Semantics & Pragmatics Volume 13, Article 20: 1–40, 2020
https://doi.org/10.3765/sp.13.20

The logic of Quantifier Raising*

Chris Barker
New York University

Submitted 2019-06-04 / First decision 2020-06-09 / Revision received 2020-08-13 / Accepted 2020-11-23 / Published 2020-12-30 / Final typesetting 2022-09-28

Abstract Displaced scope is a hallmark of natural language, and Quantifier Raising (QR) has long been the standard tool for analyzing scope. Yet despite the foundational importance of QR to theoretical linguistics, as far as I know, there has never been a study of its formal properties. For instance, consider the decidability problem: given an initial syntactic structure, is there an algorithm that will determine whether a semantically coherent QR derivation exists? If at least one such derivation exists, is the number of semantically different analyses always finite? How do we know when we have found them all? Do the answers to these questions depend on imposing scope islands or other constraints on QR, such as forbidding vacuous movement, re-raising, remnant raising, raising of names, repeated type lifting, and so on? I settle these issues by defining QRT (Quantifier Raising with Types), a substructural logic that is a faithful model of QR in the following respect: every semantically coherent QR derivation corresponds to a semantically equivalent proof in QRT, and vice-versa. Since QRT is decidable and has the finite readings property, it follows that a broad class of theories that rely on QR also have these properties, without needing to place any formal constraints on QR. I go on to study the special relationship between type lifting and QR, drawing an analogy with eta reduction in the lambda calculus. Allowing unrestricted type lifting does not compromise decidability. In addition, it turns out that QR with type lifting validates the core type shifting principles of Flexible Montague Grammar, a paradigm example of an in-situ type-shifting approach to scope taking. This suggests that QR is compatible with a local,

* Thanks to Simon Charlow, Sandra Chung, Josh Dever, Berit Gehrke, Daniel Lassiter, Richard Moot, Geoffrey K. Pullum, Greg Restall, my anonymous referees, and audiences at LENLS 11, ESSLLI 2015, Stanford, and the New York Philosophy of Language Workshop. At LENLS 11, Berit Gehrke pressed me on the relation between Quantifier Raising and a particular logic related to QRT. I gave her a list of reasons why I thought they were deeply different. "So…" said Berit when I had finished, "it's Quantifier Raising!" This paper explores the sense in which she was right.

directly compositional view of scope taking. These results put Quantifier Raising on a reassuringly firm formal footing.

Keywords: quantifier raising, decidability, direct compositionality, scope, substructural logic, type shifting

1 Introduction

Scope-taking is one of the most dramatic, distinctive, and ubiquitous phenomena in natural language, and Quantifier Raising has long been the standard technique for investigating scope. Yet despite its importance to linguistic theory, as far as I know, there has never been a study of the formal properties of Quantifier Raising.

Here are some of the questions that this paper will address (and answer). Given a syntactic structure before Quantifier Raising, is there a procedure that is guaranteed to terminate and that will decide whether there is a series of QR operations that will result in a semantically coherent analysis? (Yes.) If there is at least one such analysis, is the number of semantically distinct analyses finite? (Yes.) Given that type lifting can turn an individual-denoting expression into a generalized quantifier, it creates additional opportunities for QR — does allowing lifting change the answer to either of the first two questions? (No.) Do we need to ban vacuous QR, QR of names, cyclical QR, or place limits on lifting in order to guarantee decidability? (No.) Do scope islands play a crucial role in guaranteeing decidability? (No.) Is it necessary (for reasons of decidability) to place any restrictions on what can undergo QR, or on what landing site QR chooses for adjunction? (No.)

In other words, the answers are as favorable as they could be. There may be good empirical reasons for constraining QR in various ways (imposing scope islands comes immediately to mind), but such constraints are not required in order to keep the search space for QR analyses bounded.

So what's at stake? Why should we care whether QR is decidable?

There is a theoretical answer and a practical answer. Theoretically, if QR were not decidable, then any realistic grammar that included QR would be committed to the existence of unanalyzable sentences. For such a sentence, it would be impossible to say whether it had any coherent semantic interpretation: no matter how many derivations you had already tried, the mere fact that you hadn't found one yet that works wouldn't guarantee that there isn't

one. Despite the fact that any given sentence either has a coherent QR analysis or it doesn't, if QR were not decidable, there would be specific sentences for which it would be impossible to know what predictions the theory made.

On a practical level, the finite readings property is highly desirable. If QR did not have the finite readings property, you might never be sure whether you had found all of the interpretations of a sentence, since there would be no way to know when to stop looking for the next interpretation. The results below will place an easily computable bound on the maximum derivational complexity required to account for all distinct interpretations for any given set of lexical items. Once your search reaches this bound, you can be sure you've found all of the possible analyses.

As any experienced semanticist knows, decidability worries don't arise in simple situations involving garden-variety generalized quantifiers scoping over clauses. However, there are realistic examples where finding any analysis, let alone all possible analyses, is not so obvious (see Section 3 for a concrete example). Not only do the results given below settle the theoretical issues, they provide a practical algorithm for computing all semantically distinct interpretations for an arbitrary example.

I go on to show that these results hold even if we add unrestricted type lifting, a generalized version of Partee's (1987) LIFT type shifter. LIFT is special among type shifters. For instance, there is a deep and intriguing analogy between LIFT and eta reduction in the lambda calculus. It turns out that QR with type lifting validates the core type shifting principles of Hendriks's (1993) Flexible Montague Grammar, which is a paradigm example of a non-movement, in-situ, directly compositional approach to scope taking. This suggests that QR is compatible with a local, directly compositional view of scope-taking.

My strategy will be to consider a substructural logic, QRT (Quantifier Raising with Types). QRT is faithful model of QR in the sense that every semantically coherent QR derivation has a semantically equivalent proof in QRT, and vice versa. Since QRT is decidable and has the finite readings property, it follows that the set of coherent QR derivations does too. Thus the project reported here is both foundational — seeking a deeper understanding of Quantifier Raising — and cross-disciplinary, bringing to bear the metatheoretical techniques of formal logic.

2 Adding types to Quantifier Raising

Heim & Kratzer's (1998) textbook contains lengthy and detailed discussions of Quantifier Raising from both an empirical and a technical point of view, and has served as the touchstone for Quantifier Raising for a generation of linguists and philosophers. Despite (or perhaps because of?) the centrality of the topic, they do not give a definition of Quantifier Raising. (Nor, as far as I know, has anyone else.) Nevertheless, they characterize Quantifier Raising clearly and precisely enough to provide a solid foundation for a vast amount of later work.

There is a spirit of experimentation and flexibility in Heim and Kratzer's approach. The spirit is something like this: avoid placing unnecessary formal restrictions on Quantifier Raising, since that may foreclose insightful applications to empirical problems down the line. I endorse this spirit, and one of the main takeaway messages of this paper is that Quantifier Raising requires no purely formal constraints whatsoever.

2.1 Defining Quantifier Raising and QR derivations

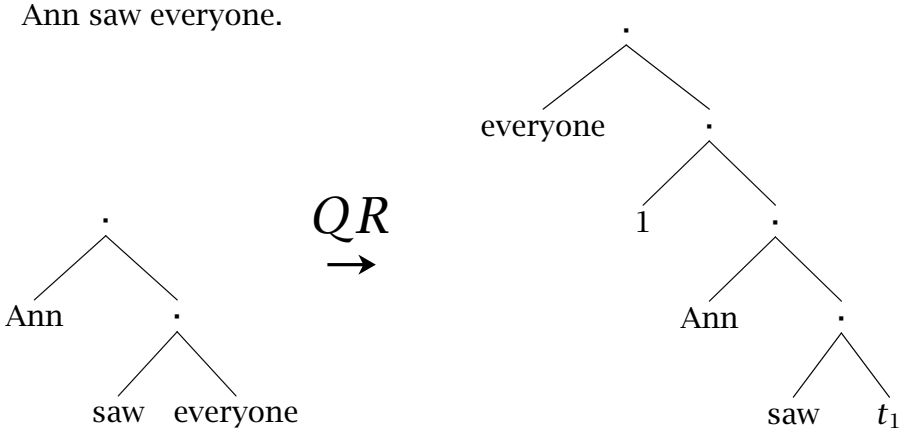
Heim & Kratzer (1998: p. 18) quote May's original 1977: p. 18 proposed rule of QR verbatim:

(1) Adjoin Q (to S)

May's rule is meant to give rise to a full specification in the context of general assumptions about grammatical theory. For instance, since adjunction is a form of movement, like all movement, QR leaves a trace that is bound by the moved expression.

Here is an example to illustrate:

(2) Ann saw everyone.



The quantifier *everyone* raises via QR to adjoin to S, leaving behind a trace t_1. In Heim & Kratzer's (1998: p. 186) treatment, the trace's index, an integer (here, '1'), is also inserted as a sibling to the adjunction site in order to provide a binder for the trace.

As schematic as (1) is, even the few specific elements it does contain are open to challenge. For instance, although May restricts adjunction to S, Heim and Kratzer motivate adjunction to VP as well. Likewise, the 'Q' in (1) is meant to cover quantificational determiner phrases, but Heim and Kratzer explicitly allow QR of non-quantificational DPs (as discussed below). Furthermore, analyses in the literature insightfully extend QR to adjectives, adverbs, comparatives, and a host of other expression types. So it seems prudent for the categorial identity of the expressions being raised, as well as the categorial identity of the adjunction site, to be left as open-ended as possible, at least as a matter of the definition of QR. In the spirit of maximizing flexibility, I will assume that QR allows an expression of any category to adjoin to a landing site of any category. If a system with such a radically unconstrained QR is nevertheless decidable with finite readings, then certainly any more highly constrained system will be as well (as long as the constraints themselves are decidable, of course).

By the way, given that QR is not limited to raising quantificational expressions, "Quantifier Raising" is not an accurate name. "Scope Taking" would be better. However, the term "Quantifier Raising" is firmly embedded in current practice.

With a view towards formalizing Quantifier Raising enough to address the formal questions of interest here, it will help to have a more precise idea of what counts as a logical form, what exactly Quantifier Raising does to logical forms, and what counts as an LF derivation.

(3) $L := W \mid L\,L \mid t_i \mid i\,L$ (**Logical Forms**)

A logical form consists of a word W (or other lexical item), or a binary branching structure consisting of two logical forms, or a trace indexed by an integer

i, or an *abstraction* consisting of an integer index i followed by a logical form.

(4) **Quantifier Raising**: Let $\sigma = [\ldots[\ldots\delta\ldots]_\gamma\ldots]$ be a logical form that contains a logical form γ that contains a logical form δ. Then Quantifier Raising applied to σ produces $[\ldots[\delta[i[\ldots t_i\ldots]_\gamma]]\ldots]$ as a result, in which γ has been replaced by a new logical form whose two daughters are δ and an abstraction, where the abstraction consists of the index i and the result of modifying the original γ by replacing δ with a coindexed trace t_i.

The index i must be chosen fresh, so that it is distinct from any other index in the original logical form.

(5) **LF derivation**: an LF derivation consists of a finite series of logical forms in which the initial logical form contains no traces or indexes, and each subsequent logical form is created from its predecessor by a single application of Quantifier Raising.

So the derivation diagramed in (2) corresponds to the following LF derivation:

(6) [Ann [saw everyone]], [everyone [1 [Ann [saw t_1]]]]

Note that the official characterization of a logical form in (3) does not provide for syntactic category labels. Since the questions addressed here concern only semantic coherence, syntactic categories are not directly relevant, though they could easily be added without affecting the results.

Just to be clear, on the terminology here, a structure that conforms to (3) but that has not undergone any QR operations (that is, a pre-QR structure) still counts as a logical form. Likewise, the proper subparts of a complete logical form also count as logical forms, with one exception: an index that helps form an abstraction structure does not by itself count as a logical form according to the specification in (3), so although Quantifier Raising can target a trace for raising (traces are logical forms), it cannot target an index.

2.2 Type coherence

With all of the desire for flexibility in the world, there is a general formal constraint on QR derivations that is non-negotiable: as Heim and Kratzer discuss, at the end of the day, after all raising and adjoining and predicate modification (and, as we'll discuss below, type shifting) is done, the final

logical form must be *interpretable*. This means in particular that the semantic types of the components of the logical form must be coherent.

We'll need to say precisely what it means for a logical form to be coherent with respect to types. Although there is reasoning about types in Heim & Kratzer 1998, there is no general method for figuring out the types of logical forms. The role of types is partially covered by making denotations partial functions that are defined only over restricted semantic domains. For instance, a denotation expression may begin "$\lambda x \in D_{\mathtt{e}} \ldots$", where $D_{\mathtt{e}}$ is the domain of individuals. Nevertheless, the way that the system has to work is clear, and the rules below are fully compatible with what I take to be Heim and Kratzer's intentions, as well as with standard practice.

I'll keep types as simple as possible, just as in Heim & Kratzer 1998: p. 28 among many others. As usual, there will be a basic type `e` for individuals and a basic type `t` for truth values, as well as functional types $A \rightarrow B$, where A and B are arbitrary types. Following common practice, types can be abbreviated: $A \rightarrow B$ will sometimes be written 'A, B' or, where no ambiguity will arise, 'AB'. Although it is common to group types using angle brackets, I'll use parentheses instead. As always, types are strictly right associative, so `eet` is the type $\mathtt{e} \rightarrow (\mathtt{e} \rightarrow \mathtt{t})$, the type of a transitive verb. Intensionality is left out of the discussion here (purely) for simplicity and clarity. Here is a summary specification of what counts as a type:

(7) $T := \mathtt{e} \mid \mathtt{t} \mid T \rightarrow T$ (**Types**)

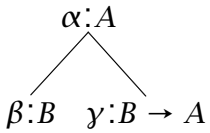
If a logical form α is a member of a type A, I'll say that α *has type* A, and write $\alpha : A$.

We can make explicit what it means for a logical form to be coherent with respect to types by supplementing Heim and Kratzer's two main rules for semantic interpretation, function application (p. 49) and Predicate Abstraction (pp. 96; 186), with typing judgments:

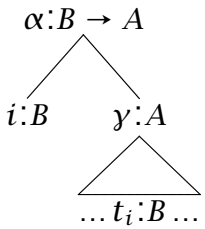
(8) **Typing rules for logical forms:**

To. A lexical item has type A just in case the lexicon says it does.

T1. Given a logical form α with two daughters β and γ (ignoring order), neither of which is an index, α will have type A whenever β has type B and γ has type $B \rightarrow A$. This typing rule corresponds to the semantic rule of function application.



T2. Given a logical form α with two daughters i and γ, where i is an index, α will have type $B \rightarrow A$ just in case γ contains exactly one trace with index i, and γ has type A whenever the coindexed trace has type B. This typing rule corresponds to the semantic rule of Predicate Abstraction.

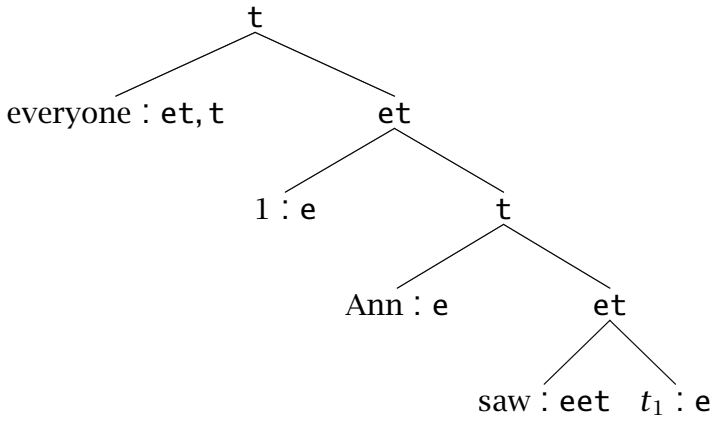


With these preliminaries, we can now say what it means for a QR derivation to be coherent with respect to types:

(9) **Coherent with respect to types**: A logical form is *coherent with respect to types* iff it is possible to label each node in the logical form with a type in such a way that each internal node of the logical form satisfies the typing rules. An LF derivation is coherent with respect to types iff its final logical form is coherent with respect to types.

For instance, the QR derivation in (6) is coherent with respect to types, since the logical form after QR is type-coherent. Here is a labeling that shows this:

(10)



In contrast, the first logical form in the derivation is not coherent with respect to types, since there is no way to combine the type of a transitive verb directly with the type of a generalized quantifier. Quantifier Raising is often motivated as a way to repair this kind of type mismatch (see, e.g., Heim & Kratzer 1998: 184 ff.). But a type mismatch is by no means a requirement for QR. For instance, QR is often called on to move quantifiers out of subject position, where there is no type mismatch. Once again, the strategy here will be

to maximize flexibility: we'll assume that QR can raise anything to anywhere. Figuring out what motivates and constrains QR remains crucially important to those grammatical theories that rely on QR, but the metatheoretical results described here are independent of such concerns. Once again, if even a maximally flexible QR is decidable, it follows that a more constrained one certainly is.

Two quick points about the flexibility of QR. First, there can be more than one coherent labeling for a given derivation. For instance, in the logical form [everyone [1 [t_1 left]]], the type of the index and the trace can either be `e` or `et,t`, leading to semantically equivalent results.

Second, the typing rules allow for a derivation on which the type of a trace cannot be determined by examining the type of the raised element that created it. For instance, in the parasitic scope derivation [A B], [A [1 [t_1 B]]], [A [B [2 [1 [t_1 t_2]]]]], the type of the trace t_1 created by raising A depends entirely on the type of the parasitic scope taker B. Parasitic scope has applications in the semantics of *same* and *different* (Barker 2007b), *average* (Kennedy & Stanley 2009), non-canonical coordination (Kubota & Levine 2015), and comparatives (Lechner 2017).

3 Stating the problems

At this point, we can state the main problems addressed in this paper.

(11) **The decidability problem for QR**: Given a logical form σ that has not yet undergone any Quantifier Raising, and a type A, is there an algorithm for deciding whether there is a QR derivation whose first logical form is σ and whose final logical form is σ_n, such that σ_n is coherent with respect to types and has type A?

Often the type A will be the type of a clause (type `t`), but we also want to be able to use QR to derive other phrase types.

This is not the same as asking, given a specific final logical form, whether there is a derivation that will produce that logical form. That would be a much simpler problem: if we have a target logical form to aim for, since every instance of QR creates a trace, we only need to count the traces in the target in order to determine the number of instances of QR needed to produce it. The problem of finding *any* coherent logical form is harder, since we don't know in advance the number of instances of QR we will need, or how they

are deployed.

(12) **The finite readings problem for QR**: Given a logical form σ and a type A, is the number of semantically distinct derivations on which the final logical form has type A finite?

Here, semantically distinct means not β-equivalent. So $\forall y \exists x.\textbf{saw}\, y\, x$ and $\exists x \forall y.\textbf{saw}\, y\, x$ are semantically distinct, but **saw ann** and $(\forall P.P\,\textbf{ann})$ **saw** are not distinct, since they are equivalent after β-reduction.

Here's a restatement of the problem: figure out what to raise to where, figure out how to type the traces, and prove the result is coherent. As if this weren't hard enough, we also have to worry about how to know when to stop: if we've already found one or more analyses, how do we know when we've found them all, so we can stop looking for more?

It may help to have a concrete example to consider:

(13) a. They ((gave them) (the (same excuse))).
b. `[e [[eeet e] [et,e [((et,et)et)et et]]]]`

Does a logical form with the lexical types as indicated have a coherent QR derivation on which it has type `t`? If so, how many distinct semantic analyses are there? (There is a hint at the end of the Appendix.)

The results below provide an algorithm that will find all semantically distinct coherent derivations. The essential move will be to translate QR and the typing rules given here into a particular Type Logical Grammar, and then leverage a metatheoretical technique pioneered by Gentzen (1935).

4 QRT, the logic of Quantifier Raising

Proving decidability and finite readings proceeds in two steps: first, defining a formal logic that characterizes the class of coherent QR derivations; and second, showing that this logic is decidable and has the finite readings property. I'll call the logic QRT, the logic of Quantifier Raising with Types. What it will mean for QRT to accurately characterize semantic coherence is that every coherent QR derivation will correspond to a semantically equivalent proof in QRT, and vice versa.

Like any formal logic, QRT involves formulas, structures built from formulas, and inference rules. The formulas of this logic will be just our set of types T as defined above in (7).

(14) $S := T \mid S \cdot S \mid t_i \mid i \cdot S$ (**Type Structures**)

Type structures are closely similar to logical forms as defined above in (3), except for two differences: type structures contain types instead of words, and the daughters of structures are unordered, since the functor and the argument in function application can occur in either order (see the discussion of type-directed interpretation in Heim & Kratzer 1998: p. 43). An example of a logical form and its corresponding type structure will illustrate:

(15) a. [**everyone** [1 [**Ann** [**saw** t_1]]]] logical form
b. $(\mathtt{et,t} \cdot (1 \cdot (\mathtt{e} \cdot (\mathtt{eet} \cdot t_1))))$ type structure

Technically, I'll assume that type structures are multisets, so the structure $\Delta \cdot \Delta'$ is formally indistinguishable from $\Delta' \cdot \Delta$. The structural punctuation mark $\cdot$ provides a visual clue that the object is a type structure rather than a logical form (mnemonic: multiplication, often written with $\cdot$, is commutative).

As for stating the inference rules of QRT, there are several standard ways to go. The most familiar are Hilbert-style axiomitizations and the Natural Deduction presentation. However, the decidability proof depends on using Gentzen's sequent presentation. A *sequent* is a structure followed by a turnstile ('$\vdash$') followed by a formula. For our purposes, sequents can be thought of as typing judgments. For instance, the sequent $\mathtt{e} \cdot \mathtt{e} \to \mathtt{t} \vdash \mathtt{t}$ can be read as "a structure consisting of a type $\mathtt{e}$ and a type $\mathtt{e} \to \mathtt{t}$ has type $\mathtt{t}$."

(16) **The inference rules of QRT:**

$$\frac{}{A \vdash A}\ \text{Axiom}$$

$$\frac{\Delta \vdash B \qquad \Sigma[A] \vdash C}{\Sigma[\Delta \cdot B \to A] \vdash C} \to L \qquad \frac{\Gamma[B] \vdash A}{i \cdot \Gamma[t_i] \vdash B \to A} \to R_i$$

(QR) $$\Gamma[\Delta] = \Delta \cdot (i \cdot \Gamma[t_i])$$

The axiom schema licenses inferring that a structure consisting of a type A has type A without needing any premises.

The next two inference rules characterize the logical content of the implication arrow ('$\to$'). The left rule ('$\to L$') has two premises: if a structure Δ has type B, and a structure Σ containing a specific occurrence of type A has type C, then we can replace the occurrence of A in Σ with a newly-created structure consisting of Δ and the type $B \to A$. This rule corresponds to the logical

form typing rule T1 above, and gives the sequent presentation of function application.

The right rule ('$\rightarrow R_i$') says that if a structure Γ containing a specific occurrence of type B has type A, then abstracting B from Γ results in a structure that has type $B \rightarrow A$. This rule corresponds to the logical form typing rule T2, and gives the sequent presentation of the operation Heim & Kratzer (1998: p. 96) call Predicate Abstraction. Just as in every version of Quantifier Raising, the index must be chosen fresh, that is, i must be distinct from any other index in Γ.

Although the $\rightarrow L$ rule given here is the standard rule of use for implication, $\rightarrow R_i$ is unusual; its relation to the standard rule is discussed below in Section 6.2.

The final inference rule ('QR') implements Quantifier Raising. This is a structural rule, rather than a logical rule. That is, it imposes a constraint on the set of structures, rather than characterizing the content of a logical connective. It says that any structure that matches one side of the equation can freely be replaced by a structure with the elements arranged as on the other side of the equation. Here are schemata that spell out the two kinds of inferences licensed by this rule, depending on the direction of the equivalence:

$$\frac{\Sigma[\Gamma[\Delta]] \vdash A}{\Sigma[\Delta \cdot (i \cdot \Gamma[t_i])] \vdash A}\,\mathrm{QR}_{\downarrow} \qquad \frac{\Sigma[\Delta \cdot (i \cdot \Gamma[t_i])] \vdash A}{\Sigma[\Gamma[\Delta]] \vdash A}\,\mathrm{QR}_{\uparrow}$$

Following Barker & Shan 2014: chapter 17 and Barker 2019, I will call $\mathrm{QR}_{\downarrow}$ 'reduction' and $\mathrm{QR}_{\uparrow}$ 'expansion'. Expansion implements Quantifier Raising (compare with (4)); reduction plays a role in a number of discussions below.

To illustrate, here's a proof justifying the judgment that *Ann saw everyone* has type t:

(17)

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\mathtt{e} \vdash \mathtt{e}}\,\mathrm{Ax} \quad \dfrac{\dfrac{}{\mathtt{e} \vdash \mathtt{e}}\,\mathrm{Ax} \quad \dfrac{}{\mathtt{t} \vdash \mathtt{t}}\,\mathrm{Ax}}{\mathtt{e} \cdot \mathtt{et} \vdash \mathtt{t}}\,{\rightarrow}L}{\mathtt{e} \cdot (\mathtt{eet} \cdot \mathtt{e}) \vdash \mathtt{t}}\,{\rightarrow}L}{1 \cdot (\mathtt{e} \cdot (\mathtt{eet} \cdot t_1)) \vdash \mathtt{et}}\,{\rightarrow}R_i \quad \dfrac{}{\mathtt{t} \vdash \mathtt{t}}\,\mathrm{Ax}}{\mathtt{et},\mathtt{t} \cdot (1 \cdot (\mathtt{e} \cdot (\mathtt{eet} \cdot t_1))) \vdash \mathtt{t}}\,{\rightarrow}L}{\mathtt{e} \cdot (\mathtt{eet} \cdot \mathtt{et},\mathtt{t}) \vdash \mathtt{t}}\,\mathrm{QR}_{\uparrow}$$

I've used the traditional abbreviations for types, so that $\mathtt{et} = \mathtt{e} \to \mathtt{t}$ is the type of a verb phrase, $\mathtt{eet} = \mathtt{e} \to \mathtt{e} \to \mathtt{t}$ is the type of a transitive verb, and $\mathtt{et,t} = (\mathtt{e} \to \mathtt{t}) \to \mathtt{t}$ is the type of a generalized quantifier.

In order to check that this proof is valid according to the inference rules just given, we can first read the proof in the direction of deductive inference, that is, from top to bottom. The only inference that does not require any premises is the axiom, so every branch of the proof must begin with an axiom instance. The first (topmost) $\to L$ inference says that a clause can consist of a subject and a predicate. The second $\to L$ inference says that a verb phrase can consist of a transitive verb and a direct object (so in this instantiation of the $\to L$ rule, $\Sigma[A] = \mathtt{e} \cdot [\mathtt{et}]$). The instance of $\to R_i$ abstracts the direct object. The third (lowest) instance of $\to L$ says that a clause can consist of a generalized quantifier combined with its nuclear scope. Finally, the $\mathrm{QR}_\uparrow$ inference drops the generalized quantifier into its surface position by replacing the structure $\mathtt{et,t} \cdot (1 \cdot (\mathtt{e} \cdot (\mathtt{eet} \cdot t_1)))$ (matching the right hand side of the structural equation) with $\mathtt{e} \cdot (\mathtt{eet} \cdot \mathtt{et,t})$ (matching the left hand side). The final conclusion shows that the structure containing the lexical types of the sentence *Ann saw everyone* has type $\mathtt{t}$.

But we can also read the proof from the bottom up, in the direction of proof search. That direction matches the normal approach to constructing a QR derivation. Starting with the desired conclusion, we ask: is $\mathtt{e} \cdot (\mathtt{eet} \cdot \mathtt{et,t}) \vdash \mathtt{t}$ a theorem? That is, does *Ann saw everyone* have a semantically coherent QR derivation on which it has type $\mathtt{t}$? We try Quantifier Raising the direct object, adjoining it to its nuclear scope. The remaining inference steps confirm that this is a winning strategy.

One of the pleasant properties of Type Logical Grammar is that the compositional semantic content of the proofs is automatically determined by the Curry-Howard correspondence. Under the correspondence, each formula in the proof receives a lambda term as a label. The label of the result type of the final conclusion gives the semantic composition of the expression as a whole based on the labels of the lexical items. According to the standard correspondence (e.g., Moortgat 1997), $\to L$ corresponds to function application, $\to R_i$ corresponds to Predicate Abstraction, and structural rules have no effect on the labeling. The net result is that this proof automatically receives the same semantic compositional content as the corresponding QR derivation given above in (10), namely, $\mathbf{everyone}(\lambda x.\mathbf{saw}\, x\, \mathbf{Ann})$.

In fact, we've already seen the Curry-Howard correspondence at work above in the typing rules for QR derivations: Heim and Kratzer's semantic

rules and the typing rules are not two things that can be artfully chosen in a way that brings them into alignment; rather, they are two aspects of a single thing. For discussion of the Curry-Howard correspondence specifically as applied to Type Logical Grammars, see Moortgat 1997, Jäger 2005, and Barker & Shan 2014: chapter 13.

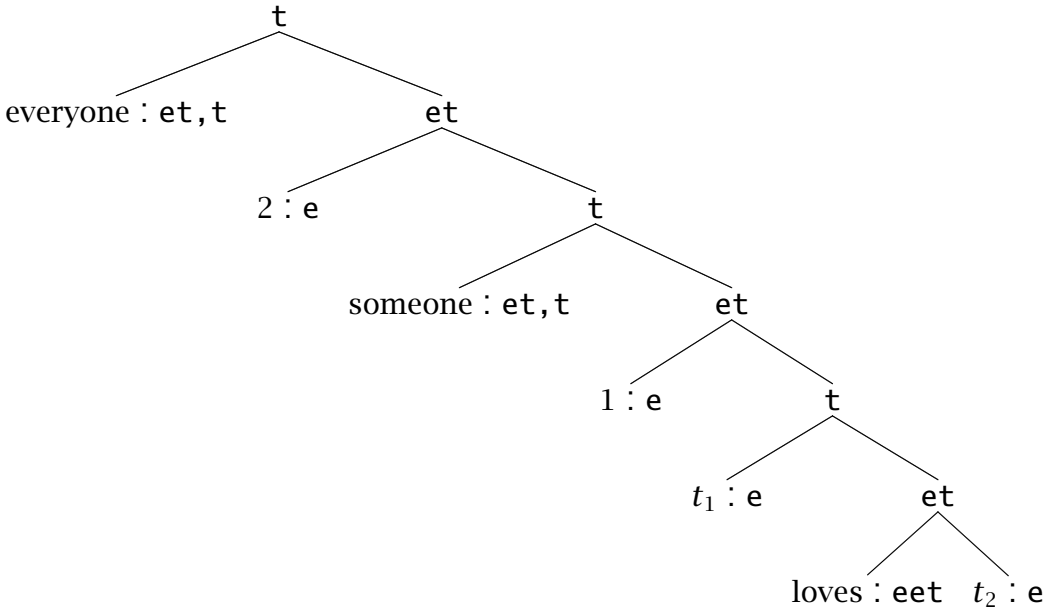
4.1 The equivalence between QR derivations and QRT

The set of analyses characterized by QR derivations and by QRT proofs are the same up to semantic equivalence: for every semantically coherent QR derivation there is a semantically equivalent QRT proof, and for every valid QRT proof there is a semantically equivalent QR derivation. Here, 'semantically equivalent' means having the same compositional semantic value, where two lambda terms related by beta reduction count as the same (e.g., $((\lambda x x)(\mathsf{e}))$ and e).

Specifying how to turn an LF derivation into a QRT proof involves building a QRT proof in two phases. The first phase is a one to one mapping between QR derivations and sequences of expansion inferences. The essential similarity between the QR operation and the QR structural rule allows a QRT proof to recapitulate an arbitrary sequence of QR operations exactly. The second phase is a correspondence between the internal nodes of a labeled logical form and logical inferences: nodes licensed by typing rule T1 correspond to $\rightarrow L$; nodes licensed by typing rule T2 correspond to $\rightarrow R_i$; certain mother-daughter pairs that are licensed by typing rules T1 and T2 correspond to $\text{QR}_{\downarrow}$; and $\text{QR}_{\uparrow}$, of course, corresponds to Quantifier Raising. Full details are given in the Appendix, and several illustrating examples are given here. For instance, the QR derivation summarized in (10) corresponds to the QRT proof in (17), and vice versa.

Here's an additional example involving a QR derivation of the inverse scope reading of *Someone loves everyone*:

(18) a. Someone loves everyone.
b. QR derivation = σ, σ_1, σ_2, where
c. σ =[someone [loves everyone]]
d. σ_1 =[someone [1 [t_1 [loves everyone]]]]
e. σ_2 =[everyone [2 [someone [1 [t_1 [loves t_2]]]]]]
f. Type labeling of σ_2:



Here is the corresponding QRT proof delivered by the algorithm in the Appendix:

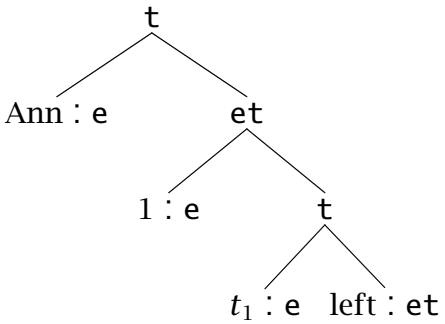
(19)

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathtt{e} \vdash \mathtt{e} \quad \mathtt{t} \vdash \mathtt{t}}{\mathtt{e} \vdash \mathtt{e} \quad \mathtt{e} \cdot \mathtt{et} \vdash \mathtt{t}} \rightarrow L}{\mathtt{e} \cdot (\mathtt{eet} \cdot \mathtt{e})) \vdash \mathtt{t}} \rightarrow L}{1 \cdot (t_1 \cdot (\mathtt{eet} \cdot \mathtt{e})) \vdash \mathtt{et} \quad \mathtt{t} \vdash \mathtt{t}} \rightarrow R_i}{\mathtt{et},\mathtt{t} \cdot (1 \cdot (t_1 \cdot (\mathtt{eet} \cdot \mathtt{e}))) \vdash \mathtt{t}} \rightarrow L}{2 \cdot (\mathtt{et},\mathtt{t} \cdot (1 \cdot (t_1 \cdot (\mathtt{eet} \cdot t_2)))) \vdash \mathtt{et} \quad \mathtt{t} \vdash \mathtt{t}} \rightarrow R_i}{\mathtt{et},\mathtt{t} \cdot (2 \cdot (\mathtt{et},\mathtt{t} \cdot (1 \cdot (t_1 \cdot (\mathtt{eet} \cdot t_2))))) \vdash \mathtt{t}} \rightarrow L}{\mathtt{et},\mathtt{t} \cdot (1 \cdot (t_1 \cdot (\mathtt{eet} \cdot \mathtt{et},\mathtt{t}))) \vdash \mathtt{t}} \mathrm{QR}_{\uparrow}}{\mathtt{et},\mathtt{t} \cdot (\mathtt{eet} \cdot \mathtt{et},\mathtt{t}) \vdash \mathtt{t}} \mathrm{QR}_{\uparrow}$$

The expansion inferences track the QR derivation exactly. More specifically, the final sequent corresponds to logical form σ, the penultimate logical form corresponds to logical form σ_1, and the third sequent from the bottom corresponds to logical form σ_2. The remaining inferences justify the claim that σ_2 is coherent with respect to types.

As mentioned, the correspondence between nodes in a labeled logical form and logical inferences is one to one, with the exception of QR↓, which corresponds to two nodes. Predicate Abstraction in a QR derivation sometimes corresponds to $\rightarrow R_i$, and sometimes to a reduction instance of the QR structural rule (QR↓), depending on whether the raised structure takes the nuclear scope as its argument, or the other way around. An example will make this clear:

(20) Ann left.



$$\dfrac{\dfrac{\dfrac{\mathtt{e} \vdash \mathtt{e} \quad \mathtt{t} \vdash \mathtt{t}}{\mathtt{e} \cdot \mathtt{et} \vdash \mathtt{t}} \rightarrow L}{\mathtt{e} \cdot (1 \cdot (t_1 \cdot \mathtt{et})) \vdash \mathtt{t}}\ \mathrm{QR}_{\downarrow}}{\mathtt{e} \cdot \mathtt{et} \vdash \mathtt{t}}\ \mathrm{QR}_{\uparrow}$$

QR of names, as shown here, is explicitly allowed by Heim & Kratzer 1998: p. 210. And why not? QR operates just like normal, and it is easy to find a labeling that satisfies the typing rules. The corresponding QRT proof in (20), delivered by the algorithm in the Appendix, is likewise valid. Performing an expansion immediately after a reduction has a null effect, of course, and if we eliminate both QR inferences, we have an equally valid proof with the same semantics. The semantic content of the QR derivation is $(\lambda x_1.\textbf{left}\, x_1)\, \textbf{ann}$, which beta reduces to the semantic content of the QRT proof, **left ann**.

Thus the QR derivation wastes semantic effort. Like the Duke of York, QR marches the subject into scope-taking position, only to have the semantics of Predicate Abstraction beta-reduce the subject back into argument position. One of the results proven in the Appendix (namely, *cut elimination*) guarantees that whenever a QRT proof contains such a semantically useless excursion, there is a semantically equivalent proof without the excursion.

The relationship between QR derivations and QRT proofs cannot be one to one. For one thing, type structures ignore linear order, so for each QRT proof, there will be many semantically equivalent QR derivations that differ only in the order of siblings within a logical form. For another, as usual with substructural logics, there are many distinct QRT proofs that are semantically equivalent. The reason is that QRT proofs differ according to the order

in which the logical inferences are executed, which can be irrelevant to the final result (see the Appendix for examples).

Within those constraints, the correspondence is as close as it could be: QRT covers the full space of semantic analyses generated by QR, adding nothing extra.

5 QR is decidable and has the finite readings property

We can now answer our two main questions affirmatively. Yes, finding out whether a semantically coherent QR derivation exists is decidable; and for any particular logical form, the number of such derivations that are semantically distinct is finite. Here's a synopsis: given that QRT is decidable and has the finite readings property, there is an algorithm that will deliver all semantically distinct QRT proofs. Since there is a coherent QR derivation for every QRT proof, we can translate the results of the algorithm into a set of QR derivations. We know that we haven't missed any semantically distinct QR derivations, since there is a semantically equivalent QRT proof for every QR derivation — and since the proof search algorithm delivers all semantically distinct QRT proofs, we can be sure we got them all.

The argument that QRT is decidable and has the finite readings property follows Gentzen's (1935) Hauptsatz. In any logic, proving a conclusion depends on proving a set of premises from which the conclusion follows. One of the distinctive advantages of presenting the logic in sequent form is that the logical rules have the *subformula* property: each formula in a premise appears (exactly once) in the conclusion. This limits the possible premises that need to be considered to a finite number. Furthermore, there is (exactly) one formula appearing in the conclusion that does not appear in any of the premises, namely, the formula created by the logical rule. Since logical connectives, once introduced, can never be eliminated, it follows that the number of logical inferences cannot exceed the number of logical connectives present in the final conclusion.

So much for the logical rules. As for the structural rule, we must treat each direction of substitution separately. The $\mathrm{QR}_{\downarrow}$ inference is easy, since it has the subformula property, and its premise contains strictly fewer structural connectives than the conclusion.

As for the $\mathrm{QR}_{\uparrow}$ inference, although it has the subformula property, its premise is not simpler than its conclusion: it contains more structural connectives than the conclusion, as well as an index and a trace that are not

present in the conclusion. In order to argue that QRT is decidable, we must show that there are limits to the number of $QR_{\uparrow}$ inferences that are needed in order to construct all semantically distinct proofs.

There are two cases to consider. The first case is when the index eliminated by the inference was introduced by an instance of $\rightarrow R_i$. Since we already know that the number of instances of $\rightarrow R_i$ is limited by the complexity of the final conclusion, the number of coindexed instances of $QR_{\uparrow}$ is limited as well.

The second case is when the index eliminated by $QR_{\uparrow}$ was introduced by an instance of $QR_{\downarrow}$ as in (20). However, it turns out that whenever this configuration occurs, there is an equivalent proof in which both inferences have been removed. This is obviously true for the proof in (20), since the two QR inferences are adjacent and have a null effect. Full details of the argument are given in Barker 2019 for a closely related logic, and carry over here with minor adjustments. In brief: it is possible to show that if we simply remove the matching instances of $QR_{\downarrow}$ and $QR_{\uparrow}$, every inference in between their original positions in the proof can be still be instantiated with the same net effect. With the exception of the two removed QR inferences, every other inference remains, in the same order. Since structural inferences don't affect the semantic labeling, the modified proof is semantically equivalent and has the same final conclusion. Since we can safely remove every instance of $QR_{\uparrow}$ in the second case, only the first case remains.[1]

Is there a particular feature of QR that is responsible for decidability? Not exactly. Rather, decidability follows from the way in which QR accomplishes non-trivial semantic work in concert with the other elements in the logical system. To see this, if we abandoned the requirement that the final logical

1 After publication of the early-access version of this paper, I discovered that this proof is incomplete due to the way that commutativity interacts with remnant movement (i.e., when $QR_{\uparrow}$ raises an abstraction structure). Rather than adding the details of the complete proof here, I will suggest reparing the proof by adopting two adjustments. The first adjustment is to replace unordered type structures (schema (14) in section 4) with ordered pairs. This would be a change in the interpretation of the structural connective $\cdot$, with no change in the appearance of any type structure. The second adjustment is to add inference rules to QRT (and to QRST, discussed below) for a new logical connective that allows the argument of an implication to appear on its right hand side. Having both left and right implications is standard in Type Logical Grammar, where the connectives are written as $A\backslash B$ and B/A. Logical proofs would remain unchanged, except that inferences that rely on structures being unordered (such as the middle instance of $\rightarrow L$ in (17)) will be replaced with $/L$ inferences, without any change to the premises or the conclusions. With these adjustments in place, the decidability argument given here is correct and complete as written.

form must be coherent with respect to types, there would be no limit to the number of QR operations. The key to decidability is the requirement that each instance of QR must have a non-trivial semantic effect. In order for QR to have a non-null semantic effect, it must interact with one of the logical rules; the only option in QRT is $\rightarrow R_i$. Since each expansion can interact with at most one instance of $\rightarrow R_i$ — after all, a given index can only be eliminated once — the fact that the number of $\rightarrow R_i$ inferences is limited means that the number corresponding expansion inferences is also limited.

Since the proof search space is finite, we not only guarantee decidability, but finite readings as well.

For the sake of concreteness, here is one especially simple algorithm for finding all semantically distinct analyses: given a sequent to be proven, try all possible ways of constructing premises from which the sequent follows by one of the inference rules. Recursively repeat the search for each premise. Abandon trying to prove any sequent in which the number of abstraction indexes exceeds the number of logical connectives.

It is important to say that having an algorithm (i.e., a method that is guaranteed to terminate) is not the same thing as having an *efficient* algorithm. There are many studies giving bounds on the time complexity of various grammatical formalisms. For example, Kuhlmann, Satta & Jonsson 2018 discusses the parsing complexity of Combinatory Categorial Grammar, a formalism with some important similarities to Type Logical Grammar. For an especially relevant investigation, Moot 2020 considers the complexity of NL_λ, a logic that is closely related to QRT (as we'll see in the next section). I make no claims here about computational complexity, except to say that the simple procedure just sketched can be vastly improved upon from the point of view of time cost. Note that the question of the computational complexity of parsing a theory that includes QR is not even well posed unless QR derivations are decidable with finite readings.

6 Type shifting and direct compositionality

This paper is about Quantifier Raising, and QRT delivers exactly standard Quantifier Raising, no more, no less. This section pulls back and takes a slightly wider view, by supplementing QR with the familiar type shifting operation LIFT.

I'll mention three reasons why this is worthwhile in the context of this paper. First, LIFT interacts with scope taking. Lifting a name of type e turns

it into a scope-seeking generalized quantifier of type `et,t`. Since any expression can potentially undergo the lift operation — including a previously lifted expression — lifting threatens decidability and finite readings. Is QR still decidable in the presence of LIFT? (Yes!)

Second, we'll see that adding LIFT to QR is parallel to adding eta reduction to the lambda calculus. This provides a new conceptual perspective on LIFT. It also helps explain why it is such a natural and indispensable type shifter, and suggests that within the class of type shifters, LIFT bears a special relationship to QR.

Third, it turns out that QR with LIFT validates the core type-shifting principles of Hendriks's (1993) Flexible Montague Grammar. Flexible Montague Grammar is a paradigm example of an in-situ account of scope taking (see, e.g., Jacobson 2012). So LIFT characterizes the difference between a pure movement approach to scope taking and an in-situ type shifting approach. This shows that it is not the presence of Quantifier Raising that forces movement in a QR theory, or that renders it non directly compositional, but rather the nature of the rest of the inferential system.

On the logic side, adding LIFT to QR derivations corresponds to adding a new structural rule ("eta") to QRT. I show that QRT + eta is equivalent to a logic that I will call QRST (Quantifier Raising with Shifty Types). Since QRST is a fragment of NL_λ (Barker 2019), it is decidable with finite readings. It follows that unrestricted QR with unrestricted lifting is also decidable with finite readings.

6.1 LIFT as eta expansion

Partee & Rooth (1983) pioneered type shifting as a technique for adjusting the types of linguistic expressions. LIFT is among their type shifters (p. 378), and is included in some form by every system I'm aware of that allows any type shifting at all. One of the more compelling advantages of LIFT is that it allows individual-denoting expressions such as proper names to have type `e`, at the same time their lifted versions can coordinate with generalized quantifiers (e.g., *Ann and every student*).

Partee (1987) says of LIFT that it "falls directly out of the type theory," and observes that it generalizes to lift expressions of type A to type AB, B for any A and B. She speculates that although it may not be universal, it should be available in any particular language "at 'low cost' or 'no cost'."

To the extent that LIFT is universal, or near-universal, we should consider whether QR remains decidable in the presence of LIFT.

Type shifters are usually framed as silent semantic operators that adjust the type and the denotation of an expression without affecting its syntactic category (see Hendriks 1993 and Winter 2007 for discussion, among others). I will suggest here a novel approach, treating LIFT as a rule of logical form, with exactly the same status as the rule of Quantifier Raising:

(21) **LIFT**: Let $\sigma = [\ldots \gamma \ldots]$ be a logical form that contains a logical form γ. Then LIFT applied to σ produces $[\ldots [i[t_i \gamma]] \ldots]$.

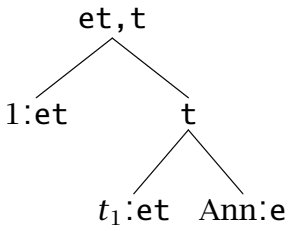
Note that the result of LIFT is a well-formed logical form, so there is no need to adjust the typing rules for LFs. We do, however, need to enlarge our set of LF derivations to include sequences of logical forms in which each logical form in the sequence is formed from the previous one by a single application of either QR or LIFT.

Here is a simple derivation in which a proper name is lifted into a generalized quantifier:

(22) Ann, $[1[t_1 \text{ Ann}]]$

And here is a type labeling for the final LF:

(23)



This logical form contains one instance of Predicate Abstraction and one instance of Function Application, so it translates into the lambda term $\lambda P.P(\textbf{ann})$ — the exact generalized quantifier delivered by the standard LIFT type shifter.

What, if anything, is special about LIFT, in comparison with all of the logical form rules we could have added to QR? An intriguing answer comes from the correspondence between QR derivations and the lambda calculus. It is obvious that Quantifier Raising closely resembles beta equivalence in the lambda calculus. In the lambda calculus, combining a lambda term with an argument results in replacing occurrences of the distinguished variable in

the body of the lambda term with a copy of the argument. This substitution operation is known as beta reduction: $(\lambda x. \ldots x \ldots) a \rightsquigarrow_\beta \ldots a \ldots$. QR, then, is the inverse operation, taking a subexpression, replacing it with a distinguished variable, and creating an abstraction structure — that is, QR is beta *expansion*. (The fact that the argument appears to the left of its functor in logical form is not important.)

The lambda calculus is a theory of functional equivalence. If one term can be derived from another via beta reduction, they are *beta equivalent*, which means one can be substituted for the other without affecting the computational result. Two functions are said to be *extensionally equivalent* just in case they deliver the same result for every choice of argument: $(\forall a. fa = ga) \rightarrow (f = g)$. There are terms in the lambda calculus that are extensionally equivalent but not beta equivalent. For instance, $\lambda x. fx$ is extensionally equivalent to f, since for any choice of argument a, $(\lambda x. fx)\, a \rightsquigarrow_\beta fa$. But $\lambda x. fx$ is not beta equivalent to f. The addition of eta reduction to the system, which says that $\lambda x. fx \rightsquigarrow_\eta f$, guarantees (Barendregt 1984: p. 63) that any two extensionally equivalent terms (that have normal forms) are provably equivalent, in that they can be reduced to the same beta-eta normal form. So eta reduction renders the lambda calculus complete with respect to extensionality.

LF is a theory of *syntactic* equivalence. If one logical form can be derived from another via Quantifier Raising, they are *QR-equivalent*, which means one can be substituted for the other without affecting the syntactic result. We'll say that two logical forms are extensionally equivalent just in case they deliver the same syntactic result for every choice of sibling: $(\forall \alpha. [\alpha\, \beta] = [\alpha\, \gamma]) \rightarrow (\beta = \gamma)$. Then $[1[t_1\ \textit{left}]]$ and *left* are extensionally equivalent, since for any choice of sibling, $[\alpha\, [1[t_1\ \textit{left}]]]$ and $[\alpha\, \textit{left}]$ are QR-equivalent. But $[1[t_1\ \textit{left}]]$ and *left* are not QR-equivalent. The addition of LIFT, which says that $[i[t_i\, \gamma]] \rightsquigarrow_{\text{LIFT}} \gamma$, guarantees that any two extensionally equivalent logical forms are provably equivalent, that is, they can be derived via QR and LIFT from the same starting logical form. So LIFT renders LF complete with respect to extensional equivalence.

Note that LF is a theory of *syntactic* equivalence, not semantic equivalence. For instance, the two semantically distinct scope readings of *Someone saw everyone*, namely [someone[1[everyone[2[t_1[saw t_2]]]]]] and [everyone[2[someone[1[t_1[saw t_2]]]]]], are QR-equivalent, since they can both be derived via QR from the same syntactic structure, namely, [someone[saw everyone]].

If you want your lambda calculus to fully characterize functional extensional equivalence, you need both beta reduction and eta reduction. Likewise, if you want your LF theory to fully characterize syntactic extensional equivalence, you need both Quantifier Raising and LIFT. This suggests that adding LIFT to QR derivations is as natural as adding eta reduction to the lambda calculus. So LIFT is related to QR in a uniquely natural and intimate way.

6.2 QRT + eta = QRST

Adding LIFT to QR derivations corresponds to adding a second structural rule to QRT:

(24) $i \cdot (t_i \cdot \Delta) \Rightarrow \Delta$ (eta)

This rule says that any structure of the form $i \cdot (t_i \cdot \Delta)$ in a premise can be replaced by Δ in the conclusion.

The same reasoning that justified decidability for QRT can be extended to QRT + eta. However, we can gain additional insight by an indirect approach. I'll show that QRT + eta is equivalent to a logic I call QRST (Quantifier Raising with Shifty Types). Since QRST is a fragment of NL_λ, and Barker (2019) proves that NL_λ is decidable with finite readings, it follows that QRT + eta is decidable and has the finite readings property.

QRST has the same set of types as QRT, and the same structures.

(25) **The inference rules of QRST:**

$$\frac{}{A \vdash A}\,\text{Axiom}$$

$$\frac{\Delta \vdash B \qquad \Sigma[A] \vdash C}{\Sigma[\Delta \cdot B \rightarrow A] \vdash C} \rightarrow L \qquad \frac{B \cdot \Gamma \vdash A}{\Gamma \vdash B \rightarrow A} \rightarrow R$$

(QR) $$\Gamma[\Delta] = \Delta \cdot (i \cdot \Gamma[t_i])$$

Comparison with (16) reveals that QRT is identical to QRST except that the $\rightarrow R_i$ rule in QRT is replaced with a simpler rule $\rightarrow R$ in QRST:

(26)

$$\frac{\Gamma[B] \vdash A}{i \cdot \Gamma[t_i] \vdash B \rightarrow A} \rightarrow R_i \qquad \frac{B \cdot \Gamma \vdash A}{\Gamma \vdash B \rightarrow A} \rightarrow R$$

The original rule given above for QRT is on the left. The modified rule is the standard right rule for → in Gentzen's (1935) system LJ, which is a sequent presentation of intuitionistic logic. This same rule of proof for implication is used throughout the Type Logical Grammar literature.

It's easy to see that every QRST proof can be reproduced by QRT + eta, thanks to the following equivalence:

(27)

$$\dfrac{\dfrac{B \cdot \Gamma \vdash A}{i \cdot (t_i \cdot \Gamma) \vdash B \to A}\,{\to}R_i}{\Gamma \vdash B \to A}\,\text{eta} \qquad \equiv \qquad \dfrac{B \cdot \Gamma \vdash A}{\Gamma \vdash B \to A}\,{\to}R$$

In the other direction, any instance of $\to R_i$ in a QRT proof can be reproduced in QRST using the following inferences:

(28)

$$\dfrac{\Gamma[B] \vdash A}{i \cdot \Gamma[t_i] \vdash B \to A}\,{\to}R_i \qquad \equiv \qquad \dfrac{\dfrac{\Gamma[B] \vdash A}{B \cdot (i \cdot \Gamma[t_i]) \vdash A}\,\text{QR}_{\downarrow}}{i \cdot \Gamma[t_i] \vdash B \to A}\,{\to}R$$

This equivalence reveals that the QRT rule $\to R_i$ is a combination of the standard rule with an instance of $\text{QR}_{\downarrow}$. That is, $\to R_i$ has some structural reasoning baked in. Because of this conflation of logical and structural inference, the QRT version is able to limit the availability of →-right inferences to situations in which an abstraction created by $\text{QR}_{\uparrow}$ is present. This restriction is precisely what limits QRT to Quantifier Raising with nothing added.

In order to establish the equivalence between QRT + eta and QRST, it remains only to show that proofs in QRT + eta that contain eta inferences can be faithfully simulated in QRST. Consider a QRT + eta proof that contains an eta inference. First, note that there are only two ways that a structure of the form $i \cdot (t_i \cdot \Gamma)$ can be created in a conclusion: by an instance of the QR structural rule, or by an instance of $\to R_i$. If an instance of QR is immediately followed by eta reduction, the net result of the two inferences is no change, and the pair of inferences can be safely omitted. If an instance of $\to R_i$ is immediately followed by eta reduction, this is exactly an $\to R$ inference, as shown in (27). In all other cases, the order of an inference immediately followed by eta reduction can be exchanged without affecting the final conclusion. It follows

that any instance of eta reduction can be moved higher in the proof until it meets the instance of either QR or $\rightarrow R_i$ that introduced the relevant index. So every proof in QRT + eta has a semantically equivalent proof in QRST, and vice versa.

Generalized lifting is straightforward in QRST:

$$\dfrac{\dfrac{\mathtt{e} \vdash \mathtt{e} \quad \mathtt{t} \vdash \mathtt{t}}{\mathtt{et} \cdot \mathtt{e} \vdash \mathtt{t}} \rightarrow L}{\mathtt{e} \vdash \mathtt{et,t}} \rightarrow R$$

The Curry-Howard labeling for this proof gives exactly the generalized quantifier semantics of Partee's LIFT type shifter. The reasoning here is fully general, and not specific to type e — any types A and B can be substituted here for e and t.

As mentioned, QRST is a fragment of NL_λ (discussed in Barker 2019) with the structural rule Exchange added. Since NL_λ is decidable and has the finite readings property, and adding Exchange does not change these results, the modified logic here is also decidable and has the finite readings property.

This means that we can freely allow unrestricted generalized type lifting without compromising the formal properties of QR derivations. On the QR derivation side, we can add the LIFT rule given above. On the logic side, we can either use QRT with a structural rule of eta reduction added, or else QRST, which has the standard rule of proof for implication. One additional consideration in favor of going with QRST (besides having a standard rule of proof for implication) is that there is a sound and complete interpretation for QRST (see Barker 2019).

6.3 In-situ scope and direct compositionality

A semantic theory is *directly compositional* only if every syntactic constituent has a well-defined semantic value. Here is what Barker & Jacobson (2007: p. 2) say about Quantifier Raising:

> [In the standard analysis using Quantifier Raising,] a verb phrase such as *saw everyone* fails to have a semantic interpretation until it has been embedded within a large enough structure for the quantifier to raise and take scope (e.g., *Someone saw everyone*). On such an analysis, there is no semantic value to assign to the verb phrase *saw everyone* at the point in the derivation

> in which it is first formed by the syntax (or at any other point in the derivation, for that matter). A directly compositional analysis, by contrast, is forced to provide a semantic value for any expression that is recognized as a constituent in the syntax. Thus if there are good reasons to believe that *saw everyone* is a syntactic constituent, then a directly compositional analysis must provide it with a meaning.

And indeed, if we examine either the QR derivation of *Ann saw everyone* in (10) or the corresponding QRT proof in (17), there is no stage at which the structure corresponding to *saw everyone* is established as a constituent: there is no type associated with that particular substructure either in the QR derivation or in the corresponding QRT proof, and no Curry-Howard labeling that contains the semantic contribution of *saw* and *everyone* and nothing else.

There are good reasons, of course, to suppose that verb phrases such as *saw everyone* are constituents. For instance, this particular verb phrase can serve as the antecedent of verb phrase ellipsis, as in *Ann saw everyone, and Bill did too*, on the interpretation on which Bill saw everyone. In the kind of directly compositional system that Barker and Jacobson have in mind, there will be a semantic value computed for the structure *saw everyone* that will conveniently make salient the semantic value captured by the ellipsis.

Quantifier Raising contrasts in this regard with Hendriks's (1993) Flexible Montague Grammar. In that system, expressions take displaced scope entirely through strategically deployed type shifting, without any movement or other structural reconfiguration. The two main type shifters are Argument Raising, which allows an arbitrary argument of a predicate to take scope over the other arguments of that predicate, and Value Raising, which lifts the result type of a predicate into a type suitable for taking scope. (There is a third type shifting rule that deals with de dicto/de re ambiguities that I will not discuss here.) Flexible Montague Grammar is often held up as a paradigm example of a directly compositional approach to scope taking (e.g., Jacobson 2012).

It turns out that all instances of Argument Raising and Value Raising are theorems in QRST. To illustrate, Argument Raising applied to the first argument of an extensional transitive verb (type `eet`) creates a shifted verb

that takes a generalized quantifier direct object (type `(et,t)et`):

(29)

$$\dfrac{\dfrac{\dfrac{\vdots}{\mathtt{et,t}\cdot(1\cdot(\mathtt{e}\cdot(\mathtt{eet}\cdot t_1)))\vdash \mathtt{t}}\,\mathrm{QR}_{\uparrow}}{\mathtt{e}\cdot(\mathtt{eet}\cdot\mathtt{et,t})\vdash \mathtt{t}}\to R}{\mathtt{eet}\cdot\mathtt{et,t}\vdash \mathtt{et}}\to R}{\mathtt{eet}\vdash \mathtt{(et,t)et}}$$

Reading from the bottom up, the proof pushes each of the argument types of the conclusion result across the turnstile, building a structure consisting of the extensional verb and its arguments. The generalized quantifier argument undergoes QR to take scope over the clause created by the saturated verb, and the rest of the proof proceeds as in, e.g., (19). The Curry-Howard labeling is exactly the same as the shifted meaning given in Flexible Montague Grammar, that is, $\lambda Q\lambda x.Q(\lambda y.\mathbf{saw}\,y\,x)$, where Q is a generalized quantifier and **saw** is the meaning of the extensional transitive verb in question.

The fact that QRST validates Argument Raising and Value Raising means that we can prove that the verb phrase *saw everyone* has each of the types (and their corresponding denotations) that Flexible Montague Grammar gives it. For instance, here is a proof that *saw everyone* is a predicate of type `et`:

(30)

$$\dfrac{\dfrac{\dfrac{\vdots}{\mathtt{et,t}\cdot(1\cdot(\mathtt{e}\cdot(\mathtt{eet}\cdot t_1)))\vdash \mathtt{t}}\,\mathrm{QR}_{\uparrow}}{\mathtt{e}\cdot(\mathtt{eet}\cdot\mathtt{et,t})\vdash \mathtt{t}}\to R}{\mathtt{eet}\cdot\mathtt{et,t}\vdash \mathtt{et}}\to R$$

The Curry-Howard labeling is $\lambda x.\mathbf{everyone}(\lambda y.\mathbf{saw}\,y\,x)$. When this verb phrase meaning is applied to the denotation of *Ann* (or *Bill*), the result is a proposition that entails that everyone was seen by Ann (or Bill), as desired.

QRST is not a *strictly* directly compositional theory, since there are plenty of derivations on which some syntactic structure does not receive a semantic value, as we have seen. Nevertheless, we can compute the missing semantic interpretations whenever desired. Barker (2007a) calls this kind of situation

'Direct Compositionality on Demand'. In the same spirit, we can reconstruct the same type-shifted interpretations delivered by Flexible Montague Grammar whenever desired, so we can also consider QRST to provide in-situ scope taking on demand.

By the way, the fact that QRST is decidable with finite readings, along with the fact that every Flexible Montague Grammar derivation can be reproduced by a QRST proof means that (the Argument Raising/Value Raising core of) Flexible Montague Grammar is also decidable with finite readings. As far as I know, this is the first time that fact has been established.

What should we make of this situation? There is no difference between QRT and QRST with respect to the structural rule that encodes scope-taking (the QR rule). Yet in bare QRT, scope taking always requires movement and is not directly compositional; but with the addition of LIFT (equivalently, replacing $\rightarrow R_i$ with the standard $\rightarrow R$), we have in-situ analyses and direct compositionality on demand. Apparently, the difference between a pure movement theory of scope and an in-situ, directly compositional theory resides not in the conception or the implementation of scope-taking, but in the nature of the larger inferential system in which the scope taking analysis is embedded.

One way to see this is to note that adding LIFT is not the only way to embed QRT in a directly compositional system. For instance, another strategy is to add a conjunction to the logic. After all, many logics (to put it mildly) have at least one conjunction in addition to implication. In particular, Type Logical grammars routinely include a conjunction, starting with Lambek 1958 (see Moortgat 1997).

Without going into complete detail, here's how it works. First, we expand the set of types to include $A \wedge B$ for all types A and B. Then $\text{QRT}_\wedge$ (QRT with conjunction) is QRT plus the following two standard logical inference rules:

(31)
$$\frac{\Sigma[A \cdot B] \vdash C}{\Sigma[A \wedge B] \vdash C} \wedge L \qquad \frac{\Sigma \vdash A \qquad \Gamma \vdash B}{\Sigma \cdot \Gamma \vdash A \wedge B} \wedge R$$

The interpolation theorem of Barker 2019 holds for $\text{QRT}_\wedge$: given any theorem $\Sigma[\Delta] \vdash A$, there is a type B such that $\Delta \vdash B$ and $\Sigma[B] \vdash A$. That is, given any specific proof, it is possible to assign an arbitrary substructure Δ a type B that characterizes its role in that proof.

For instance, the proof in (17) establishes that *Ann (saw everyone)* is a clause, that is, that $\texttt{e} \cdot (\texttt{eet} \cdot \texttt{et}, \texttt{t}) \vdash \texttt{t}$. The construction given by the inter-

polation theorem proposes $\mathtt{et,t} \wedge \mathtt{eet}$ as the type of the verb phrase.

(32)
$$\dfrac{\dfrac{\mathtt{et,t} \vdash \mathtt{et,t} \quad \dfrac{\dfrac{\mathtt{e} \vdash \mathtt{e} \quad \mathtt{et} \vdash \mathtt{et}}{\mathtt{eet} \cdot \mathtt{e} \vdash \mathtt{et}} \to L}{1 \cdot (\mathtt{eet} \cdot t_1) \vdash \mathtt{eet}} \to R_i}{\mathtt{et,t} \cdot (1 \cdot (\mathtt{eet} \cdot t_1)) \vdash \mathtt{et,t} \wedge \mathtt{eet}} \wedge R}{\mathtt{eet} \cdot \mathtt{et,t} \vdash \mathtt{et,t} \wedge \mathtt{eet}} \mathrm{QR} \qquad \dfrac{\dfrac{\dfrac{\vdots}{\mathtt{e} \cdot (\mathtt{e} \cdot \mathtt{eet}) \vdash \mathtt{t}}}{2 \cdot (\mathtt{e} \cdot (t_2 \cdot \mathtt{eet})) \vdash \mathtt{et}} \to R_i \quad \mathtt{t} \vdash \mathtt{t}}{\mathtt{et,t} \cdot (2 \cdot (\mathtt{e} \cdot (t_2 \cdot \mathtt{eet}))) \vdash \mathtt{t}} \to L}{\mathtt{e} \cdot (\mathtt{et,t} \cdot \mathtt{eet}) \vdash \mathtt{t}} \mathrm{QR}}{\mathtt{e} \cdot \mathtt{et,t} \wedge \mathtt{eet} \vdash \mathtt{t}} \wedge L$$
$$\mathtt{e} \cdot (\mathtt{eet} \cdot \mathtt{et,t}) \vdash \mathtt{t} \quad \mathrm{cut}$$

See Appendix A.1 for a discussion of cut inferences. Using the standard Curry-Howard semantics for conjunction, the semantic labeling for the verb phrase (the left hand branch of the proof) is the ordered pair $\langle$**everyone**, $\lambda x.$**saw** $x\rangle$, which means the label for the final conclusion is **everyone**$(\lambda y.$ $(\lambda x.$**saw** $x)$ y **ann**$)$, which beta reduces to **everyone**$(\lambda y.$**saw** y **ann**$)$ as usual.

The interpolation theorem allows breaking up an arbitrary QR movement into a series of strictly local hops. The addition of conjunction to the grammar allows these intermediate stopping points to be given a type and a Curry-Howard labeling that accurately tracks the non-local version of the derivation.

The conjunction approach matches the expressive power of QR more closely than Flexible Montague Grammar. Not only does Flexible Montague Grammar allow analyses that QRT does not (such as lifting), it fails to allow analyses that QRT does, such as parasitic scope (as noted by Barker & Shan 2014: p. 70). In contrast to the addition of LIFT, adding a conjunction to QRT is conservative, in the sense that there are no (conjunction-free) sequents that are theorems of $\mathrm{QRT}_\wedge$ that were not already theorems of plain QRT. So $\mathrm{QRT}_\wedge$ is pure QR but with direct compositionality on demand.

In any case, QRST and $\mathrm{QRT}_\wedge$ show that two different ways of extending QRT can supply direct compositionality on demand. This means that whether a grammar is directly compositional — that is, whether it provides a type and an interpretation for every syntactic structure — does not depend on whether it allows a long-distance movement operation such as QR, but depends instead on properties of the system as a whole.

7 Four additional issues

7.1 Unbound traces

As May noticed, unconstrained Quantifier Raising can create unbound traces. This can happen when material containing the trace of a previously raised scope taker raises higher than the lambda that binds the trace. The traditional solution for Quantifier Raising is to simply prohibit unbound traces. There is no need to take any special action here, since any QR derivation that creates an unbound trace will not be semantically coherent. Likewise, in QRT, using $\text{QR}_{\uparrow}$ to create unbound traces will never lead to a complete and valid proof.

7.2 Higher-order traces

There are many analyses that rely on higher-order traces, including semantic reconstruction (e.g., Cresti 1995, Barker & Shan 2014) and split-scope analyses (German *kein* (Jacobs 1980), donkey anaphora (Barker & Shan 2014), Haddock sentences (Bumford 2017), and cumulative readings (Charlow 2020, to appear)). Higher-order traces are perfectly compatible with the system here. See Charlow 2020 for a discussion of the details and the trade-offs of having higher-order traces in a Quantifier Raising analysis.

7.3 Quantifier Raising is syntactic

As discussed in Section 6.1, LF is a theory of syntactic equivalence: a quantifier in-situ in its surface position and the corresponding logical form created by QR are *syntactically* equivalent. Likewise, on the logical side, the bidirectional structural equation in QRT that characterizes Quantifier Raising is a structural inference rule. Put another way, recall that the QR structural rule does not affect semantic labeling at all, since the Curry-Howard correspondence ignores structural inferences. It follows that displaced scope is an essentially, purely syntactic phenomenon, on a par with other grammatical phenomena that correspond to structural rules, such as scrambling (corresponding to the structural rule of Exchange) or so-called non-constituent coordination (corresponding to the structural rule of associativity).

7.4 The logic of movement?

If computing covert scope analyses is decidable, what about overt movement? QRST is a fragment of NL_λ (with Exchange added), a substructural logic first proposed in Barker 2007b. As shown in Barker & Shan 2014 and in Barker 2019, NL_λ is able to account not only for in-situ scope-taking, but syntactic movement as well. Because NL_λ is also decidable with finite readings, it shows how to combine syntactic movement and scope-taking in a single unified grammar that is computationally well-behaved. A thorough exploration of the logic of overt movement along the lines of this paper will have to wait for another occasion.

8 Conclusion

Quantifier Raising has long been the standard tool for analyzing displaced scope in natural language. When Quantifier Raising is combined with an explicit method for checking type compatibility, it is decidable, and provides a strictly finite number of distinct semantic interpretations for any given expression, even in the presence of type lifting. These results taken together justify full confidence in Quantifier Raising as a coherent and formally well-behaved technique for analyzing scope.

A Appendix

This Appendix proves cut elimination for QRT, and gives details of the mappings from QR derivations to QRT and back again. It also gives a hint for the problem posed in (13).

A.1 Cut elimination

A standard inference rule now enters the story, the *cut* rule.

$$\text{(33)}\qquad \frac{\Delta \vdash A \qquad \Sigma[A] \vdash B}{\Sigma[\Delta] \vdash B}\ \text{cut}$$

The cut rule is valid in every logic, and expresses the transitivity of deduction. (Well, *almost* every logic — see Weir 2015 for a discussion of non-transitive logics that address certain paradoxes. Yet even these logics endorse a restricted version of cut.) The cut rule says that if a type structure Δ has type

A, we can safely replace any occurrence of A in some other proof with the material in Δ without affecting the larger proof.

As usual for decidability proofs in logic, the decidability of QRT will hinge on proving that the cut rule is *admissible* (redundant): that any theorem provable using the cut rule can be proven without using cut.

(34) **Cut elimination**. Given an arbitrary QRT proof, there is an equivalent proof that does not contain any cut inferences.

'Equivalent' here means same final sequent, and equivalent Curry-Howard semantic labeling up to beta reduction.

The proof here is almost completely standard (see especially Gentzen 1935, Restall 2000, Jäger 2005: p. 41). Some vocabulary: the *cut formula* is the formula shared by the two premises of the cut inference (the formula matching A in the schema above), and the *active formula* of a logical inference is the formula created in the conclusion that is not present in any of the premises.

As in all cut elimination proofs, the general strategy here is to push each cut inference higher in the proof until it reaches an axiom. When one premise of a cut is an axiom, the other premise must be identical to the conclusion, and the cut can be safely eliminated.

The structural rules do not impede pushing cut inferences higher in the proof. Since structural rules do not add or subtract formulas, the cut formula will be present in the premise of the structural inference, so the order of the structural inference and the cut can always be swapped.

The logical rules have the subformula property, which means that every formula in the conclusion appears in (exactly) one of the premises, with the exception of the active formula. As a result, whenever the cut formula is not the active formula in one of the premises of a cut inference, it is possible to push the cut upwards.

The only remaining case to consider is when the cut formula is the active formula in both of the premises (a *principal cut*).

$$\dfrac{\dfrac{\Gamma[B] \vdash A}{i \cdot \Gamma[t_i] \vdash B \to A} \to R_i \quad \dfrac{\Delta \vdash B \qquad \Sigma[A] \vdash C}{\Sigma[\Delta \cdot B \to A] \vdash C} \to L}{\Sigma[\Delta \cdot (i \cdot \Gamma[t_i])] \vdash C} \text{ cut}$$

The cut formula is $B \to A$, which is the active formula for both the $\to R_i$ inference and the $\to L$ inference. This cut can be transformed into a pair of

smaller cuts using the same initial premises and arriving at the same final conclusion.

$$\dfrac{\dfrac{\Delta \vdash B \qquad \dfrac{\Gamma[B] \vdash A \qquad \Sigma[A] \vdash C}{\Sigma[\Gamma[B]] \vdash C}\,\text{cut}}{\Sigma[\Gamma[\Delta]] \vdash C}\,\text{cut}}{\Sigma[\Delta \cdot (i \cdot \Gamma[t_i])] \vdash C}\,\text{QR}_{\downarrow}$$

Here, 'smaller' refers to the total number of base types and logical connectives in the premises; see, e.g., Jäger 2005: p. 43 for a more detailed definition. The only non-standard wrinkle in the proof is that the transformed reasoning with smaller cuts requires the addition of a $\text{QR}_{\downarrow}$ inference. The reason is that the refactoring eliminates an instance of $\rightarrow R_i$, and since the $\rightarrow R_i$ rule in effect incorporates an instance of reduction (as discussed in Section 6.2), it is necessary to add a compensating reduction inference in the replacement proof fragment.

A.2 Every QR derivation has an equivalent QRT proof

Assume we have a semantically coherent QR derivation $\sigma, \sigma_1, \ldots, \sigma_n$ where σ_n has type A. Our goal is to build a QRT proof of $\Sigma \vdash A$, where Σ is a type structure corresponding to σ.

We build the proof starting from the bottommost conclusion and working upwards. Building the proof divides into two phases. The first phase tracks the instances of Quantifier Raising that constitute the QR derivation. The initial (lowest) sequent is $\Sigma \vdash A$, the next is $\Sigma_1 \vdash A$, and so on up to $\Sigma_n \vdash A$. Each sequent is related to the one below it by an expansion inference that exactly matches the corresponding instance of QR from the logical form derivation.

The second phase uses the type labeling of σ_n to guide the instantiation of the inference rules needed to prove $\Sigma_n \vdash A$. The construction proceeds recursively based on two parameters: a labeled logical form π which is a part of the logical form σ_n, and a type structure Π, which is the part of Σ_n corresponding to π. Phase 2 begins with $\pi = \sigma_n$, and $\Pi = \Sigma_n$. Note that the initial values for π and Π are isomorphic up to the order of siblings, and each recursive application of the construction will maintain that isomorphism. Then there are three cases, corresponding to the three typing rules:

To. π consists of a single word, in which case the sequent to be proven has the form $P' \vdash P$, where P' is a single type. By construction, P is the label of the (only) lexical item in π. So $P = P'$, and we have an instance of the axiom inference rule.

T1. π consists of two daughters, δ and γ, where neither is an index, and $\Pi = \Delta \cdot \Gamma$. Because π satisfies the typing rules, we can assume that δ has type B and that γ has type $B \rightarrow A$ (or vice versa, with small changes below). We extend the proof upwards as follows:

$$\dfrac{\Gamma \vdash B \rightarrow A \qquad \dfrac{\Delta \vdash B \qquad A \vdash A}{\Delta \cdot B \rightarrow A \vdash A} \rightarrow L}{\Delta \cdot \Gamma \vdash A} \text{cut}$$

We've now reduced proving $\Delta \cdot \Gamma \vdash A$ into two strictly smaller problems, namely, proving $\Delta \vdash B$ and proving $\Gamma \vdash B \rightarrow A$. We recursively call the construction algorithm twice: once with $\pi = \delta$ and $\Pi = \Delta$, and again with $\pi = \gamma$ and $\Pi = \Gamma$.

T2. π is an abstraction with form $i\,\gamma[t_i]$, and $\Pi = i \cdot \Gamma[t_i]$, where i is an index. We instantiate the $\rightarrow R_i$ rule to extend the proof upwards, where $B \rightarrow A$ is the label at the root of π:

$$\dfrac{\Gamma[B] \vdash A}{i \cdot \Gamma[t_i] \vdash B \rightarrow A} \rightarrow R_i$$

We now recursively call the construction algorithm with $\pi = \gamma[t_i]$ and $\Pi = \Gamma[B]$. The typing rule for abstractions guarantees that γ has type A when t_i has type B, so we can be sure that the labeling of γ will guide construction of a valid proof of $\Gamma[B] \vdash A$.

These three cases show that we can always divide up the second phase of proof building into strictly smaller problems. Since Σ is of finite complexity, we will eventually reach a point at which either Δ or Γ is a single type, so every subproblem will terminate in axiom instances.

The official inference rules for QRT given in (16) do not contain cut, and the proof construction just given does contain cut; but of course the method described in Section A.1 explains how to arrive at an equivalent cut-free proof.

A.3 Pushing structural inferences lower

QRT places no restrictions on when a QR inference can occur, so establishing the correspondence between QRT proofs and QR derivations depends on pushing structural inferences lower in the proof.

(35) **Fact**: whenever the conclusion of a QR instance is a premise of a logical inference, the order of the inferences can be reversed without affecting the proof.

To see why, consider first all possible configurations in which the conclusion of an expansion inference is a premise of a following logical inference. In each case, the order of the inferences can be reversed without affecting the initial or the final sequent. For instance, if the expansion affects the first premise of an instance of $\rightarrow L$, the expansion can safely be delayed till after the logical rule. If the expansion affects the second premise, there are two subcases: the raised element is the distinguished occurrence of A, or not. If not, the expansion can be delayed, in which case the delayed expansion simply raises the structure $\Delta \cdot B \rightarrow A$ instead of A. If the expansion affects the premise of a $\rightarrow R_i$ inference, there are two subcases: the raised element is B, or not. If not, the expansion can clearly be delayed. If so, the delayed expansion raises the trace t_i instead of B.

Here is an illustration of the last subcase:

$$\dfrac{\dfrac{\Sigma[B \cdot (j \cdot \Pi[t_j])] \vdash A}{\Sigma[\Pi[B]] \vdash A}\,\mathrm{QR}_{\uparrow}}{i \cdot \Sigma[\Pi[t_i]] \vdash B \rightarrow A}\,{\rightarrow}R_i \quad \equiv \quad \dfrac{\dfrac{\Sigma[B \cdot (j \cdot \Pi[t_j])] \vdash A}{i \cdot \Sigma[t_i \cdot (j \cdot \Pi[t_j])] \vdash B \rightarrow A}\,{\rightarrow}R_i}{i \cdot \Sigma[\Pi[t_i]] \vdash B \rightarrow A}\,\mathrm{QR}_{\uparrow}$$

The beginning and ending sequents for the unswapped inferences on the left are identical to the beginning and ending sequents for the swapped inferences.

A similar argument holds for swapping reduction inferences with logical inferences.

A.4 Every QRT proof has an equivalent QR derivation

Let p be any QRT proof whose final conclusion is $\Sigma \vdash A$, where Σ does not contain any abstraction structures. Our goal is to use p to find a semantically coherent QR derivation $\sigma, \sigma_1, \ldots, \sigma_n$ such that Σ is a type structure for σ and σ_n has type A.

Requiring the final sequent of p to be abstraction-free is parallel to requiring that QR derivations begin with a logical form that has not yet undergone any Quantifier Raising operations. This restriction can be relaxed, but in order to maintain the correspondence between QRT proofs and QR derivations, we would also need to generalize QR derivations to include sequences of logical forms in which the initial logical form has already undergone some number of Quantifier Raising operations. The decidability result and the finite readings result would continue to apply.

Relying on the results in the previous sections of the Appendix and the decidability result for QRT, we begin by replacing p with an equivalent proof p' that is cut-free, in which each $\text{QR}_\uparrow$ inference follows every logical inference, and in which all coindexed $\text{QR}_\downarrow/\text{QR}_\uparrow$ inference pairs have been eliminated (along the lines sketched in the decidability proof). After these adjustments, all $\text{QR}_\uparrow$ inferences will be gathered together as the final n inferences of p'. Then the $\text{QR}_\uparrow$ inferences in p' induce a QR derivation $\sigma, \sigma_1, \ldots, \sigma_n$ such that each Σ_i is a type structure for σ_i, and each logical form is related to the previous one by an application of QR in lock step with the corresponding expansion inferences that relate the structures in the sequence $\Sigma, \Sigma_1, \ldots, \Sigma_n$.

In order to demonstrate that this is a semantically coherent QR derivation, we must show how p' determines a type labeling for σ_n that satisfies the typing rules and on which σ_n has type A. Because σ_n and Σ_n are isomorphic (up to sibling order), it will be convenient to associate labels with the structure of Σ_n, relying on the isomorphism to map the labels onto the corresponding nodes of σ_n.

Consider the portion of p' that proves $\Sigma_n \vdash A$, that is, the portion of p' up to but not including the final expansion inferences. Let p'' be this portion of p'. The argument proceeds by induction on the structure of p''.

The base case is when p'' consists of an axiom inference of the form $A \vdash A$. Any logical form labeled with the left hand side (trivially) has the result type, so we have a labeling for σ_n.

For the inductive case of the argument, consider the final inference in p''. By the recursive assumption, we can assume that we have a suitable labeling for each premise. That is, if the premise is $\Delta \vdash B$, we can assume that we have a labeling for the structure Δ that satisfies the typing rules such that the root of Δ has label B. We need to prove that $\Delta_n \vdash A$. Since all expansion inferences have already been pushed lower in the proof, there are three subcases to consider: $\rightarrow L$, $\rightarrow R_i$, and $\text{QR}_\downarrow$. We reason as follows:

$\to L$:
$$\frac{\Delta \vdash B \qquad \Sigma[A] \vdash C}{\Sigma[\Delta \cdot B \to A] \vdash C} \to L$$

By the inductive assumption applied to the left premise, we have a labeling of Δ on which it has type B according to the typing rules. Then the newly-created substructure $(\Delta \cdot B \to A)$ has type A, by virtue of the typing rule T1, so we add the label A to the newly-created substructure. By the inductive assumption applied to the right premise, there is now a complete labeling of the conclusion sequent on which it has type C.

$\to R_i$:
$$\frac{\Gamma[B] \vdash A}{i \cdot \Gamma[t_i] \vdash B \to A} \to R_i$$

Since the labeling of $\Gamma[B]$ has type A, the newly created structure $i \cdot \Gamma[t_i]$ has type $B \to A$ by virtue of the typing rule T2. We label the new structure accordingly.

$\text{QR}\downarrow$:
$$\frac{\Sigma[\Gamma[\Delta]] \vdash C}{\Sigma[\Delta \cdot (i \cdot \Gamma[t_i])] \vdash C} \text{QR}\downarrow$$

Let A be the label at the root of $\Gamma[\Delta]$, and let B be the label at the root of Δ. Then the newly created structure $i \cdot \Gamma[t_i]$ has type $B \to A$ by virtue of the typing rule T2, and so the larger newly created structure $\Delta \cdot (i \cdot \Gamma[t_i])$ has type A, by virtue of the typing rule T1. We label the newly created structures accordingly. Since the larger newly created structure and $\Gamma[\Delta]$ both have type A, the recursive assumption guarantees that the newly-extended labeling has type C.

When the labeling on Σ_n is copied onto σ_n, the labeling satisfies the typing rules, and justifies the claim that σ_n has type A.

A.5 Hint for the problem posed in (13)

One of the two semantically distinct analyses requires at least one instance of quantifier raising; the other requires at least two. The paraphrases of the interpretations are *They all gave them the same excuse* and *They gave them all the same excuse*. See Bumford & Barker 2013 for a discussion of how the type given to *same* accounts for the ambiguity in the presence of Quantifier Raising.

References

Barendregt, Henk. 1984. *The lambda calculus*. Amsterdam: Elsevier. https://doi.org/10.1016/c2009-0-14341-6.

Barker, Chris. 2007a. Direct compositionality on demand. In Chris Barker & Pauline Jacobson (eds.), *Direct compositionality*, 102–131. Oxford: Oxford University Press.

Barker, Chris. 2007b. Parasitic scope. *Linguistics and Philosophy* 30(4). 407–444. https://doi.org/10.1007/s10988-007-9021-y.

Barker, Chris. 2019. NL_λ as the logic of scope and movement. *Journal of Logic, Language and Information* 28. 217–237. https://doi.org/10.1007/s10849-019-09288-1.

Barker, Chris & Pauline Jacobson. 2007. Introduction: Direct compositionality. In Chris Barker & Pauline Jacobson (eds.), *Direct compositionality*, 1–19. Oxford: Oxford University Press.

Barker, Chris & Chung-chieh Shan. 2014. *Continuations and natural language*. Oxford: Oxford University Press. https://doi.org/10.1093/acprof:oso/9780199575015.001.0001.

Bumford, Dylan. 2017. Split-scope definites: Relative superlatives and haddock descriptions. *Linguistics and Philosophy* 40(6). 549–593. https://doi.org/10.1007/s10988-017-9210-2.

Bumford, Dylan & Chris Barker. 2013. Association with distributivity and the problem of multiple antecedents for singular different. *Linguistics and Philosophy* 36(5). 355–369. https://doi.org/10.1007/s10988-013-9139-z.

Charlow, Simon. 2020. The scope of alternatives: Indefiniteness and islands. *Linguistics and Philosophy* 43. 427–472. https://doi.org/10.1007/s10988-019-09278-3.

Charlow, Simon. to appear. Postsuppositions and semantic theory. *Journal of Semantics*.

Cresti, Diana. 1995. Extraction and reconstruction. *Natural Language Semantics* 3. 79–122. https://doi.org/10.1007/BF01252885.

Gentzen, Gerhard. 1935. Untersuchungen über das logische schließen. I. *Mathematische Zeitschrift* 39. Reprinted in Gentzen 1969, 176–210. https://doi.org/10.1007/bf01201353.

Gentzen, Gerhard. 1969. *The collected papers of Gerhard Gentzen*. Trans. by Miklos Erdelyi Szabo (Studies in Logic and the Foundations of Mathematics 55). https://doi.org/10.1016/s0049-237x(08)x7039-9.

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

Hendriks, Herman. 1993. *Studied flexibility: Categories and types in syntax and semantics*. Amsterdam: Institute for Logic, Language & Computation (ILLC) dissertation. https://hdl.handle.net/11245/1.392531.

Jacobs, Joachim. 1980. Lexical decomposition in montague grammar. *Theoretical Linguistics* 7. 121–136. https://doi.org/10.1515/thli.1980.7.1-3.121.

Jacobson, Pauline. 2012. Direct compositionality. In Wolfram Hinzen, Edouard Machery & Markus Werning (eds.), *The Oxford handbook of compositionality*. Oxford: Oxford University Press. https://doi.org/10.1093/oxfordhb/9780199541072.013.0005.

Jäger, Gerhard. 2005. *Anaphora and type logical grammar*. Dordrecht: Springer. https://doi.org/10.1007/1-4020-3905-0.

Kennedy, Christopher & Jason Stanley. 2009. On 'average'. *Mind* 118(471). 583–646. https://doi.org/10.1093/mind/fzp094.

Kubota, Yusuke & Robert Levine. 2015. Against ellipsis: Arguments for the direct licensing of 'noncanonical' coordinations. *Linguistics and Philosophy* 38(6). 521–576. https://doi.org/10.1007/s10988-015-9179-7.38.

Kuhlmann, Marco, Giorgio Satta & Peter Jonsson. 2018. On the complexity of CCG parsing. *Computational Linguistics* 44(3). 447–482. https://doi.org/10.1162/coli_a_00324.

Lambek, Joachim. 1958. The mathematics of sentence structure. *The American Mathematical Monthly* 60(3). 154–170. https://doi.org/10.1075/llsee.25.12lam.

Lechner, Winfried. 2017. Phrasal comparatives and parasitic scope. *Wiener Linguistische Gazette* 82. 181–191. https://wlg.univie.ac.at/fileadmin/user_upload/p_wlg/822017/Lechner_wlg.pdf.

May, Robert. 1977. *The grammar of quantification*. Cambridge, MA: MIT dissertation. http://dspace.mit.edu/handle/1721.1/16287.

Moortgat, Michael. 1997. Categorial type logics. In Johan van Benthem & Alice ter Meulen (eds.), *The handbook of logic and language*, 93–177. Amsterdam: Elsevier. https://doi.org/10.1016/b978-0-444-53726-3.00002-5.

Moot, Richard. 2020. *Proof-theoretic aspects of NL_λ*. arXiv preprint. https://doi.org/10.48550/arXiv.2010.12223.

Partee, Barbara. 1987. Noun phrase interpretation and type-shifting principles. In Jeroen Groenendijk, Dick de Jongh & Martin Stokhof (eds.), *Studies in Discourse Representation Theory and the theory of general quantifiers*, 115–143. Berlin: de Gruyter. https://doi.org/10.1515/9783112420027-006.

Partee, Barbara & Mats Rooth. 1983. Generalized conjunction and type ambiguity. In Rainer Bäuerle, Christoph Schwarze & Arnim von Stechow (eds.), *Meaning, use, and interpretation of language*, 361–383. Berlin: de Gruyter. https://doi.org/10.1515/9783110852820.361.

Restall, Greg. 2000. *An introduction to substructural logics*. Routledge. https://doi.org/10.4324/9780203252642.

Weir, Alan. 2015. A robust non-transitive logic. *Topoi* 34(1). 99–107. https://doi.org/10.1007/s11245-013-9176-9.

Winter, Yoad. 2007. Type shifting with semantic features: A unified perspective. In Chris Barker & Pauline Jacobson (eds.), *Direct compositionality*, 164–187. Oxford: Oxford University Press.

Chris Barker
10 Washington Place
New York, NY 10003 USA
chris.barker@nyu.edu