EXPLAINING CROSSOVER AND SUPERIORITY AS LEFT-TO-RIGHT EVALUATION

CHUNG-CHIEH SHAN AND CHRIS BARKER

ABSTRACT. We present a general theory of scope and binding in which both crossover and superiority violations are ruled out by one key assumption: that natural language expressions are normally evaluated (processed) from left to right. Our theory is an extension of Shan's (2002) account of multiple-*wh* questions, combining continuations (Barker 2002) and dynamic type-shifting. Like other continuation-based analyses, but unlike most other treatments of crossover or superiority, our analysis is directly compositional (in the sense of, e.g., Jacobson 1999). In particular, it does not postulate a level of Logical Form or any other representation distinct from surface syntax. One advantage of using continuations is that they are the standard tool for modeling order-of-evaluation in programming languages. This provides us with a natural and independently-motivated characterization of what it means to evaluate expressions from left to right. We give a combinatory categorial grammar that models the syntax and the semantics of quantifier scope and *wh*-question formation. It allows quantificational binding but not crossover, in-situ *wh* but not superiority violations. In addition, the analysis automatically accounts for a variety of sentence types involving binding in the presence of pied piping, including reconstruction cases such as *Whose criticism of his$_i$ mother did each person$_i$ resent?*.

1. CROSSOVER AS A PROCESSING CONSTRAINT

Typically, a quantifier must precede a pronoun in order to bind it. Thus (1a) can mean that each person loves their own mother, but (1b) cannot (easily) be interpreted to mean that each person's mother loves them.

(1) a. Everyone$_i$ loves his$_i$ mother.
b. *His$_i$ mother loves everyone$_i$.

A processing explanation seems promising: if humans process sentences from left to right, and if a quantifier must be processed before any pronoun it binds, then arriving at the ungrammatical crossover interpretation in (1b) would require postponing the interpretation of the pronoun until encountering the quantifier that binds it. In general, allowing for such *crossover* interpretations would require the processing mechanism to postpone interpretation of pronouns indefinitely just in case a quantifier occurs later. Since doing so would place an extra load on memory, we expect the bound interpretation for (1b) to be at least dispreferred.

One challenge for a processing approach is distinguishing quantificational binding from quantifier scope. Sometimes quantifier scope also seems to show a left-to-right bias.

Date: February 19, 2005.

Thanks to Daniel Büring, Svetlana Godjevac, Pauline Jacobson, Gerhard Jäger, Chris Potts, Ivan Sag, Philippe Schlenker, Stuart Shieber, and our anonymous referees. We also profited greatly from discussions with audiences at: the Brown Workshop on Direct Compositionality; Rutgers; NYU; and at the 2004 ESSLLI, the Workshop on Syntax, Semantics and Pragmatics of Questions, and the Workshop on Semantic Approaches to Binding. The first author is supported by National Science Foundation Grants IRI-9712068 and BCS-0236592.

(2) a. A student called every professor.
b. Every professor called a student.

For instance, (2a) is more likely than (2b) to be interpreted to mean that the same student spoke with each professor. If quantifiers that are processed earlier take wider scope, then this preference is again expected.

If quantifier scope preferences and crossover arise from the same mechanism, we would expect them to behave similarly. But the left-to-right pattern is at best a preference for quantifier scope, while it is usually a requirement for quantificational binding. In fact, a quantifier can take scope over a pronoun that it cannot bind.

(3) *A student of his$_i$ called every professor$_i$.

In (3), the quantificational NP *every professor* can take scope over the subject, as long as the pronoun is taken to refer to some deictic individual (say, the Dean). So right-to-left scope, though awkward, is at least possible. But it is (almost) impossible to interpret the pronoun as bound by the quantifier, even when the quantifier takes scope over it.[1]

The puzzle for a processing story is that there is no obvious difference between quantificational binding and scope from a processing point of view: it seems as if any story that rules out inverse binding would rule out inverse scope too. Two obvious questions, then, are: if linear scope and the crossover constraint are both due to processing biases, why is the crossover constraint so much stronger? And: when a quantifier does take scope over a pronoun, what prevents the quantifier from binding the pronoun?

The traditional way out of this dilemma is to propose multiple stages for the derivation of a sentence, with binding and scoping taking place at different stages. Indeed, the very name "crossover" comes from Postal's (1971; p. 62) proposal for a general constraint on movement, roughly: an NP may not move across a coindexed pronoun. Reinhart's (1983) approach is an influential example, and Büring (2001, 2004) provides a recent analysis that uses the same basic strategy. The idea is to postulate two distinct levels of representation: first a level of surface structure at which binding is established by some syntactic relationship based on c-command, then a level of Logical Form at which quantifiers take their semantic scope. This way, even though a quantifier may raise at LF to take scope over a pronoun, it can still only bind the pronoun if it c-commands the pronoun from its surface position. The claim is that crossover is syntactic: quantificational binding depends on syntactic relations at the surface. In contrast, quantifier scope depends on relations at the level of Logical Form.

1.1. **A new solution based on continuations.** In contrast to QR-based accounts, we present in this paper a new account of quantifier scope and quantificational binding that does not postulate multiple stages of derivation or multiple levels of representation, and in fact does not even mention c-command. On our account, crossover is a consequence of the following processing default.

(4) Evaluate expressions from left to right.

[1]Postal (1993) (see also the discussion in Potts 2001) points out that crossover interpretations sometimes become better if *own* or *only* is added: ?*A student of his$_i$ own advisor contacted every professor$_i$*. This amelioration effect is poorly understood, and we will not discuss it further here.

We formally characterize what it means to evaluate expressions from left to right using CONTINUATIONS, a concept developed independently for describing the evaluation order of expressions in programming language semantics (see Section 7.1).[2]

The processing default (4) naturally follows from the commonplace assumption that people process (evaluate) expressions that they hear first before those they hear later. There is abundant evidence (e.g., Tanenhaus et al. 1990 and references there) that people begin to process language immediately, i.e., as soon as they hear the beginning of an utterance, and incrementally, building partial interpretations without waiting for the remainder of the utterance. Furthermore, incremental processing seems to immediately integrate information from all grammatical levels, including morphological, syntactic, semantic, and pragmatic. Of course, there are many ways that a grammatical framework could be consistent with incremental processing, but (4) embodies incrementalism in a natural and direct manner.

But if people process utterances from left to right, how could the quantifier in, say, (2a) be evaluated last in the sentence yet take semantic scope over the subject? It seems paradoxical, yet it is exactly how our system works.

1.2. **Continuations.** Since it is continuations that let us reason about order of evaluation, we need to say what they are. In fact, one of the larger purposes of this paper is to support the claim that continuations are a useful tool—perhaps an indispensable tool—for understanding natural language interpretation. We cannot possibly fit a complete discussion of continuations into this paper; nevertheless, we will provide some introductory comments.

Continuations were first studied explicitly in theoretical computer science. For instance, Plotkin (1975) uses continuations to characterize call-by-value versus call-by-name evaluation for the λ-calculus. More recently, in formal logic, Griffin (1990), Parigot (1992, 1993), and others use continuations to endow classical (as opposed to intuitionistic) proofs with computational content.

Our system uses *delimited* (or *composable*) continuations (Felleisen 1987, 1988; Danvy and Filinski 1989, 1992, 1990). Roughly, a delimited continuation is a prefix of the computational future of an expression, i.e., what is about to happen to it.

$$[\ldots \mathrm{A} \ldots]_{\mathrm{B}}\,, \quad \text{for example:} \quad [2 + [3 \times 4]_{\mathrm{A}} \times 5]_{\mathrm{B}}$$

The continuation of a subexpression A embedded in a larger expression B is a function from A's type to B's type. For linguistic intuition, consider the focus denotation of a containing expression when a subexpression is placed in focus.

$$[\![\textit{John saw } [\textit{everyone}]_F]\!]^{\mathrm{FOC}} = \lambda x.\, \mathbf{saw}\; x\, \mathbf{j}$$

The focus value of the sentence *John saw everyone* when *everyone* is in focus is the property of being seen by John. This property is the continuation of the NP *everyone* with respect to the sentence.

Continuations are immediately related to quantification: not coincidentally, the nuclear scope of the generalized quantifier denoted by *everyone* is the same property $\lambda x.\, \mathbf{saw}\, x\, \mathbf{j}$. In general, the nuclear scope of a quantificational expression is always its continuation with

[2]This default is similar to Phillips's (2003) Incrementality Hypothesis, which says "Sentence structures are built incrementally from left to right." Our work was developed entirely independently of his. Phillips's main concern is explaining conflicting constituency tests. On his proposal, incremental structure-building allows constituency relationships to change as the sentence structure is incrementally built. In contrast, we assume that any disambiguated sentence has exactly one constituent structure. For the most part Phillips discusses complementary phenomena to those discussed here, and he argues explicitly that a combinatory categorial grammar like the one we propose here is incompatible with his main hypothesis.

respect to the expression over which it takes scope. Thus the problem of accounting for in-situ quantification is equivalent to the problem of providing each expression with access to its own continuation. This is why a system that explicitly recognizes and manipulates continuations is ideally suited to reasoning about scope taking. This connection is developed by Barker (2002) and de Groote (2001).

Evaluation order enters the picture when more than one element manipulates continuations. In that case, the final result depends on which element's computational future is taken to include the others' evaluation. This amounts to deciding which element gets evaluated first. For instance, as we shall see, when an expression contains two quantificational noun phrases, order of evaluation corresponds to relative scope. More specifically, earlier evaluation corresponds to wider scope.

In programming-language terminology, scope-taking is a SIDE EFFECT of evaluating a quantificational NP.[3] All expressions serve a main semantic role (say, as a functor or an argument) and may also incur side effects. Other semantic phenomena besides scope-taking that we treat below as involving side effects include serving as a binder, being bound, and asking a question. Continuations let us keep track of side effects explicitly alongside the main computation.

1.3. **C-command versus linear order.** Besides doing without multiple levels of representation, there is a second respect in which our account differs significantly from the standard approach: on the standard account, the relationship between a potential binder and the element to be bound depends only on hierarchical notions based on c-command. On our account, the relationship is procedural: the binder must be evaluated before the bound element.

Purely hierarchical accounts of binding traditionally place themselves in opposition to analyses that involve some combination of c-command and linear order. Indeed, command relations were invented as part of Langacker's (1969) 'precede and command' constraint on pronominal anaphora. Since English syntax predominantly branches to the right, requiring that a binder c-command a bound pronoun coincides mostly with requiring that a binder precede the pronoun. Yet sometimes c-command and leftness arguably diverge. For instance, from examples such as *Near him$_i$, Dan$_i$ saw a snake*, Reinhart (1983) concludes that c-command is the only relevant constraint on binding.[4]

But even if anaphora in general allows a pronoun to precede its binder, the binding constraints specifically involved in weak crossover violations may nevertheless have a linear component. For instance, Higginbotham's (1980) analysis of crossover relies on a reindexing rule that is sensitive to linear order. More recently, Bresnan (1994, 1998) argues that there is crosslinguistic support for the importance of linear order in weak crossover patterns, and she cites a number of studies that suggest that linear order may be operative in English. Jäger (2001) also discusses this issue, and concludes that linear order is indeed a constraint on binding in English. Consequently, he builds a leftness condition into the

[3]The terms "side effect" and "effect" are equivalent in programming-language terminology.

[4]Reinhart discusses two additional classes of examples in which the pronoun precedes a quantificational binder but that space precludes us from discussing here. The first class involves apparently preposed phrases of various types (*Thinking about his$_i$ problems, everyone$_i$ got depressed*; cf. Higginbotham's (1980) PRO-gate examples such as *Having to make his$_i$ mother breakfast kept everyone$_i$ in the kitchen*). Reinhart suggests such examples may be due to "a later stylistic rule" that preposes after binding relation have been established. In our terms, this would be a case of delayed evaluation similar to the reconstruction cases discussed in Section 1.4 and Section 6.2. The second set of cases involves c-command from one argument of a verb into another (*?You may show his$_i$ file to each patient$_i$ who wants to see them*). Reinhart gives each example of this type one question mark, and we are content for now to categorize them as crossover violations.

binding rule in his account of binding and crossover. (We will discuss Jäger's analysis in more detail in Section 6.2.)

1.4. **Linear order versus evaluation order.** Yet, even restricting attention to quantificational binding, an unqualified leftness condition on quantificational binding is simply wrong. Perhaps the most prominent cases in point involve so-called reconstruction.

(5) Whose criticism of his$_i$ mother does every Englishman$_i$ resent?

In order to understand how standard accounts explain how the pronoun *his* comes to precede its binder in (5), imagine that the *wh*-phrase *whose criticism of his mother* is syntactically lowered ('reconstructed') into its logical position as the direct object of *resent* before binding takes place. After this reconstruction, the binder *every Englishman* c-commands (and precedes) the pronoun. Thus the standard analysis involves reconstruction, binding, QR, then interpretation.

Reconstruction examples such as (5) fall out automatically on our analysis. In this case, the general mechanism that accounts for displaced *wh*-phrases allows the *wh*-trace to, in effect, transmit binding power from the quantifier to the pronoun in the fronted *wh*-phrase. In processing terms, independently-motivated aspects of a construction may delay the processing of elements involved, including binders and pronouns, so that some right-to-left binding is okay and some left-to-right binding is not. We discuss binding by *wh*-phrases and reconstruction in more detail in Section 6.

1.5. **Varieties of crossover: explaining why weak crossover is weak.** The literature cross-classifies crossover into four varieties according to whether the binder or the pronoun is or is not embedded within a containing NP.[5] When the classification criterion depends on whether the pronoun is embedded, the varieties are called 'strong crossover' and 'weak crossover'.

(6) a. *He$_i$ likes every man$_i$. strong crossover
b. *His$_i$ mother likes every man$_i$. weak crossover

Examples like (6a), in which the pronoun is unembedded, are usually judged to be significantly worse than their embedded counterparts like (6b). Thus the crossover types are called 'strong' and 'weak' respectively.

When it is the embedding status of the binder that is in question, the varieties are called 'primary crossover' and 'secondary crossover' (Postal 1993).

(7) a. *He$_i$ likes every man$_i$. primary crossover
b. *He$_i$ likes every man$_i$'s mother. secondary crossover

When the quantificational NP is unembedded, we have primary crossover, and when the quantificational NP is embedded within a larger NP, we have secondary crossover. Some theories of quantification have great difficulty allowing embedded quantifiers to bind pronouns they don't c-command (see Heim and Kratzer 1998 (p. 234), Büring 2001, 2004, and Barker to appear for discussion). In our theory of quantificational binding, secondary binding (what Büring calls "binding out of DP", e.g., *Everyone$_i$'s mother loves him$_i$* or *Some person from every city$_i$ loves it$_i$*) falls out without special stipulation. Because our binding mechanism works equally well when a binder c-commands a bound pronoun and

[5]One type of crossover that we will not discuss at all is what Lasnik and Stowell (1991) call 'weakest crossover', which they argue involves binders that are not genuinely quantificational. In this paper, we will consider only cases in which the potential binder is either a quantificational NP or a *wh*-phrase (see immediately below), both of which qualify as genuinely quantificational in Lasnik and Stowell's terms.

when it merely take scope over a bound pronoun, we provide a unified account of primary and secondary weak crossover.

Our analysis targets weak crossover. But since every instance of strong crossover also qualifies (at least on our analysis) as weak crossover, we rule out strong and weak crossover alike. Nevertheless, there does seem to be a genuine and significant difference between the weak and strong varieties. Weak crossover examples are 'not so bad': they are easily interpretable, and a number of situations can ameliorate their ungrammaticality. Strong crossover examples, in contrast, are always completely ungrammatical.

Our analysis below suggests that there must be some factor involved in strong crossover that is independent of our explanation for weak crossover. For instance, only in strong crossover does the pronoun c-command its binder. Without speculating in detail why strong crossover is strong, we will assume that strong crossover violates some independent binding constraint in addition to whatever rules out weak crossover.

We do explain, however, why weak crossover is weak. Since weak crossover arises from a default preference for left-to-right processing, weak crossover examples should be interpretable if language users are able to resort to right-to-left processing when the situation demands it. And since our analysis formally locates the order of evaluation in a single rule (in fact, the rule governing scope-taking), it is easy to construct a variant rule that characterizes right-to-left evaluation. We perform this experiment in Section 7. With right-to-left evaluation, pronouns may precede their binders (and in-situ *wh*-phrases may intervene between a *wh*-phrase and its trace). Thus we predict that the effort required to interpret a weak crossover or superiority violation is exactly the effort required to exceptionally evaluate expressions from right to left.

2. Superiority and its relation to crossover

'Superiority' is the fact observed by Kuno and Robinson (1972; p. 474) that a *wh*-phrase cannot move across an in-situ *wh*-phrase.

(8) a. Who ate what?
b. *What did who eat __?

The term "superiority" comes from Chomsky's (1973) proposal for a general constraint that prohibits moving a phrase if a superior (roughly, a c-commanding) phrase could have been moved instead. As a result, (8b) is ruled out because the *wh*-phrase in object position moved when the superior *wh*-phrase in subject position could have moved instead.

Superiority bears an intriguing resemblance to crossover. Unlike crossover, however, the scope-taking element is an overtly moved *wh*-phrase rather than a covertly moved quantifier, and the crossed element is an in-situ *wh*-phrase rather than a bound pronoun.

For many years the dominant explanation for superiority attempted to reduce superiority to the ECP, which is a constraint on legitimate LF representations. Unfortunately, as Hornstein (1995; p. 124) and others point out, the ECP analysis of superiority has empirical shortcomings. Because the ECP distinguishes subjects from other argument positions, it can account for the basic contrast in (8), but it fails when neither *wh*-phrase is a subject.

(9) a. Who did Tom persuade __ to buy what?
b. *What did Tom persuade who to buy __?

Some Minimalist accounts of superiority recast it as a combination of Greed, Procrastinate, and Shortest Move, along with a transderivational constraint; others, as a combination of Attract with the Minimal Link Condition; still others, as an Optimality Theoretic interaction among a constraint requiring that Spec of CP be filled by a *wh*-phrase, a constraint

that *wh*-phrases raise for interpretation, and a general prohibition against movement. The details of these accounts are fairly intricate (see Dayal 2003 for a survey), and we will not discuss them further here, except to note that none of them connects superiority with crossover, and none of them contemplates any non-hierarchical relationship involving linear (or any other type of) order.

In contrast to the ECP and Minimalist accounts, Hornstein (1995) and Dayal (1996) (building on Chierchia's (1991; 1993) analysis of pair-list readings) separately argue that superiority reduces to weak crossover.

(10) Who$_i$ __ bought [*pro*$_i$ what]?
*What$_i$ did [*pro*$_i$ who] buy __$_i$?

For instance, on Hornstein's analysis, an in-situ *wh*-phrase denotes a function of type $\langle \mathtt{e}, \mathtt{e} \rangle$ whose argument corresponds to a silent pronoun (*pro*). If we stipulate that that *pro* inside an in-situ *wh*-phrase must be bound by the fronted *wh*-phrase, then superiority violations create a classic weak crossover configuration as shown in (10).

We propose not to reduce superiority to crossover (or vice versa). Rather, we will argue that crossover and superiority arise through somewhat different binding mechanisms, both of which are sensitive to order of evaluation, and it is this sensitivity that accounts for their mutual resemblance.[6] Unlike the ECP analysis, our account does not care whether the elements involved are subjects, and naturally generalizes to configurations like (9). Unlike Hornstein's and similar analyses, our account does without covert syntax entirely: there is no distinct level of LF, no covert pronominals, no syntactic indexing, and no difference in meaning between a raised *wh*-phrase and an in-situ *wh*-phrase. In fact, there is no stipulation at all beyond providing *wh*-phrases with a basic syntactic category and truth conditions, since superiority falls out from the independently-motivated theory of scope given in Section 3.

More specifically, just as crossover arises from the fact that binders must be evaluated before the pronoun that they bind, superiority arises from the fact that a raised *wh*-word can only bind its trace if the trace is evaluated before any in-situ *wh*-word. Given left-to-right evaluation, this means the trace must precede any in-situ *wh*-phrases. In other words, given left-to-right evaluation, an intervening in-situ *wh*-word fatally disrupts the binding relation between a raised *wh*-word and its trace.

3. A GENERAL THEORY OF SCOPE, QUANTIFICATIONAL BINDING, AND WH-QUESTIONS

In this section we present a general system for describing quantifier scope, quantificational binding, and *wh*-question formation. We say that the system is 'general' not because it is comprehensive, but because it contains no assumptions, mechanisms, or stipulations specific to crossover or superiority. That is, everything in this section is motivated by the simplest considerations of scope, binding, and *wh*-question formation. Nevertheless, we shall see that the basic analysis automatically explains weak crossover and superiority.

We present below a fragment of English expressed in the form of a combinatory categorial grammar. Our analysis has close similarities to, as well as important differences from, other combinatory categorial grammars (and systems equivalent to combinatory categorial grammars), including analyses due to Hendriks (1993), Jacobson (1999), and Steedman

[6] O'Neil (1993) also proposes that crossover and superiority be analyzed as special cases of a more general consideration, namely his Generalized Scope Marking Condition. However, his two special cases are individually stipulated and lack independent motivation.

(2000), as well as certain type-logical accounts discussed by Dowty (1991). Like Hendriks's and Steedman's, our system handles scope. Like Jacobson's account and those discussed by Dowty, it handles quantificational binding. In addition, it extends naturally to handle single- and multiple-*wh* questions. We have also implemented the ideas in this paper within a type-logical framework (Barker and Shan to appear).

Though we are indebted to the work just mentioned, the main ideas below are not developments or elaborations of those accounts. Rather, our approach is more directly related to work in the computer science literature concerning type shifting (Hindley and Seldin 1986), continuations (Plotkin 1975; Meyer and Wand 1985; inter alia), and composable control (Felleisen 1987, 1988; Danvy and Filinski 1989, 1992, 1990; inter alia) ('control' in the computer science sense of order-of-evaluation).

It is particularly intriguing to note the resemblance between our approach and Jacobson's variable-free account of binding, as formulated in, e.g., Jacobson 1999. For example, one crucial element of her account is that pronouns denote the identity function, and that is true of our account as well. Unlike Jacobson, however, we did not start with a desire to eliminate variables and then build a system that made that desire come true; rather, our starting point was the idea of using continuations to describe scope, which required figuring out how pronouns and binding could fit into a continuation-based interpretation strategy. As we will see below, this naturally leads to a system in which pronouns denote the identity function. Thus the fact that both Jacobson's and our approaches arrive at the same idea of pronouns as denoting the identity function is not a case of theoretical copying, but an instance of homologous convergence.

3.1. **Syntactic categories and semantic types.** In the traditions of Montague grammar, categorial grammar, and type-logical grammar, we keep the mapping between syntactic category and semantic type as transparent and direct as possible. Therefore we have two basic syntactic category labels, `e` and `t`, whose semantic types are individuals and truth values. Complex syntactic categories have the form $A\backslash B$, B/A, $A\mathbin{\backslash\!\!\backslash}B$, $B/\!\!/A$, $A\triangleright B$, or $A?B$, where A and B are again syntactic categories. In each case, the semantic type is $\langle\alpha,\beta\rangle$ (the class of functions from type α to type β), where α is the type of expressions in category A, and β is the type of expressions in category B.

We will motivate and comment on each of the connectives separately in due course. For now, / and \ describe basic function/argument structure; $/\!\!/$ and $\backslash\!\!\backslash$ describe continuations; $\triangleright$ encodes binding; and ? builds categories for questions. Thus in addition to making syntactic distinctions, the various connectives also track conceptual differences, distinguishing properties from continuations, questions, and so on.

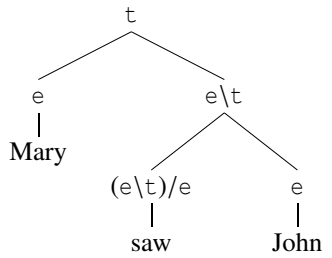
The basic connectives / and \ govern syntactic combination in the usual way:[7]

(11)

Syntax:	A	followed by	$A\backslash B$	yields	B
Semantics:	x		f		fx
Syntax:	B/A	followed by	A	yields	B
Semantics:	f		x		fx

Then if proper names such as *John* and *Mary* have category `e`, and transitive verbs have category `(e\t)/e`, we have the following derivation.

[7]The first half of (11), for backward function application, is redundant (derivable, in fact) given the Lift rule introduced shortly below, but we include it for clarity.



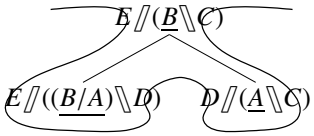
The semantic value is (**saw j**) **m**, which is (the extension of) the proposition that Mary saw John.

3.2. **Continuations and quantifier scope.** Continuations have types of the form $A \backslash\!\!\backslash B$. Since by hypothesis quantificational expressions are functions on continuations, quantificational expressions have types of the form $C /\!\!/ (A \backslash\!\!\backslash B)$. This is interpreted as the type of an expression that functions locally (i.e., syntactically) as if it had category A, but that takes scope over an expression of type B to create an expression of type C. For instance, *everyone* will have type $\mathtt{t} /\!\!/ (\mathtt{e} \backslash\!\!\backslash \mathtt{t})$: it functions locally as an NP but takes scope over a clause to create a new clause. This category has the same semantic type $(\langle\langle \mathtt{e}, \mathtt{t}\rangle, \mathtt{t}\rangle)$ as an (extensional) generalized quantifier. Indeed, Montague's conception of NPs as denoting generalized quantifiers is a special case of adding continuations uniformly throughout the grammar (Barker 2002).

The following type-shifting rules allow continuations to combine with other expressions and each other.

(12)			
	Lift	$(B/(A \backslash B))/A$	$\lambda x.\, \lambda F.\, Fx$
	Up	$(B /\!\!/ (A \backslash\!\!\backslash B))/A$	$\lambda x.\, \lambda F.\, Fx$
	Down	$A/(A /\!\!/ (\mathtt{t} \backslash\!\!\backslash \mathtt{t}))$	$\lambda F.\, F(\lambda x.\, x)$
	Scope	$((E /\!\!/ (B \backslash\!\!\backslash C))/(D /\!\!/ (A \backslash\!\!\backslash C)))/(E /\!\!/ ((B/A) \backslash\!\!\backslash D))$	$\lambda L.\, \lambda R.\, \lambda \kappa.\, L(\lambda l.\, R(\lambda r.\, \kappa(lr)))$

The crucial rule is the Scope rule.[8] This rule combines two expressions with side effects.



The underlined parts buried in these categories show normal categorial cancellation: B/A followed by A yields B. Thus the underlined parts indicate the main computation, and the additional decorations indicate side effects.

The curve connects side effects linked by the rule: it connects E on the mother category with E on the left daughter, D with D, and C with C. The curve shows how the rule imposes left-to-right evaluation: since the mother and the left daughter share the result type E, we

[8] Some notes to help make sense of the three simpler type-shifting rules.

Lift is the same rule given by Partee and Rooth (1983), as well as pretty much everyone who allows any type-shifting, including Partee (1987), Moortgat (1997), Jacobson (1999), Steedman (2000), inter alia.

Up performs the corresponding lifting for continuations. In the computer science literature on monads (e.g., Wadler 1992), Up it is known as 'unit', so U can stand for either 'Up' or 'Unit'. In Barker and Shan's type-logical implementation (to appear), / and \ as well as // and \\ characterize two logical modes, related such that Lift and Up are theorems.

What goes up must eventually come down, so Down complements Up. Down is analogous to Partee's LOWER (1987) and Chierchia's ↓ (1995; p. 86). The semantic values for each of these rules is simply the Curry-Howard labeling of the appropriate derivation (Moortgat 1997; Barker 2004; Barker and Shan to appear).

know the left daughter takes priority (i.e., scopes) over the right daughter. (In Section 7 below we show what Scope would look like if it imposed right-to-left evaluation order instead.)

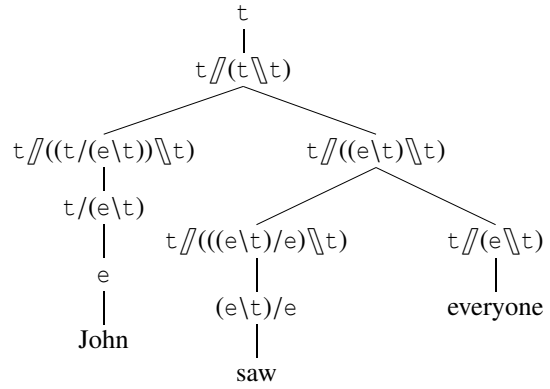
An example will show how a quantificational NP in direct object position takes semantic scope over an entire clause. First, we need lexical items that exploit the presence of continuations.

(13) everyone $\mathtt{t}/\!\!/(\mathtt{e}\backslash\!\!\backslash\mathtt{t})$ $\lambda\kappa.\,\forall x.\,\kappa x$
someone $\mathtt{t}/\!\!/(\mathtt{e}\backslash\!\!\backslash\mathtt{t})$ $\lambda\kappa.\,\exists x.\,\kappa x$

These lexical items give rise to the following derivation:

(14) John saw everyone.
(D ((S (U (L john))) ((S (U saw)) everyone)))
$= \forall x.\,(\textbf{saw}\ x)\ \textbf{j}$

which justifies the following category judgments:



Note that the quantificational expression *everyone* does not undergo movement. Furthermore, the verb phrase *saw everyone* is a constituent in the final derivation, just as in the surface syntax. In general, this system strictly maintains direct compositionality: anything that is a constituent in the syntax is a constituent in the semantics as well, and vice versa.

Many of the crossover examples below will contain possessive constructions. For simplicity, we will always use a relational head noun, which we assume denotes a function from individuals to individuals, so we can treat the possessive clitic *'s* as semantically null. For instance, given that a relational noun such as *mother* has category $\mathtt{e}\backslash\mathtt{e}$, we have:

(15) John's mother left.
(john mother) left
$=$ **left**(**mother j**)

The grammar automatically handles quantificational NPs embedded arbitrarily deep within a possessive phrase.

(16) Everyone's mother left.
(D ((S ((S (U L)) ((S ((S (U L)) everyone)) (U mother)))) (U left)))
$= \forall x.\,$**left**(**mother** x)

Example derivations in which the quantificational NP binds a pronoun without c-commanding it appear below.[9]

[9]The type-shifting rules here are chosen to be individually simple, but they give rise to somewhat complex derivations. We provide online at semanticsarchive.net complete details for each of the derivations in this paper,

3.3. **Continuation levels and quantifier scope ambiguity.** When a sentence contains more than one quantificational NP, the simplest analysis delivers linear scope. In other words, the system has a built-in left-to-right bias, which means that, all else being equal, expressions on the left outscope expressions to their right.

(17) Someone saw everyone.
(D ((S ((S (U L)) someone)) ((S (U saw)) everyone))) $= \exists y.\, \forall x.\, (\mathbf{saw}\ x)\ y$

To get inverse scope, it is necessary to lift *everyone* to operate at a higher CONTINUATION LEVEL.

We can informally define continuation levels as follows. Let a 'pure' expression be an expression without side effects, that is, an expression whose category contains neither $\backslash\!\backslash$ nor $/\!/$. Call this continuation level 0. Applying Up to a pure expression once produces one $\backslash\!\backslash$ and a matching occurrence of $/\!/$. Call this continuation level 1. In particular, a name like *John* with category $\mathtt{e}$ is pure, so (Up *John*) has category $\mathtt{t}/\!/(\mathtt{e}\backslash\!\backslash\mathtt{t})$, which is at continuation level 1. In the derivation of *John saw everyone* in (14) above, Up must apply to *John* before *John* can combine with the quantificational verb phrase. In general, there is no way to combine two expressions that have different continuation levels.

Lifting an expression to a higher level can allow it to take wider scope, including scoping over elements that precede it. Thus lifting to higher levels of continuations can overcome the bias towards linear scope. But lifting with L alone is not sufficient.

(18) a. U everyone $(A/\!/((\mathtt{t}/\!/(\mathtt{e}\backslash\!\backslash\mathtt{t}))\backslash\!\backslash A))$ $\lambda F.\, F(\lambda\kappa.\, \forall x.\, \kappa x)$
b. (S(U U)) everyone $(\mathtt{t}/\!/((A/\!/(\mathtt{e}\backslash\!\backslash A))\backslash\!\backslash\mathtt{t}))$ $\lambda F.\, \forall x.\, F(\lambda\kappa.\, \kappa x)$

Since the lexical category of *everyone* is at level one to begin with, these derived categories are both at level two. Simply lifting *everyone* with U, as in (18a), turns a one-story house into a two-story house in the obvious way: by adding a new story on top of the old one. In contrast, lifting with the complex type-shifter (S(U U)), as in (18b), adds a second story in a less obvious way: by jacking up the first story and placing a new level underneath. In each case, the new level corresponds to the category variable A. The fact that the quantificational element *everyone* takes wider scope in the second derivation is reflected in the syntactic categories as well as the semantic denotations.

Both expressions in (18) are at continuation level two. Nevertheless, *everyone* performs quantification at level one in the first derivation, but at level two in the second derivation. It is the level at which a quantifier takes effect that determines its scope relative to other elements in the expression. In general, quantifiers that take effect on the same level will always take relative scope according to evaluation order, while quantifiers that take effect at higher levels take wide scope over quantifiers at lower levels. In *someone saw everyone*, if *everyone* takes effect at level two while *someone* takes effect at level one, inverse scope results:

(19) Someone saw everyone.
(D((S(U D))((S(U(S((S(U L)) someone))))((S(U U))((S(U saw)) everyone)))))
$= \forall x.\, \exists y.\, \mathbf{saw}\ y\ x$

as well as a working parser in standard Scheme that the reader can use to try out different examples or even different rules. The parser gives categories and denotations for all constituents, as well as multiple analyses for examples that the fragment predicts are ambiguous. As far as we know, this is the first practical parser to treat crossover and superiority.

As expected from the discussion of (18) above, if the underlined (S (U U)) were replaced with just U, the resulting derivation would once again produce a linear scope interpretation. The way to tell that this is a two-level derivation (without calculating categories) is to observe that there are two occurrences of the Down operator near the left end of the derivation. If two lowering operations are required to reach the ground level, the derivation must have risen to level two.

Multiple continuation levels were first investigated in the context of programming languages by Sitaram and Felleisen (1990) and Danvy and Filinski (1990).

3.4. **Pronouns and binding.** The type-shifting operators introduced above constitute basic machinery in the grammar to handle side effects in general. The lexical entries for *everyone* and *someone* in (13) incur the side effect of quantification. Building on this foundation, we can handle quantificational binding by adding a single general type-shifting operator (for the side effect of binding), along with suitable lexical entries for pronouns (for the side effect of being bound).

As in most variable-free analyses, the presence of bindable pronouns within a clause affects its syntactic category and its semantic type.

(20) a. John left. $\mathtt{t}$
b. He left. $\mathtt{e} \triangleright \mathtt{t}$

The sentence *John left* has category $\mathtt{t}$ and denotes the closed proposition that John left. The pronominal version, *He left*, is not closed: it depends for its value on the choice of an individual for the referent of *he*. On the standard treatment, this dependence is handled by relativizing the denotation of an expression to an assignment function that specifies the values of pronouns. In Jacobson's work and here, the open nature of the pronominal sentence is encoded more directly, both in the category structure and in the denotation of the expression itself: instead of denoting a proposition relative to an assignment, (20b) denotes a function from individuals to propositions, with semantic type $\langle \mathtt{e}, \mathtt{t} \rangle$: the function from possible choices for the value of *he* to the proposition that the chosen individual left.

In order for this approach to work, a pronoun must be capable of affecting the status of a clause containing it. But another way of saying this is that the pronoun must take scope over that constituent. In other words, the influence that a pronoun has on the category and denotation of a clause containing it is a side-effect of evaluating the pronoun, and so we can use the same mechanism that enables a quantifier to take scope.

Given this strategy, we can deduce the syntactic category of a pronoun as follows. Since the pronoun has side effects, its category will involve continuations, and will have the form $C \mathbin{/\!\!/} (A \mathbin{\backslash\!\!\backslash} B)$. Clearly, pronouns function locally as noun phrases, so in our system, $A = \mathtt{e}$. And since pronouns can take scope over a clause, we can have $B = \mathtt{t}$. When a normal quantificational NP such as *everyone* takes scope over a clause of category $\mathtt{t}$, the result is the same category ($C = \mathtt{t}$); but the result of embedding a pronoun within a clause is not a proposition, but a function from individuals to propositions. We have seen a variety of categories with the appropriate semantic type, including $\mathtt{t}/\mathtt{e}$, $\mathtt{e}\backslash\mathtt{t}$, $\mathtt{t} \mathbin{/\!\!/} \mathtt{e}$, and $\mathtt{e} \mathbin{\backslash\!\!\backslash} \mathtt{t}$. But since a clause containing a pronoun is not the same thing either syntactically or conceptually as any of these other expression types, we need a new connective, $\triangleright$. This yields $(\mathtt{e} \triangleright \mathtt{t}) \mathbin{/\!\!/} (\mathtt{e} \mathbin{\backslash\!\!\backslash} \mathtt{t})$ as the category of a pronoun embedded within a clause of category $\mathtt{t}$ (this category will be slightly generalized below).

The denotation of the pronoun follows directly. The semantic type of the category $(\mathtt{e} \triangleright \mathtt{t}) \mathbin{/\!\!/} (\mathtt{e} \mathbin{\backslash\!\!\backslash} \mathtt{t})$ is $\langle \langle \mathtt{e}, \mathtt{t} \rangle, \langle \mathtt{e}, \mathtt{t} \rangle \rangle$: a function from sets of individuals to sets of individuals. The semantic argument to the pronoun will be a function from individuals to truth values,

with category $\mathtt{e} \backslash\!\backslash \mathtt{t}$. This is the continuation of the pronoun with respect to the clause. For example, in the sentence *John saw him*, the continuation of the pronoun *him* is $\lambda x.\, \mathbf{saw}\ x\ \mathbf{j}$, the property of being seen by John. But this is exactly the function from individuals to truth values we would like to assign as the meaning of the pronoun-containing sentence as a whole. Hence the denotation of the pronoun must be the identity function, $\lambda\kappa.\, \kappa$.

Thus whether in subject or object position, whether embedded or not, pronouns reach outwards to add a layer of functional dependence to the denotation of the larger clause.

(21)
a. He left. $\mathtt{e} \triangleright \mathtt{t}$: $\lambda x.\, \mathbf{left}\ x$
b. John saw him. $\mathtt{e} \triangleright \mathtt{t}$: $\lambda x.\, \mathbf{saw}\ x\ \mathbf{j}$
c. Everyone saw him. $\mathtt{e} \triangleright \mathtt{t}$: $\lambda x.\, \forall y.\, \mathbf{saw}\ x\ y$
d. John thought she left. $\mathtt{e} \triangleright \mathtt{t}$: $\lambda x.\, \mathbf{thought}(\mathbf{left}\ x)\ \mathbf{j}$

This ability of an embedded constituent to gain access to and control over material that surrounds it is the hallmark of continuations.

We need to generalize the category of pronouns somewhat. So far, we have considered only clauses that contain a single pronoun. But a sentence can contain several bindable pronouns, each one of which adds a new layer of functional dependence. Thus a bindable pronoun may need to take scope over a clause that already contains another bindable pronoun. The generalized category for a pronoun, then, is $(\mathtt{e} \triangleright A) /\!\!/ (\mathtt{e} \backslash\!\backslash A)$: the pronoun takes any noun phrase continuation (category $\mathtt{e} \backslash\!\backslash A$) and turns it into a phrase whose denotation depends on specifying an individual (category $\mathtt{e} \triangleright A$). Put slightly differently, whatever category the expression would have been, the result of inserting a pronoun will be to prefix '$\mathtt{e} \triangleright$' to the final syntactic category, and a layer of functional dependence to the denotation.

Another route to the same analysis is to think of a pronoun as a lifted proper name that is missing one crucial piece of information, namely, the identity of the individual involved.

(22)
a. Proper name john $\mathtt{e}$ $\mathbf{j}$
b. Lifted proper name U john $A /\!\!/ (\mathtt{e} \backslash\!\backslash A)$ $\lambda\kappa.\, \kappa\mathbf{j}$
c. Pronoun he $(\mathtt{e} \triangleright A) /\!\!/ (\mathtt{e} \backslash\!\backslash A)$ $\lambda\kappa.\, \kappa$

The lifted proper name (22b) asks for its continuation (κ) and then feeds to it the individual denoted by *John* (i.e., applies κ to $\mathbf{j}$). The pronoun, also, asks for its continuation, but has no individual to feed to it, and so simply gives back the unsatisfied continuation to whoever might be prepared to supply it with a value.

Supplying such a value, of course, amounts to binding the pronoun. A binder reverses the effect of a pronoun: instead of adding a layer of functional dependence, the binder seeks an expression whose category already exhibits a layer of functional dependence (i.e., has form $\mathtt{e} \triangleright A$) and resolves that dependence (resulting in A). Of course, a binder must first have some kind of individual in view to serve as the binder; any expression of category $\mathtt{e}$ will do.

(23) Bind = $(B /\!\!/ (\mathtt{e} \backslash\!\backslash (\mathtt{e} \triangleright A)))\ /\ (B /\!\!/ (\mathtt{e} \backslash\!\backslash A))$ $\lambda X.\, \lambda\kappa.\, X(\lambda x.\, (\kappa x)x)$

This type-shifter targets a binder of category $B /\!\!/ (\mathtt{e} \backslash\!\backslash A)$. For instance, a quantificational NP such as *everyone*, with category $\mathtt{t} /\!\!/ (\mathtt{e} \backslash\!\backslash \mathtt{t})$, can be a binder. What Bind does is shift the binder to category $(B /\!\!/ (\mathtt{e} \backslash\!\backslash (\mathtt{e} \triangleright A)))$: the same as before, except that the shifted category now expects a bindable pronoun ('$\mathtt{e} \triangleright$') in its scope. Once the semantics receives the binder individual x, it makes a copy of x to give to the waiting pronoun, and delivers the original to the binder's continuation $((\kappa(x_{\text{original}}))(x_{\text{copy}}))$.

For example, we can use Bind to type-shift *everyone* into a quantificational NP that simultaneously binds and quantifies.

(24) B everyone = $\mathtt{t}/\!\!/(\mathtt{e}\backslash\!\!\backslash(\mathtt{e} \triangleright \mathtt{t})) : \lambda\kappa.\, \forall x.\, \kappa x x$

The normal *everyone* in (13) expects a continuation of type $\mathtt{e}\backslash\!\!\backslash\mathtt{t}$ and returns a truth value. The binding version of *everyone*, computed in (24) as *B everyone*, also returns a truth value, but expects a continuation that has been augmented (by some pronoun) with an extra argument of type e.

Given *thought* of category $(\mathtt{e}\backslash\mathtt{t})/\mathtt{t}$, we can now derive the following two examples:[10]

(25) Everyone$_i$ thought he$_i$ left
(D ((S ((S (U L)) (B everyone))) ((S (U thought)) ((S ((S (U L)) he)) (U left)))))
= $\forall x.$ **thought** (**left** x) x

(26) Everyone$_i$'s mother thought he$_i$ left
(D ((S ((S (U L)) ((S ((S (U L)) (B everyone))) (U mother)))) ((S (U thought)) ((S ((S (U L)) he)) (U left)))))
= $\forall x.$ **thought** (**left** x) (**mother** x)

Just as in (16), the operators as given generalize to binding out of a possessive NP without further stipulation. Note in (26) that *everyone* need not c-command the pronoun to bind it. LF-based theories of binding typically need some principle which says in effect that if A can bind B, and A contains C, and C can take scope over B, then C can bind B (e.g., Ruys 2000, p. 517). No such stipulation is required here.

A binder must take effect at the same continuation level as the pronoun in order for binding to occur. This fact is crucial for our account of crossover in Section 4. In the presence of a quantifier, a deictic interpretation for a subsequent pronoun requires at least two continuation levels, one for the quantifier to take scope, and a separate higher level for the pronoun to gain access to the sentence as a whole.

(27) Everyone$_i$ thought he$_j$ left
(D ((S (U D)) ((S (U (S ((S (U L)) everyone)))) ((S (U (S (U thought)))) ((S (U U)) ((S ((S (U L)) he)) (U left)))))))
= $\mathtt{e} \triangleright \mathtt{t} : \lambda y.\, \forall x.$ **thought** (**left** y) x

The presence of two applications of the Down operator shows that there are two continuation levels here. The final result is of category $\mathtt{e} \triangleright \mathtt{t}$, as expected of any sentence with a single unbound pronoun.

When there is more than one pronoun (e.g., *He saw her*), the resulting denotation will have two layers of functional dependence. In order for the two pronouns to keep out of each other's way, they may need to seek out different continuation levels at which to be bound (e.g., *Every woman$_i$ thought every man$_j$ thought he$_j$ saw her$_i$*). In effect, what most binding theories encode using indices on pronouns corresponds to a continuation level here. As a result, we do not need to assume that pronouns are lexically polysemous: a single denotation will suffice, in combination with the independently needed category-shifting rules for scope manipulation. Nor is it necessary to provide indices for binders; in effect, each anaphoric dependency is indexed by the continuation level at which the binding occurs. Put more simply, on this account, coindexation is a matter of relative scope.

3.5. ***Wh*-phrases, in-situ and raised.** An in-situ *wh*-phrase transforms the sentence in which it is embedded into a question. More precisely, it functions locally as an NP, takes

[10]Here and throughout, subscripts in example sentences serve purely to indicate the intended interpretation, and have no counterpart in the formal system.

scope over a clause, and produces a result that is functionally dependent on an individual-type argument.

(28) a. who $(\mathtt{e}\ ?\ A)/\!\!/(\mathtt{e}\backslash\!\!\backslash A)$ **who**
b. what $(\mathtt{e}\ ?\ A)/\!\!/(\mathtt{e}\backslash\!\!\backslash A)$ **what**

The syntactic category of *who* and *what* closely resembles that of *he* and *she*, with one difference: the type of the final result returned is $\mathtt{e}?\mathtt{t}$ rather than $\mathtt{e} \triangleright \mathtt{t}$. The type $\mathtt{e}?A$ is the type of a question in which an NP has been questioned, i.e., that 'asks for' an individual.[11]

For example, we can derive a simple *wh*-question, a multiple-*wh* question, and a question in which the *wh*-word is embedded within a possessive as follows.

(29) Who left?
(D ((S ((S (U L)) who)) (U left)))
$= \mathtt{e}\ ?\ \mathtt{t} : \mathbf{who}(\lambda x.\, \mathbf{left}\ x)$

(30) Who bought what?
(D ((S ((S (U L)) who)) ((S (U bought)) what)))
$= \mathtt{e}\ ?\ (\mathtt{e}\ ?\ \mathtt{t}) : \mathbf{who}(\lambda x.\, \mathbf{what}(\lambda y.\, \mathbf{bought}\ y\ x))$

(31) Whose mother left?
(D ((S ((S (U L)) ((S ((S (U L)) whose)) (U mother)))) (U left)))
$= \mathtt{e}\ ?\ \mathtt{t} : \mathbf{who}(\lambda x.\, \mathbf{left}(\mathbf{mother}\ x))$

Just as for pronouns, the generality of the lexical entry for the *wh*-pronoun allows it to occur as the only *wh*-phrase in a sentence, transforming a proposition into a single-*wh* question; or else in the presence of additional *wh*-words (here, transforming a single-*wh* question into a multiple-*wh* question). And just as for quantificational possessors, the scoping mechanism allows *wh*-possessors embedded arbitrarily deeply (e.g., *whose mother's friend's dog*, or *whose mother* in (31)) to take effect at the top level of the sentence without special stipulation.

In order for a *wh*-phrase to occur in clause-initial position (which we will inaccurately call "raised" or "fronted" in deference to universal usage), we now provide a pair of type-shifters Trace and Question:

(32) a. Trace $B/\!\!/B$ $\lambda\kappa.\,\kappa$
b. Question $((\mathtt{e}\ ?\ A)/B)/((\mathtt{e}\ ?\ A)/\!\!/B)$ $\lambda X.\,X$

In the examples in this section and the next two, we can choose $B = (\mathtt{e}\backslash\!\!\backslash A)$, so that Trace is instantiated as type $(\mathtt{e}\backslash\!\!\backslash A)/\!\!/(\mathtt{e}\backslash\!\!\backslash A)$. This instantiation is similar to the pronoun denotations given above in that it records on the result type the fact that an individual-type argument needs to be supplied before the proposition is complete. (The discussion of reconstruction in Section 6.2 will consider a different instantiation of the general Trace shifter.)

The Question operator converts a *wh*-phrase into the kind of phrase that can bind a trace. Because this operator is specific to *wh*-expressions, we call it Q, though in general, other

[11] On our analysis, multiple-*wh* questions have a different semantic type from single-*wh* questions. This reflects the fact that an answer to a multiple-*wh* question can be of a different type from an answer to a single-*wh* question (for instance, a list of pairs of individuals rather than a list of individuals). On other analyses, notably Karttunen's, multiple-*wh* questions have the same semantic type as single-*wh* questions (namely, a set of propositions). As a referee points out, Karttunen's approach simplifies the analysis of predicates that embed questions, since they never seem to distinguish between single- versus multiple-*wh* questions. It is perfectly feasible to accommodate a Karttunen-style analysis if desired; however, we have chosen to emphasize the parallels between *wh*-phrases and pronouns.

types of expression can also front in English. In order to simulate auxiliary inversion, we provide a lexical entry for *did* that has no syntactic or semantic effect.

(33) did $\quad \mathtt{t/t} \quad \lambda p.\, p$

This provides for matrix questions containing one raised *wh*-phrase along with any number of in-situ *wh*-phrases:

(34) Who __ left?
((Q who) (D ((S ((S (U L)) T)) (U left))))
$= \mathtt{e} \,?\, \mathtt{t} : \mathbf{who}(\lambda x.\, \mathbf{left}\ x)$

(35) What did Mary buy __?
((Q what) (D ((S (U did)) ((S (U (L mary))) ((S (U buy)) T)))))
$= \mathtt{e} \,?\, \mathtt{t} : \mathbf{what}(\lambda x.\, \mathbf{buy}\ x\ \mathbf{m})$

(36) Who did Mary give __ what?
((Q who) (D ((S (U did)) ((S (U (L mary))) ((S ((S (U give)) T)) what)))))
$= \mathtt{e} \,?\, (\mathtt{e} \,?\, \mathtt{t}) : \mathbf{who}(\lambda x.\, \mathbf{what}(\lambda y.\, \mathbf{give}\ x\ y\ \mathbf{m}))$

In addition to the in-situ analysis of *Who left?* given above in (29), there is now an analysis in which *who* has been string-vacuously displaced to the left, shown in (34).[12] This analysis of English *wh*-phrases automatically rules out any sentence in which more than one *wh*-word has been fronted (e.g., **What who did Mary say __ bought __?*).

3.6. **Assessment of the fragment.** We have introduced a combinatory categorial grammar with seven type-shifting operators: Lift, Up, Down, Scope, Bind, Trace, and Question. Given suitable lexical categories, this grammar is expressive enough to describe unbounded quantifier scope displacement, quantifier scope ambiguity, quantificational binding, scoping and binding out of NP, and single- and multiple-*wh* question formation. In particular, (26) shows how a quantifier can bind a pronoun without c-commanding it. Most other accounts require special stipulations concerning quantifier raising out of NP, such as May's Logical Form (1985), Ruys's transitivity property (2000), or Büring's functions on situations (2001, 2004). The fact that our system handles binding out of NP without any stipulation enables us to give a unified account of primary and secondary crossover.

When assessing the complexity of this fragment, it is important to bear in mind what is not present: there is no movement, no level of Logical Form, no variable assignment functions, and no constraints on category shifting. There is nothing in the fragment beyond the bare minimum necessary to treat scope, binding, and question formation. Nevertheless, the remaining sections will show that crossover and superiority fall out without any additional stipulation.

4. Explaining crossover

Now we are ready to show how the analysis just presented accounts for crossover. Let us compare the following two examples:

(37) a. Everyone$_i$ saw his$_i$ mother.
b. *His$_i$ mother saw everyone$_i$.

The grammatical example (37a) is perfectly straightforward to derive.

[12] Like the Bind operator, the Trace and Question operators are formally indistinguishable from any of the other type-shifting operators such as Lift, Up, Down, and Scope, and like those operators, Trace and Question apply freely without constraint. Purely as a visual aid, we have indicated in the example sentences the linear position in which Trace has been applied in the corresponding derivation.

(38) Everyone$_i$ saw his$_i$ mother.
(D ((S ((S (U L)) (B everyone))) ((S (U saw)) ((S ((S (U L)) his)) (U mother)))))
= $\forall x.$ **saw** (**mother** x) x

The crossover violation in example (37b), however, has no analysis with the interpretation shown. The reason is as follows. In this fragment, as in every semantically coherent theory, a quantifier must take scope over a pronoun in order to bind it. Therefore, before a quantifier can bind a preceding pronoun, it must lift to a higher continuation level, so as to take scope over that pronoun. Following this strategy as in the inverse-scope example (19), we can get as far as applying Down once to a two-level derivation in which *everyone* takes inverse scope over *his mother*:

(39) ((S(U D))((S(U(S((S(U L))((S((S(U L))his))(U mother))))))(S(U U))((S(U saw))(B everyone)))))
= $\mathtt{t}/\!\!/((\mathtt{e}\rhd\mathtt{t})\backslash\!\!\backslash(\mathtt{e}\rhd\mathtt{t})) : \lambda\kappa.\forall x.(\kappa(\lambda y.(\textbf{saw}\ x)(\textbf{mother}\ y)))x$

If we could feed the identity continuation to this denotation—in other words, if we could apply Down to it once more—we would be able to derive the crossover-violating meaning $\forall x.(\textbf{saw}\ x)(\textbf{mother}\ x)$. But Down only matches a category of the form $A/\!\!/(\mathtt{t}\backslash\!\!\backslash\mathtt{t})$. Since $(\mathtt{e}\rhd\mathtt{t})\backslash\!\!\backslash(\mathtt{e}\rhd\mathtt{t})$ does not unify with $\mathtt{t}\backslash\!\!\backslash\mathtt{t}$, Down cannot apply, and the derivation fails.

To make more precise what it means for a derivation to fail, say that a derivation of a sentence or a question is successful if the expression as a whole is at the zeroth continuation level. For practical purposes, this means that the final category contains no occurrences of the connective '$\backslash\!\!\backslash$'. Thus the main successful categories provided by the grammar are $\mathtt{t}$, $\mathtt{e}\rhd\mathtt{t}$, $\mathtt{e}\,?\,\mathtt{t}$, $\mathtt{e}\,?\,(\mathtt{e}\,?\,\mathtt{t})$, and so on. For instance, the grammar classifies the sentence *Mary left* as belonging both to the category $\mathtt{t}$ and the category $\mathtt{t}/\!\!/(\mathtt{t}\backslash\!\!\backslash\mathtt{t})$; but since $\mathtt{t}/\!\!/(\mathtt{t}\backslash\!\!\backslash\mathtt{t})$ is at the first continuation level (it contains a $\backslash\!\!\backslash$), only the derivation that justifies inclusion in the category $\mathtt{t}$ counts as successful.

In short, the category of Down forces any quantifier and any pronoun to take effect at the same level if the former is to bind the latter. Because a quantifier cannot take effect at both a higher level and the same level, the grammar fails to generate the interpretation indicated in (37b), as desired.

Furthermore, because our account predicts that binding will be possible whenever a quantifier takes scope over a subsequent pronoun (more precisely: a pronoun evaluated later), we can generate examples in which the quantificational NP fails to c-command the bound pronoun in surface structure. In addition to standard examples of binding out of NP such as (26), our analysis generalizes to other configurations of the *We will sell no wine before its time* type. To illustrate, we will consider a case of binding into an adjunct. All we need is a basic lexical entry for adjunct-creating prepositions; here, we assume that *on* is of category $((\mathtt{e}\backslash\mathtt{t})\backslash(\mathtt{e}\backslash\mathtt{t}))/\mathtt{e}$.

(40) Mary phoned everyone$_i$ on his$_i$ birthday.
(D ((S (U (L mary))) ((S ((S (U L)) ((S (U phoned)) (B everyone)))) ((S (U on)) ((S ((S (U L)) his)) (U birthday))))))
= $\forall x.$ **on** (**birthday** x) (**phoned** x) **m**

Because *everyone* is embedded within the modified VP *phoned everyone*, it does not c-command the pronoun. This is problematic for LF-style theories on which binding depends on c-command at surface structure. Our theory not only generates these cases as desired, but also correctly predicts that reversing the binder and pronoun positions results in a crossover violation. For example, our theory correctly rules out **Mary phoned his$_i$ mother on everyone$_i$'s birthday*.

5. Explaining superiority

Just as for quantificational binding and crossover, the grammar needs no additional stipulations in order to account for superiority contrasts. The explanation boils down to the relative scope between a *wh*-trace and nearby *wh*-phrases: a fronted *wh*-phrase can only bind its *wh*-trace if the *wh*-trace is evaluated before any in-situ *wh*-word taking scope at the same clause.

Consider first a single-*wh* question such as

(41) Who$_i$ did Mary say $\_\__i$ bought something?

In order for the raised *wh*-phrase to bind its trace, its complement must provide access to the trace; in semantic terms, the complement must denote a function whose first argument corresponds to the trace. Thus *Mary say* __ *bought something* might denote $\lambda t.\, \mathbf{say}(\exists y.\, \mathbf{buy}\ y\ t)\ \mathbf{m}$, where λt abstracts over the argument position corresponding to the trace. In scope terms, the trace must receive widest scope over the gapped clause.

In particular, if there is also an in-situ *wh*-phrase, the trace must outscope the in-situ *wh*-phrase. Thus in the multiple-*wh*-phrase

(42) Who did Mary say __ bought what?

the constituent *Mary say* __ *bought what* must denote the function $\lambda t.\, \mathbf{what}(\lambda y.\, \mathbf{say}(\mathbf{buy} y t)\mathbf{m})$, in which λt outscopes λy.

Now we are ready to examine a superiority-observing multiple-*wh* question in more detail.

(36) Who did Mary give __ what?
((Q who) (D ((S (U did)) ((S (U (L mary))) ((S ((S (U give)) T)) what)))))
$= \mathtt{e}\ ?\ (\mathtt{e}\ ?\ \mathtt{t}) : \mathbf{who}(\lambda x.\, \mathbf{what}(\lambda y.\, ((\mathbf{give}\ x)\ y)\ \mathbf{m}))$

In the derivation (36), the trace scopes over the in-situ *wh*-phrase *what*. This scope relationship is evident from the fact that the subexpression *did Mary give* __ *what* has the category $\mathtt{e} \backslash\!\backslash (\mathtt{e}\ ?\ \mathtt{t})$, in which the outer connective $\backslash\!\backslash$ is contributed by the trace, and the inner connective ? is contributed by *what*. Accordingly, the denotation of that subexpression is $\lambda t.\, \mathbf{what}(\lambda y.\, \mathbf{give}\ t\ y\ \mathbf{m})$ rather than $\mathbf{what}(\lambda y.\, \lambda t.\, \mathbf{give}\ t\ y\ \mathbf{m})$. Since the trace scopes over the in-situ *wh*-phrase, the fronted *wh*-phrase is able to bind its trace, and the derivation goes through.

Now consider the superiority-violating counterpart of (36).

(43) *What did Mary give who __?

An in-situ *wh*-word (*who*) intervenes between the raised *wh*-word (*what*) and its trace. By the reasoning just given, the trace must take scope over *who*. But as explained in Section 3.3, the trace must take effect at a higher continuation level in order to take scope over *who* to its left. Thus the gapped clause must be derived using two continuation levels, and we must lower it twice before combining it with the raised *wh*-phrase *what*. The normal way to lower the continuation level is to apply Down. We can apply Down once here, resulting in

(44) did Mary give who __
((S (U D)) ((S (U (S (U did)))) ((S (U (S (U (L mary))))) ((S (U (S ((S (U give)) who)))) ((S (U U)) T)))))
$= (\mathtt{e} \backslash\!\backslash A) /\!/ ((\mathtt{e}\ ?\ \mathtt{t}) \backslash\!\backslash A) : \lambda\kappa.\, \lambda x.\, \kappa(\mathbf{who}(\lambda y.\, ((\mathbf{give}\ y)\ x)\ \mathbf{m}))$

The category and the semantics above both show the trace scoping over the in-situ *wh*-phrase, as desired. But this category still contains one extra layer of continuations, which prevents it from combining with the fronted *what*. If we could feed the identity continuation to this denotation—in other words, if we could apply Down to it once more—we would be able to derive the superiority-violating meaning $\lambda x.\, \mathbf{who}(\lambda y.\, ((\mathbf{give}\ y)\ x)\ \mathbf{m})$. But Down only matches a category of the form $A /\!\!/ (\mathtt{t} \setminus\!\!\setminus \mathtt{t})$. Since $(\mathtt{e}\ ?\ \mathtt{t}) \setminus\!\!\setminus A$ does not unify with $\mathtt{t} \setminus\!\!\setminus \mathtt{t}$, Down cannot apply, and the derivation fails.

The upshot is that a *wh*-trace must be evaluated before any in-situ *wh*-phrase for the same clause. Given left-to-right evaluation, this means the trace must precede any in-situ *wh*-phrases. In other words, given left-to-right evaluation, an intervening in-situ *wh*-word fatally disrupts the binding relation between a raised *wh*-word and its trace.

5.1. **An empirical refinement: D-linked *wh*-phrases.** Since Pesetsky's (1987) work, it is usually assumed that an in-situ *wh*-phrase no longer triggers superiority violations if is somehow 'linked to the discourse' in some imprecise sense.

(45) What did which students buy __?

The question (45) would ordinarily violate superiority, since *what* raises when the in-situ *wh*-phrase *which students* could have. But *wh*-phrases headed by *which* are assumed to be intrinsically linked to the discourse (D-linked) by virtue of some aspect of their meaning, which is supposedly why sentences like (45) are grammatical.

So if we want to derive (45), we must make the in-situ *wh*-phrase let the trace take wide scope. This can be done by making the *which*-phrase anticipate a subsequent trace.

(46) which N $\quad (\mathtt{e} \setminus\!\!\setminus (\mathtt{e}\ ?\ A)) /\!\!/ (\mathtt{e} \setminus\!\!\setminus (\mathtt{e} \setminus\!\!\setminus A)) \quad \lambda\kappa.\, \lambda t.\, \mathbf{which}(\mathbf{N})(\lambda x.\, (\kappa x)t)$

The semantics simply gives the trace (λt) wide scope by fiat.

We can now derive (45).

(47) What did which students buy _?
((Q what) (D ((S (U did)) ((S ((S (U L)) which-students)) ((S (U buy)) T)))))
$= \mathtt{e}\ ?\ (\mathtt{e}\ ?\ \mathtt{t}) : \mathbf{what}(\lambda y.\, \mathbf{which}(\mathbf{students})(\lambda x.\, \mathbf{buy}\ y\ x))$

Needless to say, there are additional complexities related to D-linking that we are not able to address here.

6. Wh-binding, crossover, and reconstruction: distinguishing evaluation order from linear order

Like quantifiers, *wh*-phrases can bind pronouns, and when they do, they show crossover effects—but with some twists that any adequate theory of *wh*-questions and binding must address. One twist, which we call 'pied binding', is that binders that are pied-piped in a raised *wh*-phrase interact with crossover. Another twist, standardly called 'reconstruction', is that pronouns in a raised *wh*-phrase also interact with crossover. We first discuss basic crossover facts in *wh*-questions, then address these twists in turn.

If evaluation order were always the same as linear order, (48) would pose a serious challenge to our approach.

(48) a. Who$_i$ __ saw his$_i$ mother?
b. *Who$_i$ did his$_i$ mother see __?

This pattern seems mysterious at first glance, since in both sentences the raised *wh*-phrase is to the left of the pronoun in question, c-commands it, and outscopes it, so both examples ought to be equally grammatical.

As a first approximation, Reinhart (1983; p. 114) and others have suggested that it is the *wh*-trace that binds the pronoun, and not the raised *wh*-word directly. In our system, this corresponds to applying Bind to the *wh*-trace:

(49) B T = $(\mathtt{e} \backslash\!\backslash A) /\!/ (\mathtt{e} \backslash\!\backslash (\mathtt{e} \triangleright A)) : \lambda\kappa.\, \lambda x.\, (\kappa x) x$

This is sufficient to derive the grammatical example in (48a):

(50) Who $\_\__i$ saw his$_i$ mother?
((Q who) (D ((S ((S (U L)) (B T))) ((S (U saw)) ((S ((S (U L)) his)) (U mother))))))
= $\mathtt{e} \,?\, \mathtt{t} : \mathbf{who}(\lambda x.\, \mathbf{saw}(\mathbf{mother}\ x)\ x)$

In this derivation, the fact that Bind applies directly to Trace ('B T') shows that it is the trace that is binding *his*.

The ungrammatical sentence in (48b) has no analogous derivation, or any derivation at all. The reason is that, in order for the trace (or any other antecedent) to bind the pronoun, it must be evaluated before the pronoun, which means in this case that it must precede the pronoun.

6.1. **Pied binding.** However, merely allowing the trace to bind is not sufficient to generate the full range of grammatical interpretations.

(51) a. Whose$_i$ friend$_j$'s next-door neighbor$_k$ did Mary think $\_\__k$ saw his$_{i/j/k}$ mother?
b. Whose$_i$ friend$_j$'s next-door neighbor$_k$ did Mary think his$_{*i/*j/*k}$ mother saw $\_\__k$?

In (51a), the trace must corefer with the entire raised *wh*-phrase. In other words, it must be a next-door neighbor who does the seeing, as our analysis guarantees. But as the subscripts indicate, the pronoun *his* can be bound by not just *whose friend's next-door neighbor* but also *whose friend* or just *whose*. If the pronoun could only be bound by the trace, then the pronoun would only be able to refer to the next-door neighbor. Apparently (contra Reinhart's suggestion), this pronoun can be bound directly by a *wh*-phrase. Nevertheless, the trace still plays a critical role, as shown in (51b): if the trace fails to precede the pronoun, none of these binding possibilities are possible.

We call the pattern in (51) 'pied binding'. Remarkably, our system already predicts this intricate pattern of facts, by allowing a *wh*-phrase to bind the pronoun directly. For instance, to derive the sentence in (50), besides letting the trace bind the pronoun, we can also let *who* bind the pronoun:

(52) Who$_i$ $\_\_$ saw his$_i$ mother?
((Q (B who)) (D ((S ((S (U L)) T)) ((S (U saw)) ((S ((S (U L)) his)) (U mother))))))
= $\mathtt{e} \,?\, \mathtt{t} : \mathbf{who}(\lambda x.\, \mathbf{saw}(\mathbf{mother}\ x)\ x)$

In contrast with the derivation in (50), here Bind applies directly to *who*, not to Trace.

The pied-binding possibilities in (51a) arise from the possibilities of applying the Bind operator to either *whose*, *whose friend*, or *whose friend's next-door neighbor*. We illustrate this interaction of binding with pied-piping with the following simpler example.

(53) Whose$_i$ friend $\_\_$ saw his$_i$ mother?
((Q ((S ((S (U L)) (B whose))) (U friend))) (D ((S ((S (U L)) T)) ((S (U saw)) ((S ((S (U L)) his)) (U mother))))))
= $\mathtt{e} \,?\, \mathtt{t} : \mathbf{who}(\lambda x.\, \mathbf{saw}(\mathbf{mother}\ x)\ (\mathbf{friend}\ x))$

As desired, the trace refers to the friend, but the pronoun is bound by *whose*. (Note that Bind applies here to *whose*.)

Crucially, even considering the possibility that *wh*-phrases can bind pronouns directly, the ungrammatical interpretations in (48b) and (51b) still have no derivation. The reason is that (B who) in (48b) (likewise (B whose) in (51b)) has category $(\mathtt{e} \,?\, A)/\!\!/(\mathtt{e}\backslash\!\!\backslash(\mathtt{e} \triangleright A))$. In this category, the $\mathtt{e}\backslash\!\!\backslash$ for the trace is outside the $\mathtt{e} \triangleright$ for the pronoun, so the trace must outscope the pronoun. Thus, just as in the explanations above for crossover and superiority, the trace must precede the pronoun. In other words, the ability of a *wh*-phrase to bind its trace is disrupted by an intervening pronoun that tries to be bound from within that *wh*-phrase, just as it is disrupted by an intervening in-situ *wh*-word.

A final point on *wh*-binding: as explained above in Section 4, a quantifier embedded in a verb phrase can bind into an adjunct, and, as noted by Lasnik and Stowell (1991), so can a *wh*-phrase whose trace is embedded in a verb phrase:

(54) Which actress$_i$ did you [admit you had an affair with $\_\__i$] [after she$_i$ died]?

As the brackets above show, the adjunct containing the bound pronoun can modify the entire complex verb phrase in which the *wh*-trace is embedded. After all, presumably it is the admission and not the affair that takes place after the death (thanks to Daniel Büring for this example). We derive a simpler example that makes the same point:

(55) Who$_i$ did Mary call $\_\__i$ on his$_i$ birthday?
((Q (B who)) (D ((S (U did)) ((S (U (L mary))) ((S ((S (U L)) ((S (U call)) T))) ((S (U on)) ((S ((S (U L)) his)) (U birthday))))))))
$= \mathtt{e} \,?\, \mathtt{t} : \mathbf{who}(\lambda x.\, \mathbf{on}(\mathbf{birthday}\ x)(\mathbf{call}\ x)\ \mathbf{m})$

In the same way that a quantifier embedded in a verb phrase can take scope over and bind a pronoun in a following adjunct, the trace can take scope over a pronoun to its right, allowing that pronoun to be bound by the *wh*-phrase. Once again, examples like these are problematic for theories that attempt to derive binding from surface c-command relations, but fall out without stipulation here.

6.2. **Reconstruction.** 'Reconstruction' means that some phenomenon (in the present context, binding) appears to behave as if some fronted phrase were in the position of its trace. Reconstruction can let a pronoun linearly precede a quantifier that binds it.

(56) Whose criticism of his$_i$ mother did everyone$_i$ resent _?
((Q ((S ((S (U S)) ((S (U U)) ((S (U L)) whose)))) ((S (U (S (U criticism-of)))) (U ((S ((S (U L)) his)) (U mother)))))) (D ((S (U did)) ((S (U D)) ((S (U (S ((S (U L)) (B everyone))))) ((S (U (S (U resent)))) T))))))
$= \lambda x.\, \forall y.\, (\mathbf{resent}((\mathbf{criticism\text{-}of}(\mathbf{mother}\ y))\ x))\ y$

The derivation above shows that our system already generates this example of cataphora. The key is to instantiate the Trace type-shifter to raise not an individual but a lifted individual. Among the instantiations of the Trace type-shifter are the following:

(57) a. Trace : $B/\!\!/B$
b. Typical trace : $(\mathtt{e}\backslash\!\!\backslash\mathtt{t})/\!\!/(\mathtt{e}\backslash\!\!\backslash\mathtt{t})$
c. Reconstruction trace : $(\underline{((\mathtt{e} \triangleright \mathtt{t})/\!\!/(\mathtt{e}\backslash\!\!\backslash\mathtt{t}))}\backslash\!\!\backslash\mathtt{t})/\!\!/(\underline{((\mathtt{e} \triangleright \mathtt{t})/\!\!/(\mathtt{e}\backslash\!\!\backslash\mathtt{t}))}\backslash\!\!\backslash\mathtt{t})$

The original, fully general Trace rule, repeated in (57a), is quite simple. Every example in the preceding sections can be derived using the typical instantiation (57b), in which the gap functions locally as category $\mathtt{e}$. The instantiation (57c), required for reconstruction examples like (56), looks complex, but it is just like (57b), except that $\mathtt{e}$ is replaced by $(\mathtt{e} \triangleright \mathtt{t})/\!\!/(\mathtt{e}\backslash\!\!\backslash\mathtt{t})$ (underlined above). But $(\mathtt{e} \triangleright \mathtt{t})/\!\!/(\mathtt{e}\backslash\!\!\backslash\mathtt{t})$ is the category of a pronoun, so the

reconstruction trace in effect hypothesizes a pronoun in the gap position. This hypothetical pronoun, like a real pronoun, can seek out a binder.

Thus the reconstruction trace functions locally as a pronoun, takes scope over a clause, and returns as result an expression of category $((\mathtt{e} \triangleright \mathtt{t}) /\!\!/ (\mathtt{e} \backslash\!\!\backslash \mathtt{t})) \backslash\!\!\backslash \mathtt{t}$. This is the category of *did everyone resent* __, a continuation that maps pronoun meanings into clause meanings. The raised *wh*-phrase *whose criticism of his mother* can then turn this continuation into a question.

In processing terms, this analysis models a processor that postpones evaluating the raised *wh*-phrase, including the pied-piped pronoun, until the trace is reached. This postponement is built into the Question type-shifter, which is what combines the meaning of the fronted *wh*-phrase with the meaning of a gapped clause. If the meaning of the fronted *wh*-phrase is more complex (such as containing a bindable pronoun), then the category of the trace must also be more complex. Regardless, the net result is that the entire fronted *wh*-phrase is evaluated when the trace is.

Sternefeld (1997) also handles reconstruction using complex traces. His complex traces are called 'pseudo-variables', and have the form $\lambda g.\, g(i)$. Two ways in which Sternefeld's approach differs significantly from ours are that he makes heavy use of assignment functions, and that he adds special semantic machinery in order to interpret pseudo-variables differently from normal variables. Our system handles reconstruction with no special stipulation: the trace category that allows reconstruction is just one of many possible instantiations of the general Trace rule posited to handle basic *wh*-question formation.

Jäger's account of binding (2001) requires that a binder linearly precede any pronoun that it binds. As a result, all instances of right-to-left binding, including reconstruction, are a challenge for his approach. Jäger handles reconstruction examples involving *which*, such as *which of his$_i$ friends did each man$_i$ call*, by providing a second lexical entry for *which* that explicitly anticipates a pronoun in the fronted *wh*-phrase and transmits binding to that pronoun from the gap. It is not clear how to extend his solution to examples like (56) above, since *whose* arguably does not take the following nominal as an argument. Furthermore, Jäger needs an infinite number of lexical entries for *which*, each anticipating a different number of pronouns that need to be bound. In our system, reconstruction binding falls out from the general treatment of *wh*-binding without additional stipulation, and in particular without requiring lexical polysemy. Though we agree with Jäger that an adequate account of weak crossover must recognize some kind of left-to-right order, the fact that our analysis automatically provides right-to-left binding in at least some reconstruction situations supports our claim that it is not linear order that matters, but evaluation order.

The general prediction of our analysis is that whenever there is independent reason to expect that evaluation can be delayed (as when establishing the semantic connection between a fronted *wh*-phrase and its trace position), we can expect that right-to-left quantificational binding will be possible. In other words, this kind of reconstruction is a case of delayed evaluation.

7. Reversing evaluation order: explaining the weakness of weak crossover

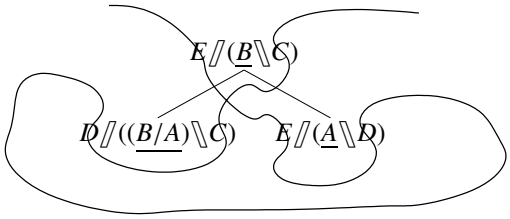
We have claimed that our Scope rule is biased for left-to-right evaluation, but it may not be obvious how the bias is built into the rule. Perhaps the best way to understand the role of the Scope rule in determining order of evaluation is to consider the following alternative version of the rule, which differs only in that it evaluates expressions from right to left instead of from left to right. We call the alternative operator Z (the mirror image of an 'S', with apologies to Pauline Jacobson).

(58) Left-to-right S: $((E/\!\!/(B\backslash\!\!\backslash C))\,/\,(D/\!\!/(A\backslash\!\!\backslash C)))\,/\,(E/\!\!/((B/A)\backslash\!\!\backslash D))$
Semantics: $\lambda L.\,\lambda R.\,\lambda\kappa.\,L(\lambda l.\,R(\lambda r.\,\kappa(lr)))$

(59) Right-to-left Z: $((E/\!\!/(B\backslash\!\!\backslash C))\,/\,(E/\!\!/(A\backslash\!\!\backslash D)))\,/\,(D/\!\!/((B/A)\backslash\!\!\backslash C))$
Semantics: $\lambda L.\,\lambda R.\,\lambda\kappa.\,R(\lambda r.\,L(\lambda l.\,\kappa(lr)))$

One place to see the change in evaluation direction is in the semantics: in the left-to-right version, the first, leftmost argument L has in view all of the information contained in R. In the right-to-left version, R has L in view, and the scope relations are reversed.[13]

As with S, it is helpful to show Z in terms of the configurations of categories it licenses.



The underlined parts show that the directionality of the main semantic combination remains unchanged: B/A followed by A still produces B. However, the directionality of side effects is flipped: it is now the rightmost daughter that determines E, the result type of the mother category.[14]

If we replace S with Z in our grammar, the order of evaluation switches from left-to-right to right-to-left. Consequently, we should expect a reversal in the behavior of phenomena that are sensitive to order of evaluation. And this is in fact exactly what happens. We have discussed three order-sensitive phenomena—quantifier scope, crossover, and superiority—so let us consider each of these in turn.

First, as we saw in (17) (repeated here), S is biased in favor of linear scope in the following sense: on the simplest derivation, quantificational expressions on the left outscope those to the right.

(17) Someone saw everyone.
(D ((S ((S (U L)) someone)) ((S (U saw)) everyone))) = $\exists y.\,\forall x.\,(\mathbf{saw}\ x)\ y$

(60) Someone saw everyone.
(D ((Z ((Z (U L)) someone)) ((Z (U saw)) everyone))) = $\forall x.\,\exists y.\,(\mathbf{saw}\ x)\ y$

In the first derivation (17), S is used, and the default scope is left-to-right. The second derivation (60) is identical, except that Z is used instead of S, with the result that scope relations are reversed. Thus Z induces a right-to-left bias in quantificational scoping.

Second, if we replace S with Z in our grammar, then the crossover-violation example (37b) becomes easy to derive.

(61) *His$_i$ mother saw everyone$_i$.
(D ((Z ((Z (U L)) ((Z ((Z (U L)) his)) (U mother)))) ((Z (U saw)) (B everyone))))
= $\forall x.\,(\mathbf{saw}\ x)\ (\mathbf{mother}\ x)$

[13] In each case, the semantic values are completely determined by the categories of the type-shifting rules via the Curry-Howard isomorphism. See Barker (2004) for a derivation of the semantic value for the S rule.

[14] The fact that the curve crosses itself in this diagram is coincidental. If we had chosen the reverse slash convention for $/\!\!/$ and $\backslash\!\!\backslash$, then the curve would be uncrossed for Z and crossed for S instead. However, given any slash convention, the curve will be crossed for either S or Z. Since we believe that S is the default, we prefer the slash convention that leads to a crossed curve for the Z rule.

As explained in Section 4, a quantifier must be evaluated before any pronoun it binds. In (61), since the order of evaluation is reversed using Z, the quantifier does get evaluated first, and the ungrammatical crossover violation is incorrectly generated. At the same time, the grammatical sentence (37a) is no longer generated, for reasons exactly analogous to the explanation in Section 4.

Finally, we can convince ourselves that evaluation order plays a crucial part in the explanation for superiority. With Z in play, superiority predictions are reversed: the normally grammatical *Who did Tom say bought what?* fails to be generated, and superiority violations can be derived as follows.

(62) *What did who buy __?
((Q what) (D ((Z (U did)) ((Z ((Z (U L)) who)) ((Z (U buy)) T)))))
= e ? (e ? t) : **what**(λy. **who**(λx. (**buy** y) x))

As explained in Section 5, a *wh*-trace must be evaluated before any in-situ *wh*-phrase for the same clause. For left-to-right evaluation (using S), then, the *wh*-trace must linearly precede the in-situ *wh*-phrase, but for right-to-left evaluation (using Z), the *wh*-trace must linearly follow the in-situ *wh*-phrase, since expressions on the right get evaluated first.

We can now make precise our proposal that English evaluates expressions from left to right: we stipulate that the grammar contains S but not Z.[15] Such a grammar embodies a bias for linear scope, yet still allows inverse scope; allows quantificational binding, but not crossover; and generates multiple-*wh* questions, but not superiority violations.

Unlike some types of ungrammaticality, crossover violations and superiority violations remain interpretable with enough mental effort, and violations are often judged as not so bad, especially in comparison with, e.g., strong crossover. Our explanation suggests that speakers can resort to a non-default processing strategy (temporarily using Z instead of S) given sufficient motivation.

To recap, our predictions of both crossover and superiority depend on the order of evaluation, in that reversing the order of evaluation reverses the pattern of predicted grammaticality. In this sense, left-to-right evaluation provides a unified account of scope, crossover and superiority.

7.1. **Independent motivation for our characterization of evaluation order: S versus Z in programming languages.** When we say that S and Z differ in evaluation order, we are using the notion of evaluation order that is standard in programming language research (Meyer and Wand 1985, p. 223; Danvy and Filinski 1989, p. 15; Papaspyrou 1998). To illustrate this connection explicitly, let us analyze the evaluation order of a simple programming language using continuations.

Consider the following language of arithmetic expressions: an expression is either a number or two expressions conjoined with a plus sign +. Each expression can be *evaluated* to give a numeric result.

(63)

	Expression	*Result*
a.	2	2
b.	2 + 3	5

[15] Jacobson (1999) speculatively proposes an account of crossover with exactly the same form, i.e., accepting one type-shifter while rejecting another. Extending Jacobson's approach to a grammar with broader coverage requires considerable complication (Barker to appear); furthermore, there is no obvious processing motivation for Jacobson's choice of operator. Nevertheless, our proposal is very much in the same spirit as hers, and we acknowledge the inspiration provided by her work.

Suppose that our computer is connected to a printer. To use the printer, we add a new feature `print` to our programming language. If e is an expression, then `print(`e`)` is also an expression. The expression `print(`e`)` evaluates just as e does, but in addition sends the result to the printer as a side effect.

(64)

	Expression	*Result*	*Printer*
a.	`2 + 3`	5	(no output)
b.	`print(2 + 3)`	5	5
c.	`2 + print(3)`	5	3

We have yet to specify what happens if a single expression invokes `print` more than once. For instance, suppose we evaluate `print(2) + print(3)`. The numeric result produced should clearly be 5, but it is unclear whether the printer should print 2 followed by 3 or vice versa. Intuitively, if the subexpression `print(2)` is evaluated first, then 2 will be printed first.

The behavior of this programming language can be modeled using continuations. Let `n` be the type of numbers, and `p` be the type of printer outputs. (Perhaps each `p`-value is a sequence of `n`-values.) On the continuations analysis, every expression denotes something of the type $\langle\langle n, p\rangle, p\rangle$. The semantic rules are as follows.

(65)

a. $[\![\texttt{2}]\!] = \lambda c.\, c(2)$

b. $[\![\texttt{3}]\!] = \lambda c.\, c(3)$

c. i. $[\![e_1 \texttt{ + } e_2]\!] = \lambda c.\, [\![e_1]\!](\lambda n_1.\, [\![e_2]\!](\lambda n_2.\, c(n_1 + n_2)))$ left to right

ii. $[\![e_1 \texttt{ + } e_2]\!] = \lambda c.\, [\![e_2]\!](\lambda n_2.\, [\![e_1]\!](\lambda n_1.\, c(n_1 + n_2)))$ right to left

d. $[\![\texttt{print(}e\texttt{)}]\!] = \lambda c.\, [\![e]\!](\lambda n.\, \mathbf{cons}(n, c(n)))$

Under (65c) above are two alternative semantic rules for expressions built with the plus sign `+`. The first alternative (65ci) implements left-to-right evaluation and is analogous to S; the second (65cii) implements right-to-left evaluation and is analogous to Z. Also, the rule (65d) uses an auxiliary two-place function **cons**, defined as follows: if n is a number and p is a piece of printer output, then $\mathbf{cons}(p, n)$ is another piece of printer output, the result of prepending n to p.

Given any expression e, we can use the semantic rules in (65) to compute the printer output generated by e: it is $[\![e]\!](\lambda n.\, \mathbf{nil})$, where $\lambda n.\, \mathbf{nil}$ is the constant function returning **nil**, the empty output. For example, if we pick the left-to-right rule (65ci) and disregard the right-to-left rule (65cii), then the denotation of `print(2) + print(3)` is

$$
\begin{aligned}
&[\![\texttt{print(2) + print(3)}]\!] \\
&\quad = \lambda c.\, [\![\texttt{print(2)}]\!](\lambda n_1.\, [\![\texttt{print(3)}]\!](\lambda n_2.\, c(n_1 + n_2))) \\
&\quad = \lambda c.\, (\lambda c.\, [\![\texttt{2}]\!](\lambda n.\, \mathbf{cons}(n, c(n))))\\
&\qquad\qquad (\lambda n_1.\, (\lambda c.\, [\![\texttt{3}]\!](\lambda n.\, \mathbf{cons}(n, c(n))))(\lambda n_2.\, c(n_1 + n_2))) \\
&\quad = \lambda c.\, (\lambda c.\, (\lambda c.\, c(2))(\lambda n.\, \mathbf{cons}(n, c(n))))\\
&\qquad\qquad (\lambda n_1.\, (\lambda c.\, (\lambda c.\, c(3))(\lambda n.\, \mathbf{cons}(n, c(n))))(\lambda n_2.\, c(n_1 + n_2))) \\
&\quad = \lambda c.\, (\lambda c.\, \mathbf{cons}(2, c(2)))(\lambda n_1.\, (\lambda c.\, \mathbf{cons}(3, c(3)))(\lambda n_2.\, c(n_1 + n_2))) \\
&\quad = \lambda c.\, \mathbf{cons}(2, \mathbf{cons}(3, c(5))),
\end{aligned}
$$

so the printer output generated is

$$(\lambda c.\, \mathbf{cons}(2, \mathbf{cons}(3, c(5))))(\lambda n.\, \mathbf{nil}) = \mathbf{cons}(2, \mathbf{cons}(3, \mathbf{nil})),$$

in which 2 precedes 3. If we pick the right-to-left rule (65cii) rather than the left-to-right rule (65ci), then 3 would precede 2 in the printer output.

8. Conclusions

We have presented a concrete analysis of quantification, binding, and *wh*-question formation, in which constraints against crossover and superiority follow from the assumption that expressions are evaluated from left to right. Furthermore, when the type-shifting rule S is replaced with Z, the pattern of configurations predicted to violate crossover and superiority is reversed. Our analysis thus explains the weakness of weak crossover and of superiority—that is, the ability of language users to process and interpret violations—as the ability of language users to temporarily resort to a non-default processing strategy (right-to-left evaluation instead of left-to-right).

There can be little doubt that language has a bias for evaluating expressions in the order in which they are heard, for some sense of 'evaluate'. Nevertheless, there are many different ways in which this general principle could be built into a grammar. We have provided one specific hypothesis giving a precise picture of what it means for a language to evaluate expressions from left to right. Furthermore, our notion of order of evaluation is the same notion proposed independently for characterizing the evaluation disciplines of computer programming languages. Remarkably, the single stipulation that processing proceeds from left to right explains crossover and superiority, once the expressions involved are provided with suitable denotations. In addition, the same analysis makes correct predictions on intricate examples involving binding out of, as well as into, raised *wh*-phrases.

In computer science, continuations are the tool of choice for analyzing execution disciplines (order of evaluation, call-by-value versus call-by-name, etc.) and side effects (printing, input, errors, etc.). In natural language, we claim that two fairly mysterious phenomena, crossover and superiority, arise from a bias for left-to-right evaluation. Whether or not our specific version of the left-to-right processing hypothesis is correct, making the hypothesis precise and exploring its predictions requires the use of continuations as an analytic tool. Thus continuations are valuable in the search for new and deeper insights into the structure of language.

References

Barker, Chris. 2002. Continuations and the nature of quantification. *Natural Language Semantics* 10(3):211–242.

———. 2004. Continuations in natural language. In *CW'04: Proceedings of the 4th ACM SIGPLAN continuations workshop*, ed. Hayo Thielecke, 1–11. Tech. Rep. CSR-04-1, School of Computer Science, University of Birmingham.

———. To appear. Remark on Jacobson 1999: Crossover as a local constraint. *Linguistics and Philosophy*.

Barker, Chris, and Chung-chieh Shan. To appear. Types as graphs: Continuations in type logical grammar. *Journal of Logic, Language and Information*.

Bresnan, Joan. 1994. Linear order vs. syntactic rank: Evidence from weak crossover. In *CLS 30-1: The main session*. Chicago Linguistic Society.

———. 1998. Morphology competes with syntax: Explaining typological variation in weak crossover effects. In *Is the best good enough? Optimality and competition in syntax*, ed. Pilar Barbosa, Danny Fox, Paul Hagstrom, Martha McGinnis, and David Pesetsky, 59–92. Cambridge: MIT Press.

Büring, Daniel. 2001. A situation semantics for binding out of DP. In *SALT XI: Semantics and linguistic theory*, ed. Rachel Hastings, Brendan Jackson, and Zsofia Zvolensky, 56–75. Ithaca: Cornell University Press.

———. 2004. Crossover situations. *Natural Language Semantics* 12(1):23–62.

Chierchia, Gennaro. 1991. Functional WH and weak crossover. In *Proceedings of the 10th West Coast Conference on Formal Linguistics*, ed. Dawn Bates, 75–90. Stanford, CA: Center for the Study of Language and Information.

———. 1993. Questions with quantifiers. *Natural Language Semantics* 1(2):181–234.

———. 1995. *The dynamics of meaning: Anaphora, presupposition, and the theory of grammar*. Chicago: University of Chicago Press.

Chomsky, Noam. 1973. Conditions on transformations. In *A festschrift for Morris Halle*, ed. Stephen Anderson and Paul Kiparsky, 232–286. New York: Holt, Rinehart and Winstron.

Danvy, Olivier, and Andrzej Filinski. 1989. A functional abstraction of typed contexts. Tech. Rep. 89/12, DIKU, University of Copenhagen, Denmark. `http://www.daimi.au.dk/~danvy/Papers/fatc.ps.gz`.

———. 1990. Abstracting control. In *Proceedings of the 1990 ACM conference on Lisp and functional programming*, 151–160. New York: ACM Press.

———. 1992. Representing control: A study of the CPS transformation. *Mathematical Structures in Computer Science* 2(4):361–391.

Dayal, Veneeta. 1996. *Locality in WH quantification: Questions and relative clauses in hindi*. Dordrecht: Kluwer.

———. 2003. Multiple WH questions. In *The Blackwell companion to syntax*, ed. Martin Everaert and Henk C. van Riemsdijk. Oxford: Blackwell.

Dowty, David. 1991. 'Variable-free' syntax, variable-binding syntax, the natural deduction lambek calculus, and the crossover constraint. In *Proceedings of the 11th West Coast Conference on Formal Linguistics*, ed. Jonathan Mead. Stanford, CA: Center for the Study of Language and Information.

Felleisen, Matthias. 1987. The calculi of λ_v-CS conversion: A syntactic theory of control and state in imperative higher-order programming languages. Ph.D. thesis, Computer Science Department, Indiana University. Also as Tech. Rep. 226.

———. 1988. The theory and practice of first-class prompts. In *POPL '88: Conference record of the annual ACM symposium on principles of programming languages*, 180–190. New York: ACM Press.

Griffin, Timothy G. 1990. A formulae-as-types notion of control. In *POPL '90: Conference record of the annual ACM symposium on principles of programming languages*, 47–58. New York: ACM Press.

de Groote, Philippe. 2001. Type raising, continuations, and classical logic. In *Proceedings of the 13th Amsterdam Colloquium*, ed. Robert van Rooy and Martin Stokhof, 97–101. Institute for Logic, Language and Computation, Universiteit van Amsterdam.

Heim, Irene, and Angelika Kratzer. 1998. *Semantics in generative grammar*. Oxford: Blackwell.

Hendriks, Herman. 1993. Studied flexibility: Categories and types in syntax and semantics. Ph.D. thesis, Institute for Logic, Language and Computation, Universiteit van Amsterdam.

Higginbotham, James. 1980. Pronouns and bound variables. *Linguistic Inquiry* 11:679–708.

Hindley, J. Roger, and Jonathan P. Seldin. 1986. *Introduction to combinators and λ-calculus*. Cambridge: Cambridge University Press.

Hornstein, Norbert. 1995. *Logical form: From GB to Minimalism*. Oxford: Blackwell.

Jacobson, Pauline. 1999. Towards a variable-free semantics. *Linguistics and Philosophy* 22(2):117–184.

Jäger, Gerhard. 2001. Anaphora and type logical grammar. Habilitationsschrift, Humboldt University Berlin. UIL-OTS Working Papers 01004-CL/TL, Utrecht Institute of Linguistics (OTS), Utrecht University.

Kuno, Susumu, and Jane J. Robinson. 1972. Multiple wh questions. *Linguistic Inquiry* 3: 463–487.

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

May, Robert. 1985. *Logical form: Its structure and derivation*. Cambridge: MIT Press.

Meyer, Albert R., and Mitchell Wand. 1985. Continuation semantics in typed lambda-calculi (summary). In *Logics of programs*, ed. Rohit Parikh, 219–224. Lecture Notes in Computer Science 193, Berlin: Springer-Verlag.

Moortgat, Michael. 1997. Categorial type logics. In *Handbook of logic and language*, ed. Johan F. A. K. van Benthem and Alice G. B. ter Meulen, chap. 2. Amsterdam: Elsevier Science.

O'Neil, John. 1993. A unified analysis of superiority, crossover, and scope. In *Harvard working papers in linguistics*, ed. Höskuldur Thráinsson, Samuel D. Epstein, and Susumu Kuno, vol. 3, 128–136. Cambridge: Harvard University.

Papaspyrou, Nikolaos S. 1998. Denotational semantics of evaluation order in expressions with side effects. In *Recent advances in information science and technology: 2nd part of the proceedings of the 2nd IMACS international conference on circuits, systems and computers*, ed. Nikos E. Mastorakis, 87–94. Singapore: World Scientific.

Parigot, Michel. 1992. $\lambda\mu$-calculus: An algorithmic interpretation of classical natural deduction. In *Proceedings of LPAR '92: International conference on logic programming and automated reasoning*, ed. Andrei Voronkov, 190–201. Lecture Notes in Artificial Intelligence 624, Berlin: Springer-Verlag.

———. 1993. Classical proofs as programs. In *Proceedings of KGC'93: Computational logic and proof theory, 3rd Kurt Gödel colloquium*, ed. Georg Gottlob, Alexander Leitsch, and Daniele Mundici, 263–276. Lecture Notes in Computer Science 713, Berlin: Springer-Verlag.

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 Groenendijk, Dick de Jongh, and Martin Stokhof, 115–143. Groningen-Amsterdam Studies in Semantics 8, Dordrecht: Foris.

Partee, Barbara H., and Mats Rooth. 1983. Generalized conjunction and type ambiguity. In *Meaning, use and interpretation of language*, ed. Rainer Bäuerle, Christoph Schwarze, and Arnim von Stechow, 361–383. Berlin: de Gruyter.

Pesetsky, David. 1987. Wh-in-situ: Movement and unselective binding. In *The representation of (in)definiteness*, ed. Eric J. Reuland and Alice G. B. ter Meulen. Cambridge: MIT Press.

Phillips, Colin. 2003. Linear order and constituency. *Linguistic Inquiry* 34(1):37–90.

Plotkin, Gordon D. 1975. Call-by-name, call-by-value and the λ-calculus. *Theoretical Computer Science* 1(2):125–159.

Postal, Paul Martin. 1971. *Cross-over phenomena*. New York: Holt, Rinehart and Winston.

———. 1993. Remarks on weak crossover effects. *Linguistic Inquiry* 24:539–556.

Potts, Christopher. 2001. (Only) some crossover effects repaired. *Snippets* 3:13–14.

Reinhart, Tanya. 1983. *Anaphora and semantic interpretation*. London: Croom Helm.

Ruys, E. G. 2000. Weak crossover as a scope phenomenon. *Linguistic Inquiry* 31(3): 513–539.

Shan, Chung-chieh. 2002. A continuation semantics of interrogatives that accounts for Baker's ambiguity. In *SALT XII: Semantics and linguistic theory*, ed. Brendan Jackson, 246–265. Ithaca: Cornell University Press.

Sitaram, Dorai, and Matthias Felleisen. 1990. Control delimiters and their hierarchies. *Lisp and Symbolic Computation* 3(1):67–99.

Steedman, Mark. 2000. *The syntactic process*. Cambridge: MIT Press.

Sternefeld, Wolfgang. 1997. The semantics of reconstruction and connectivity. Arbeitspapiere des SFB 340 97, Universität Stuttgart and Tübingen.

Tanenhaus, Michael K., S. M. Garnsey, and J. Boland. 1990. Combinatory lexical information and language comprehension. In *Cognitive models of speech processing: Psycholinguistic and computational perspectives*, ed. G. T. M. Altmann, 383–408. Cambridge: MIT Press.

Wadler, Philip L. 1992. Comprehending monads. *Mathematical Structures in Computer Science* 2(4):461–493.

Division of Engineering and Applied Sciences, Harvard University
33 Oxford Street, Cambridge, MA 02138, USA
E-mail address: ccshan@post.harvard.edu
URL: http://www.digitas.harvard.edu/~ken/

0108 Department of Linguistics, University of California, San Diego
9500 Gilman Drive, La Jolla, CA 92093, USA
E-mail address: barker@ucsd.edu
URL: http://ling.ucsd.edu/~barker/