been acted on using the BIND type-shifter.

Definition 10: PROLIFT type-shifter. For any syntactic categories A, B and C:

$$\begin{array}{c} \dfrac{B \mid C}{DP \triangleright A} \\ \text{exp} \\ f[\,] \\ \hline \lambda x\,.\,g(x) \end{array} \quad \overset{\text{PROLIFT}}{\Longrightarrow} \quad \begin{array}{c} \dfrac{DP \triangleright B \mid C}{A} \\ \text{exp} \\ \lambda x\,.\,f[\,] \\ \hline g(x) \end{array}$$

PROLIFT works as follows: given a tower where the bottom layer has syntactic category $DP \triangleright A$ — that is, given an expression of category A except that it contains an unbound pronoun of category DP and semantic type e — allow the pronoun to scope over the entire expression by lifting it onto the second layer. Semantically, the variable in g is saturated and then abstracted over on a higher level, allowing it to take scope over the entire expression, including any additional scopal values represented by f. (46) illustrates

how we can use PROLIFT to derive the expected reading of (43).

(46)

$$\frac{S \mid DP \triangleright S}{DP} \qquad \frac{DP \triangleright S \mid DP \triangleright S}{(DP \backslash S)/S} \qquad DP \triangleright S$$

John — said — he liked everyone

$[\,]\,\mathbf{John}$ — $[\,]$ — $\lambda x\,.\,\forall y\,.\,\mathbf{liked}\,y\,x$

$$\mathbf{John} \qquad \mathbf{said}$$

$$\frac{S \mid DP \triangleright S}{DP} \qquad \frac{DP \triangleright S \mid DP \triangleright S}{(DP \backslash S)/S} \qquad \frac{S \mid S}{DP \triangleright S}$$

$\xRightarrow{\text{LIFT}}$ John — said — he liked everyone

$[\,]\,\mathbf{John}$ — $[\,]$ — $[\,]$

$$\mathbf{John} \qquad \mathbf{said} \qquad \lambda x\,.\,\forall y\,.\,\mathbf{liked}\,y\,x$$

$$\frac{S \mid DP \triangleright S}{DP} \qquad \frac{DP \triangleright S \mid DP \triangleright S}{(DP \backslash S)/S} \qquad \frac{DP \triangleright S \mid S}{S}$$

$\xRightarrow{\text{PROLIFT}}$ John — said — he liked everyone

$[\,]\,\mathbf{John}$ — $[\,]$ — $\lambda x\,.\,[\,]$

$$\mathbf{John} \qquad \mathbf{said} \qquad \forall y\,.\,\mathbf{liked}\,y\,x$$

$$\frac{S \mid DP \triangleright S}{DP} \qquad \frac{DP \triangleright S \mid S}{DP \backslash S}$$

= John — said he liked everyone

$[\,]\,\mathbf{John}$ — $\lambda x\,.\,[\,]$

$$\mathbf{John} \qquad \mathbf{said}\,(\forall y\,.\,\mathbf{liked}\,y\,x)$$

$$\frac{S \mid S}{S}$$

= John said he liked everyone

$(\lambda x\,.\,[\,])\,\mathbf{John}$

$$\mathbf{said}\,(\forall y\,.\,\mathbf{liked}\,y\,x)\,\mathbf{John}$$

$\xRightarrow{\text{LOWER}}$ S

John said he liked everyone

$$(\lambda x\,.\,\mathbf{said}\,[\forall y\,.\,\mathbf{liked}\,y\,x]\,\mathbf{John})\,\mathbf{John}$$

= S

John said he liked everyone

$$\mathbf{said}\,(\forall y\,.\,\mathbf{liked}\,y\,\mathbf{John})\,\mathbf{John}$$

Beginning with the fully LOWER-ed embedded clause *he liked everyone* from (44), I apply LIFT to generate a trivial two-layered tower. I then PROLIFT the pronoun out of the bottom layer. To *said*, I apply LIFT, targeting the category $DP \triangleright S$. I LIFT *John* to the target category S, and apply BIND so that it expects a $DP \triangleright S$ expression to its right. Composition, LOWER and reduction yield as expected the desired reading where John said that for all people, John liked them.

4.2 Short-distance movement

I have so far looked at binding in sentences without gaps. In this section, I reintegrate movement into the framework. I illustrate how binding complicates the interaction between gapped clauses and fronted expressions. I show how to obtain multiple possible representations of gapped clauses using PROLIFT, and also show that two different categories of fronted expressions are needed to capture the binding potential of the fronted expression. Finally, I show how the category of fronted expression required to capture a crossover reading cannot be generated in the enhanced continuation framework.

4.2.1 Movement without binding

In the continuation framework, a sentence with an unbound pronoun is represented as an open proposition of category $DP \triangleright S$. It is thus predicted that in the sentences in (47), pronouns can remain unbound with the fronted expression to produce a final expression of syntactic category $DP \triangleright S$.

(47) a. The book, John gave ___ to her.
Target denotation: $\lambda x\ .\ \mathbf{gave}\,(\mathbf{to}\ x)\,\mathbf{book}\,\mathbf{John}$
Target category: $DP \triangleright S$
Target semantic type: $\langle e, t\rangle$

b. The book, he gave ___ to Mary.

Target denotation: λx . **gave** (**to Mary**) **book** x

Target category: $DP \triangleright S$

Target semantic type: $\langle e, t \rangle$

Sentences (47a-b) both contain a single gap and a single unbound pronoun; however, whereas in (47a) the gap linearly precedes the pronoun, in (47b) the pronoun precedes the gap. This distinction will affect how we compose them with their fronted expressions. Intuitively, this is because the continuation framework represents dependencies linearly rather than hierarchically.

In the case of (47a) where the gap linearly precedes the pronoun, the computation is relatively straightforward. I start with the gapped clause and declare a gap of category $(DP \mathbin{\diagdown\!\!\diagdown} (DP \triangleright S)) \mathbin{/\!\!/} (DP \mathbin{\diagdown\!\!\diagdown} (DP \triangleright S))$ — the category of a gap that takes in an expression missing a pronoun and returns an expression that wants first a gap and then a pronoun. This is possible since gaps are of any category $A \mathbin{/\!\!/} A$. I also apply LIFT on the remaining expressions to propagate effects on the upper layer, and combine them as shown in (48):

(48)

$$\dfrac{\dfrac{\dfrac{DP \mathbin{\diagdown\!\!\diagdown} (DP \triangleright S) \;\big|\; DP \mathbin{\diagdown\!\!\diagdown} (DP \triangleright S)}{DP}}{\begin{array}{c}\text{John}\\ [\,]\\ \hline \textbf{John}\end{array}} \quad \dfrac{\dfrac{DP \mathbin{\diagdown\!\!\diagdown} (DP \triangleright S) \;\big|\; DP \mathbin{\diagdown\!\!\diagdown} (DP \triangleright S)}{((DP \backslash S)/PP)/DP}}{\begin{array}{c}\text{gave}\\ [\,]\\ \hline \textbf{gave}\end{array}}$$

$$\dfrac{\dfrac{DP \mathbin{\diagdown\!\!\diagdown} (DP \triangleright S) \;\big|\; DP \triangleright S}{DP}}{\begin{array}{c}\text{\_\_\_}\\ \lambda x\,.\,[\,]\\ \hline x\end{array}} \quad \dfrac{\dfrac{DP \triangleright S \;\big|\; DP \triangleright S}{PP/DP}}{\begin{array}{c}\text{to}\\ [\,]\\ \hline \textbf{to}\end{array}} \quad \dfrac{\dfrac{DP \triangleright S \;\big|\; S}{DP}}{\begin{array}{c}\text{her}\\ \lambda y\,.\,[\,]\\ \hline y\end{array}}$$

$$= \dfrac{\dfrac{DP \mathbin{\diagdown\!\!\diagdown} (DP \triangleright S) \;\big|\; S}{S}}{\begin{array}{c}\text{John gave \_\_\_ to her}\\ \lambda x\,.\,\lambda y\,.\,[\,]\\ \hline \textbf{gave}\ x\ (\textbf{to}\ y)\ \textbf{John}\end{array}} \overset{\text{LOWER}}{\Longrightarrow} \dfrac{DP \mathbin{\diagdown\!\!\diagdown} (DP \triangleright S)}{\begin{array}{c}\text{John gave \_\_\_ to her}\\ \lambda x\,.\,\lambda y\,.\,\textbf{gave}\ x\ (\textbf{to}\ y)\ \textbf{John}\end{array}}$$

After combining and LOWER-ing, the expression *John gave ___ to her* has the syntactic category $DP \setminus\!\!\setminus (DP \triangleright S)$. This category correctly identifies the structure as one which would be a sentence with an unbound pronoun if it were first filled by a DP — in other words, it seeks first a filler for its gap, and then a binder for its pronoun. It combines simply with the FRONT-ed expression, as I show in (49):

(49)

$$\dfrac{\dfrac{(DP \triangleright S)_F \mid DP \triangleright S}{DP}}{\begin{array}{c}\text{the book}\\ \underline{[\,]}\\ \textbf{book}\end{array}} \qquad \begin{array}{c} DP \setminus\!\!\setminus (DP \triangleright S)\\ \text{John gave \_\_\_ to her}\\ \lambda x\,.\,\lambda y\,.\,\textbf{gave}\,x\,(\textbf{to}\,y)\,\textbf{John}\end{array}$$

$$\xRightarrow{\text{FRONT}} \begin{array}{c}(DP \triangleright S)/(DP \setminus\!\!\setminus (DP \triangleright S))\\ \text{book}\\ \lambda\kappa\,.\,\kappa\,\textbf{book}\end{array} \qquad \begin{array}{c} DP \setminus\!\!\setminus (DP \triangleright S)\\ \text{John gave \_\_\_ to her}\\ \lambda x\,.\,\lambda y\,.\,\textbf{gave}\,x\,(\textbf{to}\,y)\,\textbf{John}\end{array}$$

$$= \begin{array}{c} DP \triangleright S\\ \text{the book, John gave \_\_\_ to her}\\ \lambda y\,.\,\textbf{gave book}\,(\textbf{to}\,y)\,\textbf{John}\end{array}$$

In (49), I LIFT the fronted expression with the target category $DP \triangleright S$. I then apply FRONT to it. Finally, it combines with the gapped clause to give the expected result. So constructing an open proposition with movement where the gap precedes the pronoun is possible.

I turn now to (47b), a sentence where the pronoun linearly precedes the gap. In this case, since the gap occurs further to the right, it takes the syntactic category $(DP \setminus\!\!\setminus S) /\!\!/ (DP \setminus\!\!\setminus S)$ such that it takes in an expression of category S and returns a gapped clause. On the upper level, the pronoun *he* then takes in a gapped clause of category $DP \setminus\!\!\setminus S$ and returns a $DP \triangleright (DP \setminus\!\!\setminus S)$. All other expressions are LIFT-ed to match. Combining and applying LOWER yields an expression of category $DP \triangleright (DP \setminus\!\!\setminus S)$. I illustrate the derivation in (50):

(50)

$$\dfrac{\dfrac{DP \triangleright (DP \setminus\!\!\setminus S) \mid DP \setminus\!\!\setminus S}{DP}}{\begin{array}{c}\text{he}\\ \lambda x\,.\,[\,]\\ \hline x\end{array}} \quad \dfrac{\dfrac{DP \setminus\!\!\setminus S \mid DP \setminus\!\!\setminus S}{((DP \setminus S)/PP)/DP}}{\begin{array}{c}\text{gave}\\ [\,]\\ \hline \textbf{gave}\end{array}} \quad \dfrac{\dfrac{DP \setminus\!\!\setminus S \mid S}{DP}}{\begin{array}{c}\text{\_\_\_}\\ \lambda y\,.\,[\,]\\ \hline y\end{array}}$$

$$\dfrac{S \mid S}{PP/DP}\ \begin{array}{c}\text{to}\\ [\,]\\ \hline \textbf{to}\end{array} \qquad \dfrac{S \mid S}{DP}\ \begin{array}{c}\text{Mary}\\ [\,]\\ \hline \textbf{Mary}\end{array}$$

$$= \dfrac{\dfrac{DP \triangleright (DP \setminus\!\!\setminus S) \mid S}{S}}{\begin{array}{c}\text{he gave \_\_\_ to Mary}\\ \lambda x\,.\,\lambda y\,.\,[\,]\\ \hline \textbf{gave}\ y\ (\textbf{to Mary})\ x\end{array}} \overset{\text{LOWER}}{\Longrightarrow} \begin{array}{c}DP \triangleright (DP \setminus\!\!\setminus S)\\ \text{he gave \_\_\_ to Mary}\\ \lambda x\,.\,\lambda y\,.\,\textbf{gave}\ y\ (\textbf{to Mary})\ x\end{array}$$

We now have a gapped clause of category $DP \triangleright (DP \setminus\!\!\setminus S)$, and want it to act as an argument to a fronted expression of category $A/(DP \setminus\!\!\setminus A)$ for some category A. *Prima facie* doing so seems impossible, since the gapped clause wants to bind its pronoun before obtaining a filler for its gap. Indeed, there is no method symmetric to the derivation in (49) by which to compose a two-layered tower of category $DP \triangleright (DP \setminus\!\!\setminus S)$ with a FRONT-ed expression.

However, we can instead leverage the availability of PROLIFT to propagate the pronoun on a level higher than the fronted expression. Doing so creates a fronted expression that can compose with a gapped clause of category $DP \triangleright (DP \setminus\!\!\setminus S)$, allowing the computation to go through. I begin by defining the fronted expression with the value in (51):

(51)

$$\begin{array}{c}DP\\ \text{the book}\\ \textbf{book}\end{array} \overset{\text{LIFT}}{\Longrightarrow} \dfrac{\dfrac{S \mid S}{DP}}{\begin{array}{c}\text{the book}\\ [\,]\\ \hline \textbf{book}\end{array}} \overset{\text{FRONT}}{\Longrightarrow} \begin{array}{c}S/(DP \setminus\!\!\setminus S)\\ \text{the book}\\ \lambda\kappa\,.\,\kappa\,\textbf{book}\end{array} \overset{\text{LIFT}}{\Longrightarrow} \dfrac{\dfrac{DP \triangleright S \mid DP \triangleright S}{S/(DP \setminus\!\!\setminus S)}}{\begin{array}{c}\text{the book}\\ [\,]\\ \hline \lambda\kappa\,.\,\kappa\,\textbf{book}\end{array}}$$

In (51), I adopt a series of type-shifters that yield a three-layered tower where binding information — represented by $DP \triangleright S$ — is on the top layer, while fronting information is

on the middle layer; the binding effects and fronting effects are kept separate.

On the side of the gapped clause, I apply PROLIFT to lift the pronoun out of the bottom layer. Doing so generates a tower that can be combined with the fronted expression I made in (51), as I show in (52):

(52)

$$\dfrac{\dfrac{DP \triangleright S \mid DP \triangleright S}{S/(DP \setminus\!\!\setminus S)}\ \text{the book}\quad [\,]}{\lambda\kappa\,.\,\kappa\,\textbf{book}} \qquad \dfrac{DP \triangleright (DP \setminus\!\!\setminus S)}{\text{he gave \_\_\_ to Mary}\quad \lambda x\,.\,\lambda y\,.\,\textbf{gave}\,y\,(\textbf{to Mary})\,x}$$

$$\xRightarrow{\text{PROLIFT}} \dfrac{\dfrac{DP \triangleright S \mid DP \triangleright S}{S/(DP \setminus\!\!\setminus S)}\ \text{the book}\quad [\,]}{\lambda\kappa\,.\,\kappa\,\textbf{book}} \qquad \dfrac{\dfrac{DP \triangleright S \mid S}{DP \setminus\!\!\setminus S}\ \text{he gave \_\_\_ to Mary}\quad \lambda x\,.\,[\,]}{\lambda y\,.\,\textbf{gave}\,y\,(\textbf{to Mary})\,x}$$

$$= \dfrac{\dfrac{DP \triangleright S \mid S}{S}\ \text{the book, he gave \_\_\_ to Mary}\quad \lambda x\,.\,[\,]}{\textbf{gave book}\,(\textbf{to Mary})\,x}$$

$$\xRightarrow{\text{LOWER}} \dfrac{DP \triangleright S}{\text{the book, he gave \_\_\_ to Mary}\quad \lambda x\,.\,\textbf{gave book}\,(\textbf{to Mary})\,x}$$

After composing the towers, lowering the result gives the correct reading where some individual x gave a book to Mary.

There are two details worth noting in this pair of derivations. Firstly, note that when the gap linearly precedes the pronoun, the LOWER-ed clause is of category $DP \setminus\!\!\setminus (DP \triangleright S)$, while when the pronoun linearly precedes the gap, the LOWER-ed clause is of category $DP \triangleright (DP \setminus\!\!\setminus S)$. Secondly, note that because of these differing categories, we need two separate strategies for FRONT-ing. We need to be able to build both two- and three-layered towers in order to make the correct predictions about open propositions that involve short-

distance movement.

4.2.2 Binding from a fronted expression

I turn next to the binding potential of the fronted expression. In (53), I show two examples[6] where the fronted expression binds a pronoun in the gapped clause.

(53) a. Everything$_i$, John returned ___ to its$_i$ owner.

b. *Everything$_i$, John returned its$_i$ owner ___.

While the bound reading of (53a) is available, the same cannot be said of (53b), which is blocked as an instance of a (weak) crossover effect (Postal 1971). In the nomenclature of the continuation framework, crossover effects refer to the impossibility of a fronted phrase binding a pronoun that occurs before its gap. In (53b), the quantifier phrase *everything* is moved over the pronoun *its* such that its gap occurs to the right of the pronoun, and results in a configuration from which we cannot obtain the reading where every item is returned to its (potentially different) owner by John.

In this section, I will demonstrate how the enhanced continuation framework developed here can account for both the availability of (53a) and the unavailability of (53b). I first deal with (53a), a configuration in which the gap precedes the pronoun. First, I obtain an expression with the category $DP \setminus\!\!\setminus (DP \triangleright S)$ for the gapped clause, as shown in (54):

---

6. I use quantifier phrases here to avoid issues with accidental coreference which affect the availability of (ia) and the unavailability of (ib):

(i) a. Jack$_i$, I told his$_i$ wife that I had called ___. (Büring 2005:171)

b. *[Everybody else]$_i$, I told his$_i$ wife that I had called ___.

The pronoun in (ia) can corefer with a referential object. Similarly, (ib) is acceptable if we treat *his* as a referring pronoun. However, *his* cannot corefer with a quantifier phrase, and must be bound by the quantifier phrase in order to achieve the desired reading in (ib). In a crossover configuration, the quantifier phrase is unable to bind the pronoun. Hence, (ib) shows crossover effects.

(54)

$$\dfrac{\dfrac{DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S) \;\big|\; DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S)}{DP}}{\begin{array}{c}\text{John}\\ [\,]\\ \hline \textbf{John}\end{array}} \qquad \dfrac{\dfrac{DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S) \;\big|\; DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S)}{((DP\backslash S)/PP)/DP}}{\begin{array}{c}\text{returned}\\ [\,]\\ \hline \textbf{returned}\end{array}}$$

$$\dfrac{\dfrac{DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S) \;\big|\; DP \triangleright S}{DP}}{\begin{array}{c}\underline{\quad\quad}\\ \lambda x\,.\,[\,]\\ \hline x\end{array}} \qquad \dfrac{\dfrac{DP \triangleright S \;\big|\; DP \triangleright S}{PP/DP}}{\begin{array}{c}\text{to}\\ [\,]\\ \hline \textbf{to}\end{array}} \qquad \dfrac{\dfrac{DP \triangleright S \;\big|\; S}{DP}}{\begin{array}{c}\text{its owner}\\ \lambda y\,.\,[\,]\\ \hline \textbf{owner}\ y\end{array}}$$

$$\dfrac{DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S) \;\big|\; S}{S}$$

$$= \quad \dfrac{\text{John returned \_\_\_ to its owner}}{\lambda x\,.\,\lambda y\,.\,[\,]}$$

$$\textbf{returned}\ x\ (\textbf{to}\ (\textbf{owner}\ y))\ \textbf{John}$$

$$\overset{\textsc{lower}}{\Longrightarrow} \quad \begin{array}{c} DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S) \\ \text{John returned \_\_\_ to its owner} \\ \lambda x\,.\,\lambda y\,.\,\textbf{returned}\ x\ (\textbf{to}\ (\textbf{owner}\ y))\ \textbf{John}\end{array}$$

As in previous examples where the gap linearly precedes the pronoun, a gap is created with the syntactic category $(DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S)) \mathbin{/\!\!/} (DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S))$. Combining and applying LOWER to the result gives an expression of category $DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S)$.

Next, I apply BIND to the fronted expression *everything* so that it expects to fill a gap and also bind a rightward expression. The towers compose as shown in (55), and LOWER to give the expected reading where John returns all items to their respective owner(s).

(55)

$$\frac{S_F \mid DP \triangleright S}{DP} \qquad DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S)$$

everything — John returned ___ to its owner

$\forall z \,.\, [\,]\, z$ — $\lambda x \,.\, \lambda y \,.\, \mathbf{returned}\, x\, (\mathbf{to}\, (\mathbf{owner}\, y))\, \mathbf{John}$

$$\frac{}{z}$$

$$\xRightarrow{\text{FRONT}} \quad \underset{\lambda \kappa \,.\, \forall z \,.\, [\kappa z]\, z}{S/(DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S))} \qquad \underset{\lambda x \,.\, \lambda y \,.\, \mathbf{returned}\, x\, (\mathbf{to}\, (\mathbf{owner}\, y))\, \mathbf{John}}{DP \mathbin{\backslash\!\!\backslash} (DP \triangleright S)}$$

everything — John returned ___ to its owner

$$= \quad \underset{\forall z \,.\, [(\lambda x \,.\, \lambda y \,.\, \mathbf{returned}\, x\, (\mathbf{to}\, (\mathbf{owner}\, y))\, \mathbf{John})\, z]\, z}{S}$$

everything, John returned ___ to its owner

$$= \quad \underset{\forall z \,.\, \mathbf{returned}\, z\, (\mathbf{to}\, (\mathbf{owner}\, z))\, \mathbf{John}}{S}$$

everything, John returned ___ to its owner

I now demonstrate how the enhanced continuation framework developed here can block the crossover configuration of (53b). Firstly, I show that the syntactic category of *John returned its owner* ___ is $DP \triangleright (DP \mathbin{\backslash\!\!\backslash} S)$:

(56)

$$\frac{DP \triangleright (DP \mathbin{\backslash\!\!\backslash} S) \mid DP \triangleright (DP \mathbin{\backslash\!\!\backslash} S)}{DP} \qquad \frac{DP \triangleright (DP \mathbin{\backslash\!\!\backslash} S) \mid DP \triangleright (DP \mathbin{\backslash\!\!\backslash} S)}{((DP \backslash S)/DP)/DP}$$

John — returned

$[\,]$ — $[\,]$

$$\frac{}{\mathbf{John}} \qquad \frac{}{\mathbf{returned}}$$

$$\frac{DP \triangleright (DP \mathbin{\backslash\!\!\backslash} S) \mid DP \mathbin{\backslash\!\!\backslash} S}{DP} \qquad \frac{DP \mathbin{\backslash\!\!\backslash} S \mid S}{DP}$$

its owner — ___

$\lambda y \,.\, [\,]$ — $\lambda x \,.\, [\,]$

$$\frac{}{\mathbf{owner}\, y} \qquad \frac{}{x}$$

$$\frac{DP \triangleright (DP \mathbin{\backslash\!\!\backslash} S) \mid S}{S}$$

John returned ___ to its owner

$$= \quad \frac{\lambda y \,.\, \lambda x \,.\, [\,]}{\mathbf{returned}\, (\mathbf{owner}\, y)\, x\, \mathbf{John}}$$

$$DP \triangleright (DP \backslash\!\backslash S)$$
$$\xRightarrow{\text{LOWER}} \quad \text{John returned } \_\_\_ \text{ to its owner}$$
$$\lambda y\ .\ \lambda x\ .\ \textbf{returned}\,(\textbf{owner}\ y)\ x\ \textbf{John}$$

Using a gap of category $(DP \backslash\!\backslash S) /\!\!/ (DP \backslash\!\backslash S)$, I generate in (56) an expression of category $DP \triangleright (DP \backslash\!\backslash S)$. In Chapter 4.2.1, I alluded to the fact that separate strategies are necessary to deal with gapped clauses of categories $DP \backslash\!\backslash (DP \triangleright S)$ and $DP \triangleright (DP \backslash\!\backslash S)$. Specifically, I noted that a two-layered fronted expression must be of category $A/(DP \backslash\!\backslash B)$ for some categories A, B, and would thus seek a gapped clause of category $DP \backslash\!\backslash B$, which is fundamentally incompatible with a gapped clause of category $DP \triangleright (DP \backslash\!\backslash S)$. This fact is true here as well. Having ruled out the possibility of creating a two-layered fronted binder, let's postulate what a desired three-layered binder might look like. I lay out a putative three-layered tower for *everything* in (57).

(57) **Hypothetical syntactic category for a fronted expression binding a gapped clause of category** $DP \triangleright (DP \backslash\!\backslash S)$:

S	$DP \triangleright S$
S_F	S
DP	

everything

$\forall z\ .\ [\ ]\ z$

$[\ ]$

z

In (58), I show that *if* it were possible to generate such a syntactic category, an expression with that category could combine with the gapped clause *John returned its owner* ___ to predict the grammaticality of a crossover reading:

(58)

$$\begin{array}{c|c} S & DP \triangleright S \\ \hline S_F & S \end{array} \qquad DP \triangleright (DP \setminus\!\!\setminus S)$$

$$DP \qquad \text{John returned \_\_\_ to its owner}$$

$$\text{everything} \qquad \lambda y\,.\,\lambda x\,.\,\mathbf{returned}\,(\mathbf{owner}\,y)\,x\,\mathbf{John}$$

$$\forall z\,.\,[\,]\,z$$

$$[\,]$$

$$z$$

$$\xRightarrow{\text{PROLIFT}}$$

$$\begin{array}{c|c} S & DP \triangleright S \\ \hline S_F & S \end{array} \qquad \frac{DP \triangleright S \mid S}{DP \setminus\!\!\setminus S}$$

$$DP \qquad \text{John returned its owner \_\_\_}$$

$$\text{everything} \qquad \lambda x\,.\,[\,]$$

$$\forall z\,.\,[\,]\,z \qquad \lambda y\,.\,\mathbf{returned}\,(\mathbf{owner}\,y)\,x\,\mathbf{John}$$

$$[\,]$$

$$z$$

$$\xRightarrow{\text{FRONT}}$$

$$\frac{S \mid DP \triangleright S}{S/(DP \setminus\!\!\setminus S)} \qquad \frac{DP \triangleright S \mid S}{DP \setminus\!\!\setminus S}$$

$$\text{everything} \qquad \text{John returned its owner \_\_\_}$$

$$\forall z\,.\,[\,]\,z \qquad \lambda x\,.\,[\,]$$

$$\lambda \kappa\,.\,\kappa\,z \qquad \lambda y\,.\,\mathbf{returned}\,(\mathbf{owner}\,y)\,x\,\mathbf{John}$$

$$\frac{S \mid S}{S}$$

$$= \text{everything, John returned its owner \_\_\_} \qquad \forall z\,.\,(\lambda x\,.\,[\,])\,z$$

$$\mathbf{returned}\,(\mathbf{owner}\,z)\,x\,\mathbf{John}$$

$$S$$

$$\xRightarrow{\text{LOWER}} \text{everything, John returned its owner \_\_\_}$$

$$\forall z\,.\,(\lambda x\,.\,\mathbf{returned}\,(\mathbf{owner}\,z)\,x\,\mathbf{John})\,z$$

$$S$$

$$= \text{everything, John returned its owner \_\_\_}$$

$$\forall z\,.\,\mathbf{returned}\,(\mathbf{owner}\,z)\,z\,\mathbf{John}$$

In (58), starting with the proposed towers would successfully give the reading where John returns every item to its owner, which is unexpected and undesirable. Fortunately, it is impossible to build a three-layered fronted expression of the required syntactic category

using the type-shifters defined in this framework. This is because there is no possible order in which to apply type-shifters that will obtain the syntactic category desired.

In order to obtain the syntactic category above, we must apply LIFT, BIND and FRONT in some order to a syntactic category that starts out as $\frac{S \mid S}{DP}$. In order for binding information to be on the top layer, we must apply BIND before internally LIFT-ing the tower, since BIND always applies to the second-lowest layer in a multi-layered tower. In order for FRONT to apply only to the bottom layer of a tower, we must also apply FRONT before we LIFT, since FRONT applies to the entire syntactic category generated up till that point. These two conditions rule out every order of type-shifter other than BIND $\rightarrow$ FRONT $\rightarrow$ LIFT and FRONT $\rightarrow$ BIND $\rightarrow$ LIFT. However, if we apply both BIND *and* FRONT before LIFT, then BIND and FRONT both apply to the same second layer, generating a syntactic category of $\frac{S_F \mid DP \triangleright S}{DP}$ — not the result we want to achieve. So no sequence of the three type-shifters will yield the syntactic category we want. A clause of syntactic category $DP \triangleright DP \setminus\!\!\setminus S$ thus cannot combine with a fronted expression that wants to bind it, as a result of a mismatch in syntactic categories.

4.2.3 Summary

In this section, I presented four derivations: two used fronted expressions that bind a pronoun, while the others did not; two used gapped clauses where the pronoun preceded the gap; the others did the opposite. Through these computations, I showed that the fronted expression required to compose with a $DP \setminus\!\!\setminus (DP \triangleright S)$ is different from that required to compose with a $DP \triangleright (DP \setminus\!\!\setminus S)$ — a two-layered fronted expression is suitable for the former, while the latter demands a three-layered expression. I then pointed out that in the enhanced continuation grammar, the available type-shifters do not generate the fronted expression required to bind a pronoun in a crossover configuration. As a result, clauses of category $DP \setminus\!\!\setminus (DP \triangleright S)$ can be bound by the fronted expression, but clauses of category $DP \triangleright (DP \setminus\!\!\setminus S)$ cannot.

I summarize the patterns discussed in this section in Table 1. Note that the empty cell corresponds with the unavailability of the crossover reading.

Category of lowered gapped clause	Category of binding fronted expression	Category of non-binding fronted expression
$DP \setminus\!\!\setminus (DP \triangleright S)$	$\dfrac{S_F \mid DP \triangleright S}{DP}$	$\dfrac{(DP \triangleright S)_F \mid DP \setminus\!\!\setminus (DP \triangleright S)}{DP}$
$DP \triangleright (DP \setminus\!\!\setminus S)$	-	$\dfrac{\dfrac{DP \triangleright S \mid DP \triangleright S}{S_F \mid S}}{DP}$

Table 1: Syntactic categories necessary for composing with gapped clauses

4.3 Long distance binding fails

We have seen so far that B&S's $\triangleright$ syntactic operator integrates as expected into a continuation grammar that allows short-distance movement, demonstrating crossover effects as attested in Barker and Shan 2014. In this section, I show how the TCC poses an issue for the continuation framework's account of crossover effects in long-distance movement. Specifically, I demonstrate that in long-distance movement configurations, constructions that should exhibit crossover effects are not predicted to. Meanwhile, non-crossover constructions are predicted to demonstrate crossover effects.

I use the sentences in (59) to frame my discussion. Both sentences involve the long-distance topicalization of a quantifier phrase[7] from an embedded clause including a pro-

7. The examples in (59) may seem marked, paralleling Lasnik and Stowell (1991)'s claim that (bare) quantifier phrases cannot be topicalized, as in (i).

(i) *Everyone, I talked to ___.

Postal (1993:542) points out that the "addition of exceptive phrases... relative clauses... or adjective phrases yields grammatical results", as illustrated in (ii).

noun[8], but while the reading in (59a) is grammatical, the reading in (59b) is an instance of a crossover configuration, and should thus be ruled out.

(59) a. [Every father]$_i$, Mary expects ___ likes his$_i$ son.

b. ??[Every father]$_i$, Mary expects his$_i$ son likes ___.

4.3.1 Incorrectly predicting crossover effects

I first demonstrate that the enhanced continuation framework cannot capture the availability of the reading in (59a) because it predicts that an embedded clause that combines with an intermediate trace has the syntactic category $DP \triangleright (DP \mathbin{\backslash\!\backslash} S)$.

Recall that in long-distance movement configurations, we fully lower embedded clauses at the edge of their minimal tensed clause to comply with the TCC, at which point we add an intermediate gap and re-LIFT the clause. First, in (60), I show that the LOWER-ed form of ___ *liked his son*, the embedded gapped clause from (59a), has the syntactic category $DP \mathbin{\backslash\!\backslash} (DP \triangleright S)$. A gap of syntactic category $(DP \mathbin{\backslash\!\backslash} (DP \triangleright S)) \mathbin{/\!\!/} (DP \mathbin{\backslash\!\backslash} (DP \triangleright S))$ is required since the gap expects a pronoun to its right. I LIFT the other expressions to match.

(60)

$DP \mathbin{\backslash\!\backslash} (DP \triangleright S) \mid DP \triangleright S$	$DP \triangleright S \mid DP \triangleright S$	$DP \triangleright S \mid S$	$S \mid S$
DP	$(DP \backslash S)/DP$	DP	$DP \backslash DP$
___	likes	his	son
λw . [ ]	[ ]	λs . [ ]	[ ]
w	**likes**	s	**son**

(ii) a. Anyone else, they would have fired ___.

b. Everyone taller than Mary, I talked to ___.

c. Everyone who was sick, they would have fired ___.

Adding complexity to the QP should ameliorate any unavailable readings in (59). As these additions do not have side effects, they do not affect the proposal; I omit them for brevity.

8. In this thesis, I do not consider sentences where the pronoun is not in the embedded clause, e.g. (i).

(i) John, his sister said ___ was sad.

$$\frac{DP \backslash\!\backslash (DP \triangleright S) \mid S}{S}$$

$$= \quad \frac{\text{\_\_\_ likes his son} \qquad \lambda w\,.\,\lambda s\,.\,[\,]}{\textbf{likes}\,(\textbf{son}\,s)\,w}$$

$$\xRightarrow{\text{LOWER}} \quad \begin{array}{c} DP \backslash\!\backslash (DP \triangleright S) \\ \text{\_\_\_ likes his son} \\ \lambda w\,.\,\lambda s\,.\,\textbf{likes}\,(\textbf{son}\,s)\,w \end{array}$$

The next step in composition is to introduce an intermediate trace. I do so in (61), giving the trace the syntactic category $(DP \backslash\!\backslash S) /\!/ (DP \backslash\!\backslash S)$.

(61)

$$\frac{\dfrac{DP \backslash\!\backslash S \mid S}{DP} \qquad DP \backslash\!\backslash (DP \triangleright S)}{\dfrac{\text{\_\_\_} \qquad \langle\text{\_\_\_ likes his son}\rangle}{\dfrac{\lambda y\,.\,[\,]}{y} \qquad \lambda w\,.\,\lambda s\,.\,\textbf{likes}\,(\textbf{son}\,s)\,w}}$$

$$\xRightarrow{\text{LIFT}} \quad \frac{\dfrac{DP \backslash\!\backslash S \mid S}{DP} \qquad \dfrac{S \mid S}{DP \backslash\!\backslash (DP \triangleright S)}}{\dfrac{\text{\_\_\_} \qquad \langle\text{\_\_\_ likes his son}\rangle}{\dfrac{\lambda y\,.\,[\,]}{y} \qquad [\,]}}$$

$$\frac{\dfrac{DP \backslash\!\backslash S \mid S}{DP \triangleright S}}{= \quad \text{\_\_\_}\,\langle\text{\_\_\_ likes his son}\rangle \quad \lambda y\,.\,[\,]}$$

$$\frac{\lambda s\,.\,\textbf{likes}\,(\textbf{son}\,s)\,y}{\dfrac{\dfrac{DP \triangleright DP \backslash\!\backslash S \mid S}{S}}{\xRightarrow{\text{PROLIFT}} \quad \text{\_\_\_}\,\langle\text{\_\_\_ likes his son}\rangle \quad \lambda s\,.\,\lambda y\,.\,[\,]}}$$

$$\textbf{likes}\,(\textbf{son}\,s)\,y$$

In (61), I LIFT the gapped clause so that it can combine with the intermediate gap. After composing the towers, I PROLIFT the pronoun out of the bottom level of the tower. An

unintended side effect of doing so is that the pronoun then precedes the existing lambda operator of the intermediate gap in the upper layer — the expression morphs from one that expects a filler for its gap to one that first expects a binder for its pronoun. Correspondingly, the syntactic category of the expression changes from $DP \setminus\!\!\setminus (DP \triangleright S)$ to $DP \triangleright (DP \setminus\!\!\setminus S)$. Recall that $DP \triangleright (DP \setminus\!\!\setminus S)$ is exactly the category of a simple clause in which a pronoun precedes a gap — see Table 1. In other words, the syntactic category of the expression we are building in (61) is indistinguishable from that of a clause that exhibits a crossover configuration in local movement.

This mishap is not ameliorated by combining the expression with the rest of the clause, as I show in (62):

(62)
$$\frac{\dfrac{\dfrac{DP \triangleright DP \setminus\!\!\setminus S \;\big|\; DP \triangleright DP \setminus\!\!\setminus S}{DP}}{\text{Mary}}\;[\,]}{\textbf{Mary}} \qquad \frac{\dfrac{\dfrac{DP \triangleright DP \setminus\!\!\setminus S \;\big|\; DP \triangleright DP \setminus\!\!\setminus S}{(DP \setminus S)/S}}{\text{expects}}\;[\,]}{\textbf{expects}} \qquad \frac{\dfrac{\dfrac{DP \triangleright DP \setminus\!\!\setminus S \;\big|\; S}{S}}{\text{\_\_\_ }\langle\text{\_\_\_ likes his son}\rangle}\;\lambda s\,.\,\lambda y\,.\,[\,]}{\textbf{likes}\,(\textbf{son}\,s)\,y}$$

$$\frac{\dfrac{DP \triangleright DP \setminus\!\!\setminus S \;\big|\; S}{S}}{\text{= Mary expects \_\_\_ }\langle\text{\_\_\_ likes his son}\rangle}\;\lambda s\,.\,\lambda y\,.\,[\,]$$

$$\textbf{expects}\,(\textbf{likes}\,(\textbf{son}\,s)\,y)\,\textbf{Mary}$$

$$\frac{DP \triangleright DP \setminus\!\!\setminus S}{\text{= Mary expects \_\_\_ }\langle\text{\_\_\_ likes his son}\rangle}\;\lambda s\,.\,\lambda y\,.\,\textbf{expects}\,(\textbf{likes}\,(\textbf{son}\,s)\,y)\,\textbf{Mary}$$

In (62), I LIFT both *Mary* and *expects* so that they combine with the gapped clause. Composition yields a final tower whose syntactic category still LOWERS to $DP \triangleright DP \setminus\!\!\setminus S$. Recall from Table 1 that the type of BIND-shifted fronted expression required to compose with a gapped clause of category $DP \triangleright (DP \setminus\!\!\setminus S)$ cannot be created. Thus, the bound reading of (59a) is predicted to be unavailable — the continuation framework predicts that weak crossover effects will occur here, contrary to fact.

4.3.2 Failing to predict crossover effects

Symmetrically, the actual crossover configuration in (59b) — reproduced below as (63) — should be blocked, but is predicted to be grammatical under the enhanced continuation framework. This prediction occurs because the gapped clause *John said his son liked* ___, after LOWER-ing to satisfy the TCC, can be LIFT-ed into an expression that will eventually LOWER to a $DP \setminus\!\!\setminus (DP \triangleright S)$.

(63) ??[Every father]$_i$, Mary expects his$_i$ son likes ___. =(59b)

The LOWER-ed syntactic category of the gapped clause *his son likes* ___ in (59b) is $DP \triangleright (DP \setminus\!\!\setminus S)$. I derive this result in (64), using a gap of category $(DP \setminus\!\!\setminus S) /\!\!/ (DP \setminus\!\!\setminus S)$ and a pronoun that expects a gapped clause to its right. All other expressions are LIFT-ed to match.

(64)

$DP \triangleright (DP \setminus\!\!\setminus S) \mid DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S \mid DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S \mid DP \setminus\!\!\setminus S$	$DP \setminus\!\!\setminus S \mid S$
DP	$DP \backslash DP$	$(DP \backslash S)/DP$	DP
his	son	likes	___
$\lambda s\,.\,[\,]$	$[\,]$	$[\,]$	$\lambda w\,.\,[\,]$
s	**son**	**likes**	w

$$\frac{DP \triangleright (DP \setminus\!\!\setminus S) \mid S}{S}$$

$$= \quad \frac{\text{his son likes ___} \qquad \lambda s\,.\,\lambda w\,.\,[\,]}{\textbf{likes}\ w\ (\textbf{son}\ s)}$$

$$\xRightarrow{\text{LOWER}} \quad \frac{DP \triangleright (DP \setminus\!\!\setminus S) \qquad \text{his son likes ___}}{\lambda s\,.\,\lambda w\,.\,\textbf{likes}\ w\ (\textbf{son}\ s)}$$

In (65), I illustrate how adding an intermediate gap reverses the order in which an expression expects to fill its gap and bind the pronoun. Composing a gap of category $(DP \setminus\!\!\setminus (DP \triangleright S)) /\!\!/ (DP \setminus\!\!\setminus (DP \triangleright S))$ with the gapped clause forms an expression that will,

when LOWER-ed, be of category $DP \setminus\!\!\setminus (DP \triangleright S)$ — exactly the syntactic category of a clause in which an expression that linearly precedes a pronoun is fronted.

(65)

$$\dfrac{\dfrac{DP \setminus\!\!\setminus DP \triangleright S \;\big|\; DP \triangleright S}{DP}}{\dfrac{\_\_}{\dfrac{\lambda y\,.\,[\,]}{y}}} \qquad \begin{array}{c} DP \triangleright (DP \setminus\!\!\setminus S) \\ \text{his son likes } \_\_ \\ \lambda s\,.\,\lambda w\,.\,\mathbf{likes}\,w\,(\mathbf{son}\,s) \end{array}$$

$$\xRightarrow{\text{LIFT}} \qquad \dfrac{\dfrac{DP \setminus\!\!\setminus DP \triangleright S \;\big|\; DP \triangleright S}{DP}}{\dfrac{\_\_}{\dfrac{\lambda y\,.\,[\,]}{y}}} \qquad \dfrac{\dfrac{S \;\big|\; S}{DP \triangleright (DP \setminus\!\!\setminus S)}}{\begin{array}{c} \text{his son likes } \_\_ \\ [\,] \\ \hline \lambda s\,.\,\lambda w\,.\,\mathbf{likes}\,w\,(\mathbf{son}\,s) \end{array}}$$

$$\xRightarrow{\text{PROLIFT}} \qquad \dfrac{\dfrac{DP \setminus\!\!\setminus DP \triangleright S \;\big|\; DP \triangleright S}{DP}}{\dfrac{\_\_}{\dfrac{\lambda y\,.\,[\,]}{y}}} \qquad \dfrac{\dfrac{DP \triangleright S \;\big|\; S}{DP \setminus\!\!\setminus S}}{\begin{array}{c} \text{his son likes } \_\_ \\ \lambda s\,.\,[\,] \\ \hline \lambda w\,.\,\mathbf{likes}\,w\,(\mathbf{son}\,s) \end{array}}$$

$$= \dfrac{DP \setminus\!\!\setminus DP \triangleright S \;\big|\; S}{\begin{array}{c} S \\ \hline \_\_ \text{ his son likes } \_\_ \\ \lambda s\,.\,\lambda y\,.\,[\,] \\ \hline \mathbf{likes}\,y\,(\mathbf{son}\,s) \end{array}}$$

In (66), I feed the gapped clause with its intermediate gap into the rest of the sentence. Again, *Mary* and *expects* are LIFT-ed to match. Applying LOWER yields as predicted an expression of syntactic category $DP \setminus\!\!\setminus DP \triangleright S$.

(66)

$$\dfrac{DP \setminus\!\!\setminus DP \triangleright S \;\big|\; DP \setminus\!\!\setminus DP \triangleright S}{\begin{array}{c} DP \\ \text{Mary} \\ [\,] \\ \hline \mathbf{Mary} \end{array}} \qquad \dfrac{DP \setminus\!\!\setminus DP \triangleright S \;\big|\; DP \setminus\!\!\setminus DP \triangleright S}{\begin{array}{c} (DP \setminus S)/S \\ \text{expects} \\ [\,] \\ \hline \mathbf{expects} \end{array}} \qquad \dfrac{DP \setminus\!\!\setminus DP \triangleright S \;\big|\; S}{\begin{array}{c} S \\ \_\_ \text{ his son likes } \_\_ \\ \lambda s\,.\,\lambda y\,.\,[\,] \\ \hline \mathbf{likes}\,y\,(\mathbf{son}\,s) \end{array}}$$

$$\frac{DP \setminus\!\!\setminus DP \triangleright S \;\big|\; S}{S}$$

$$= \frac{\text{Mary expects \_\_\_ his son likes \_\_\_}}{\lambda s\,.\,\lambda y\,.\,[\,]}$$

$$\mathbf{expects}\,(\mathbf{likes}\,y\,(\mathbf{son}\,s))\,\mathbf{Mary}$$

$$\overset{\text{LOWER}}{\Longrightarrow} \begin{array}{c} DP \setminus\!\!\setminus DP \triangleright S \\ \text{Mary expects \_\_\_ his son likes \_\_\_} \\ \lambda s\,.\,\lambda y\,.\,\mathbf{expects}\,(\mathbf{likes}\,y\,(\mathbf{son}\,s))\,\mathbf{Mary} \end{array}$$

The long-distance gapped clause now composes well with the fronted expression *every father*, as I show in (67). I apply BIND to the fronted expression and then FRONT it so that it takes in the gapped clause as an argument.

(67)
$$\frac{\dfrac{S_F \;\big|\; DP \triangleright S}{DP} \qquad DP \setminus\!\!\setminus DP \triangleright S}{f}$$

$$\frac{\begin{array}{c}\text{every father} \\ \forall f\,.\,\mathbf{father}\,f \longrightarrow [\,]\,f\end{array} \qquad \begin{array}{c}\text{Mary expects \_\_\_ his son likes \_\_\_} \\ \lambda s\,.\,\lambda y\,.\,\mathbf{expects}\,(\mathbf{likes}\,y\,(\mathbf{son}\,s))\,\mathbf{Mary}\end{array}}{f}$$

$$\overset{\text{FRONT}}{\Longrightarrow} \begin{array}{c} S/(DP \setminus\!\!\setminus (DP \triangleright S)) \qquad DP \setminus\!\!\setminus DP \triangleright S \\ \text{every father} \qquad\qquad \text{Mary expects \_\_\_ his son likes \_\_\_} \\ \lambda\kappa\,.\,\forall f\,.\,\mathbf{father}\,f \longrightarrow [\kappa f]\,f \qquad \lambda s\,.\,\lambda y\,.\,\mathbf{expects}\,(\mathbf{likes}\,y\,(\mathbf{son}\,s))\,\mathbf{Mary} \end{array}$$

$$= \frac{\text{every father, Mary expects \_\_\_ his son likes \_\_\_}}{\forall f\,.\,\mathbf{father}\,f \longrightarrow [(\lambda s\,.\,\lambda y\,.\,\mathbf{expects}\,(\mathbf{likes}\,y\,(\mathbf{son}\,s))\,\mathbf{Mary})\,f]\,f}$$

$$= \frac{\text{every father, Mary expects \_\_\_ his son likes \_\_\_}}{\forall f\,.\,\mathbf{father}\,f \longrightarrow \mathbf{expects}\,(\mathbf{likes}\,f\,(\mathbf{son}\,f))\,\mathbf{Mary}}$$

Beta reduction of the lambdas in (67) yields the reading where Mary expects all sons to like their own fathers, a reading which should be unavailable. Thus, the continuation framework incorrectly predicts the availability of a crossover reading.

We have learned from the derivations in this section that LOWER-ing a gapped clause with a pronoun and introducing an intermediate gap can lead to incorrectly attributing the

syntactic category $DP \triangleright (DP \setminus\!\!\setminus S)$ to a clause that is underlyingly a $DP \setminus\!\!\setminus (DP \triangleright S)$, and $DP \setminus\!\!\setminus (DP \triangleright S)$ to a clause that is underlyingly a $DP \triangleright (DP \setminus\!\!\setminus S)$. This show us a major flaw in the continuation framework: scopal relationships devolve into linear relationships when an expression is LOWER-ed, and cannot be recovered after. While we have leveraged this behavior to prevent universals from scoping out of tensed clauses, the loss of scopal relationships in variable binding is detrimental to the theory. In particular, under the continuation framework, pronouns are proposed to scope over their surrounding clauses and are furthermore not clause-bound. I thus proposed PROLIFT to allow pronouns to scope over a LOWER-ed clause. However, applying PROLIFT interferes with the linear relationship that must be enforced between the gap and pronoun to explain crossover configurations under this framework. As a result, the grammar both undergenerates and overgenerates.

4.4 Towards a solution

Let's take a step back and consider what contributes to the above problem. Doing so will illuminate what solutions we can consider. The unexpected reordering of expected arguments is caused by forcibly LOWER-ing gapped clauses at clause boundaries to satisfy the TCC. We can thus consider either not applying the TCC to gapped clauses, or proposing alternate ways to apply the intermediate trace, LIFT and PROLIFT to move a clause from its LOWER-ed position to a position where it can continue to compose.

4.4.1 The Tensed Clause Condition does not apply

A first solution to the problem involves not LOWER-ing gapped clauses at the edge of clause boundaries at all so as to avoid the use of PROLIFT and the intermediate gap entirely. However, as discussed in Chapters 3.1 and 4.1.2, not complying with the TCC predicts that quantifiers in scope islands, such as *everyone* in (68), can take wide scope,contrary to fact.

(68) John$_i$, someone said ⟨___ gave his$_i$ book to everyone⟩.

In (68), we should not be able to obtain the reading where *everyone* takes wide scope over *someone*. However, in the absence of LOWER-ing to meet the TCC, this reading is available, as I show in (69). In (69), I use four-layered towers. I internally LIFT the universal so it is located on the third level of the tower from the bottom and can take scope over the existential.

(69)

$DP \setminus\!\!\setminus (DP \triangleright S)$	$DP \setminus\!\!\setminus (DP \triangleright S)$
S	S
S	S

DP

someone

[ ]

[ ]

$\exists y$. [ ]

y

$DP \setminus\!\!\setminus (DP \triangleright S)$	$DP \setminus\!\!\setminus (DP \triangleright S)$
S	S
S	S

$(DP \setminus S)/S$

said

[ ]

[ ]

[ ]

said

$DP \setminus\!\!\setminus (DP \triangleright S)$	$DP \triangleright S$
S	S
S	S

DP

—

λw . [ ]

[ ]

[ ]

w

$DP \triangleright S$	$DP \triangleright S$
S	S
S	S

$((DP \setminus S)/DP)/PP$

gave

[ ]

[ ]

[ ]

gave

$DP \triangleright S$	S
S	S
S	S

DP

his book

λb . [ ]

[ ]

[ ]

book b

S	S
S	S
S	S

PP

to everyone

[ ]

$\forall x$. [ ]

[ ]

to x

=

$DP \setminus\!\!\setminus (DP \triangleright S)$	$DP \setminus\!\!\setminus (DP \triangleright S)$
S	S
S	S

DP

someone

[ ]

[ ]

$\exists y$. [ ]

y

$DP \setminus\!\!\setminus (DP \triangleright S)$	$DP \setminus\!\!\setminus (DP \triangleright S)$
S	S
S	S

$(DP \setminus S)/S$

said

[ ]

[ ]

[ ]

said

$$\frac{\begin{array}{c|c} DP \setminus\!\!\setminus (DP \triangleright S) & S \\ \hline S & S \\ \hline S & S \end{array}}{S}$$

___ gave his book to everyone

$$\frac{\begin{array}{c} \lambda w\,.\,\lambda b\,.\,[\,] \\ \hline \forall x\,.\,[\,] \\ \hline [\,] \end{array}}{\textbf{gave}\,(\textbf{book}\,b)\,(\textbf{to}\,x)\,w}$$

$$\frac{\begin{array}{c|c} DP \setminus\!\!\setminus (DP \triangleright S) & S \\ \hline S & S \\ \hline S & S \end{array}}{S}$$

= someone said ___ gave his book to everyone

$$\frac{\begin{array}{c} \lambda w\,.\,\lambda b\,.\,[\,] \\ \hline \forall x\,.\,[\,] \\ \hline \exists y\,.\,[\,] \end{array}}{\textbf{said}\,(\textbf{gave}\,(\textbf{book}\,b)\,(\textbf{to}\,x)\,w)\,y}$$

$$DP \setminus\!\!\setminus (DP \triangleright S)$$

$\xRightarrow{\text{LOWER}\times\mathbf{3}}$ someone said ___ gave his book to everyone

$$\lambda w\,.\,\lambda b\,.\,\forall x\,.\,\exists y\,.\,\textbf{said}\,(\textbf{gave}\,(\textbf{book}\,b)\,(\textbf{to}\,x)\,w)\,y$$

Note that in the final result of (69), the universal takes scope over the existential quantifier. Composing the final result of (69) with a fronted, BIND-shifted expression would incorrectly predict the availability of a reading where for each person, there exists (a potentially different) someone who said that John gave John's book to that person. Thus, the TCC must continue to hold to stop universals from scoping out of their tensed clauses.

4.4.2 Altering the type-shifters

Having established that the TCC must hold for independent reasons, I now consider the possibility of proposing additional constraints on available type-shifters. I show how a deeper problem with the continuation framework emerges by going by attempting to

constrain the PROLIFT type-shifter so that crossover configurations are unable to make use of it.

Since there are two actions involved in returning a fully LOWER-ed gapped clause with a pronoun to a configuration that can compose in tower notation, we have two potential ways to constrain how gapped clauses combine with their intermediate traces:

(70) **Option 1**: Constrain the category, value, or use of PROLIFT such that clauses in crossover configurations are unable to be PROLIFT-ed.
Option 2: Constrain the category, value, or use of the intermediate trace such that clauses in crossover configurations are unable to compose with their matrix clause.

Unfortunately, the same mechanisms that would properly constrain crossover configurations such as (71a) would also predict the ungrammaticality of (71b), which resembles a crossover configuration except that the pronoun is not bound by the fronted expression.

(71) a. ?? [Every father]$_i$, Mary expects his$_i$ son likes ___. =(59b)

b. [Every father]$_i$, Mary expects his$_j$ son likes ___.

As an example, consider a scenario where we alter the PROLIFT type-shifter to only operate on $DP \triangleright S$, as opposed to the PROLIFT type-shifter proposed in Definition 10, which applies to $DP \triangleright A$ for any A.

(72) **Hypothetical restricted variant of PROLIFT**:

$$\frac{\dfrac{B \mid C}{DP \triangleright S}\ \text{exp}\ f[\,]}{\lambda x\ .\ g(x)} \quad \overset{\text{PROLIFT}}{\Longrightarrow} \quad \frac{\dfrac{DP \triangleright B \mid C}{S}\ \text{exp}\ \lambda x\ .\ f[\,]}{g(x)}$$

So constraining the type-shifter would block the gapped clause *his son likes* ___ from PROLIFT-ing after it is fully LOWER-ed at the edge of a tensed clause, since the syntactic category of the clause, shown in (73), would not meet the requirements stipulated in (72).

(73)

$$\begin{array}{c} DP \triangleright (DP \setminus\!\!\setminus S) \\ \text{his son likes } \_\_\_ \\ \lambda s\,.\,\lambda w\,.\,\textbf{likes}\ w\,(\textbf{son}\ s) \end{array} \overset{\text{LIFT}}{\Longrightarrow} \cfrac{\cfrac{S \mid S}{DP \triangleright (DP \setminus\!\!\setminus S)}}{\begin{array}{c} \text{his son likes } \_\_\_ \\ \lambda s\,.\,[\ ] \end{array}} \\ \lambda w\,.\,\textbf{likes}\ w\,(\textbf{son}\ s)$$

However, the continuation framework treats the gapped clauses in (71a-b) the same, so blocking one would force us to block the other as well. In general, because the same type-shifters that result in undesired categories are also used to derive desired syntactic categories, making changes to type-shifters would not solve the twin problems of over- and undergeneration. We cannot block undesired configurations without also blocking desired configurations.

4.5 The root of the problem

Although the continuation framework distinguishes between pronoun-bearing expressions $DP \triangleright S$ and gap-bearing expressions $DP \setminus\!\!\setminus S$ in terms of syntactic categories, both gaps and pronouns exist in the same paradigm and can therefore interact and take scope over each other. Indeed, Barker and Shan (2008:37) note that "the point of [their] system is to bring scope and binding together" (37).

B&S's continuation framework stands in contrast to more widely adopted LF-based semantic theories, such as that presented in Heim and Kratzer 1998, which treats binding and scope using two different mechanisms that do not necessarily interact. In LF-based semantic theories, binding relies on assignment functions, an entirely separate mechanism which largely does not interact with composition. However, these semantic theories do not immediately explain or rule out crossover configurations and require recourse to syntactic frameworks to do so (Heim and Kratzer 1998:265). The ability of the continuation

framework to straightforwardly obtain an analysis of local crossover is thus an important argument for the integration of binding and scope.

However, the move to integrate binding and movement is also precisely the source of the problem highlighted in this chapter. Firstly, note that in the continuation framework, binding is an inherently linear process — a pronoun makes its way upstream until a point where it is bound — while scope is not — a quantifier can take scope outside of its linear position. Then, note that the need to constrain the scope of quantifiers requires stipulating the TCC for LOWER-ing expressions, which collapses scopal relationships; in other words, linear relationships that exist across the same layer in multiple towers are not preserved long-distance.

In this thesis, I rescued pronouns trapped by LOWER-ing by appealing to on B&S's view that pronouns can take scope over their clauses. I proposed PROLIFT, by which pronouns are able to take scope over a LOWER-ed clause. The net result of my proposal is that the scope of pronouns can continually increase. However, an unwanted corollary of this move is that pronouns can unintentionally scope over gaps that linearly precede them — again, the relationship between scope and linear order breaks down. Thus, one of the core premises of the continuation framework — that binding and movement can be conceived of together — in fact poses a fundamental problem for the framework.

CHAPTER 5
CONCLUSION

In this thesis, I looked to the continuation framework proposed by B&S as a semantic theory that claims to account not only for quantifier scope ambiguity, but also for movement and variable binding.

The continuation framework that B&S propose does not immediately account for the clause-boundedness of quantifiers in embedded sentences, nor does it consider long-distance movement configurations. Bringing these two types of configurations to bear on the theory, I formulated the Tensed Clause Condition following suggestions in Barker and Shan 2008 and Charlow 2014 to account for the clause-boundedness of quantifier scope. I then proposed the use of intermediate gaps to allow for long-distance movement configurations while satisfying the TCC. Next, I attempted to extend B&S's treatment of variable binding to examples containing embedded clauses. Doing so surfaced a serious flaw in the continuation framework — the scopal relationships that are crucial in B&S's view of variable binding cannot be recovered once pronoun-containing expressions are LOWER-ed.

Indeed, the core operators of the continuation framework, when faced with the combination of clause-bound quantifiers, long-distance movement, and variable binding, seem to make fundamentally flawed predictions. A key feature of the continuation grammar — the uniform treatment of scope-taking and variable-binding — appears to be at fault. Future work should thus be done to explore how fatal a flaw this interaction presents, as well as to evaluate related frameworks such as Charlow 2014.

REFERENCES

Barker, Chris. 2002. Continuations and the Nature of Quantification. *Natural Language Semantics* 10:211–242.

Barker, Chris, and Chung-chieh Shan. 2006. Types as Graphs: Continuations in Type Logical Grammar. *Journal of Logic, Language and Information* 15:331–370.

Barker, Chris, and Chung-chieh Shan. 2008. Donkey anaphora is in-scope binding. *Semantics and Pragmatics* 1:1–1–46.

Barker, Chris, and Chung-chieh Shan. 2014. *Continuations and Natural Language*. Oxford University Press.

Bošković, Željko. 2004. Topicalization, Focalization, Lexical Insertion, and Scrambling. *Linguistic Inquiry* 35:613–638.

Büring, Daniel. 2005. *Binding Theory*. Cambridge Textbooks in Linguistics. Cambridge: Cambridge University Press.

Cecchetto, Carlo. 2004. Explaining the locality conditions of QR: Consequences for the theory of phases. *Natural Language Semantics* 12:345–397.

Charlow, Simon. 2014. On the semantics of exceptional scope. Doctoral Dissertation, New York University, New York: New York.

Chomsky, Noam. 1981. *Lectures on government and binding*. Foris Publications.

Crain, Stephen, and Diane Lillo-Martin. 1999. *An Introduction to Linguistic Theory and Language Acquisition*. Blackwell Textbooks in Linguistics. Wiley.

Dowty, David. 2007. Compositionality as an empirical problem. In *Direct compositionality*, ed. Chris Barker and Pauline Jacobson, volume 14 of *Oxford Studies in Theoretical Linguistics*, 23–101. Oxford: Oxford University Press.

Erlewine, Michael Yoshitaka. 2016. EL5101: Grammatical Analysis Handout 10. Lecture notes.

Farkas, Donka F., and Anastasia Giannakidou. 1996. How Clause-bounded is the Scope of Universals? *Semantics and Linguistic Theory* 6:35–52.

Giorgolo, Gianluca, and Ash Asudeh. 2012. Monads for conventional implicatures. In *Proceedings of Sinn Und Bedeutung 16*, volume 1, 265–278. MIT Working Papers in Linguistics.

Heim, Irene, and Angelika Kratzer. 1998. *Semantics in Generative Grammar*. Blackwell Textbooks in Linguistics. Wiley.

Hendriks, H. 1993. *Studied Flexibility: Categories and Types in Syntax and Semantics*. ILLC Dissertation Series. Institute for Logic, Language and Computation, Universiteit van Amsterdam.

Jacobson, Pauline. 1999. Towards a Variable-Free Semantics. *Linguistics and Philosophy* 22:117–184.

Johnson, Kyle. 2000. How Far Will Quantifiers Go? In *Step by step: Essays on Minimalist syntax in honor of Howard Lasnik*, ed. Roger Martin, David Michaels, and Juan Uriagereka, 187–210. MIT Press.

Kiselyov, Oleg, and Chung-chieh Shan. 2014. Continuation Hierarchy and Quantifier Scope. In *Formal Approaches to Semantics and Pragmatics*, ed. E. McCready, Katsuhiko Yabushita, and Kei Yoshimoto, 105–134. Dordrecht: Springer Netherlands.

Lasnik, Howard, and Tim Stowell. 1991. Weakest Crossover. *Linguistic Inquiry* 22:687–720.

May, Robert Carlen. 1977. The Grammar of Quantification. Doctoral Dissertation, Massachusetts Institute of Technology, Cambridge, MA.

Partee, Barbara H. 1987. Noun Phrase Interpretation and Type-shifting Principles. In *Studies in discourse representation theory and the theory of generalized quantifiers*, ed. Jeroen Groenindijk, Dick de Jongh, and Martin Stokhof, number 8 in Groningen-Amsterdam Studies in Semantics, 115–143. Dordrecht: Foris.

Postal, Paul M. 1971. *Cross-over Phenomena.* Transatlantic Series in Linguistics. Holt, Rinehart and Winston.

Postal, Paul M. 1993. Remarks on Weak Crossover Effects. *Linguistic Inquiry* 24:539–556.

Shan, Chung-chieh. 2001. Monads for natural language semantics. In *Proceedings of the ESSLLI-2001 Student Session*, ed. Kristina Striegnitz, 285–298. Helsinki.

Shan, Chung-chieh. 2007. Linguistic side effects. In *Direct compositionality*, ed. Chris Barker and Pauline Jacobson, 132–163. Oxford: Oxford University Press.

Shan, Chung-Chieh, and Chris Barker. 2006. Explaining Crossover and Superiority as Left-to-right Evaluation. *Linguistics and Philosophy* 29:91–134.

Steedman, Mark, and Jason Baldridge. 2011. Combinatory Categorial Grammar. In *Non-Transformational Syntax: Formal and Explicit Models of Grammar*, ed. Robert Borsley and Kersti Borjars, 62. Wiley-Blackwell.

Szabolcsi, Anna, ed. 1997. *Ways of Scope Taking*, volume 65 of *Studies in Linguistics and Philosophy*. Dordrecht: Springer.

Wurmbrand, Susanne. 2018. The cost of raising quantifiers. *Glossa: a journal of general linguistics* 3:19.