The logic of Quantifier Raising

Chris Barker, *New York University Linguistics*

DRAFT of August 13, 2020

Abstract: Displaced scope is a hallmark of natural language, and Quantifier Raising (QR) has long been the standard tool for analyzing displaced scope. Yet as far as I know, there has never been a study of the formal properties of Quantifier Raising. 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 there is at least one such derivation, is the number of semantically different analyses 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 answer these questions by defining QRT (Quantifier Raising with Types), a substructural logic that is a faithful model of QR in the following sense: 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. These results put Quantifier Raising on a reassuringly firm formal footing.

Acknowledgements: Thanks to Simon Charlow, Sandra Chung, Josh Dever, Berit Gehrke, Daniel Lassiter, Richard Moot, my anonymous referees, and audiences at LENLS 11, ESSLLI 2015, and Stanford. 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.

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 analyses? (Yes.) If there is at least one such an analysis, is the number of semantically distinct analyses finite? (Yes.) Given that 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? There is a theoretical answer and a practical answer. Theoretically, if QR were not decidable, any theory of 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. In other words, 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. If QR were not decidable or did not have the finite readings property, it would literally be impossible to ever write a computer program that would list all coherent derivations for an arbitrary example.

As any experienced semanticist knows, such 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 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.

The approach to studying QR here 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 Quantifier Raising

Heim and 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 generations of linguists and philosophers. Despite (or perhaps because of?) the centrality of the topic, they do not give a definition of Quantifier Raising. 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 constraint motivated from a formal point of view other than type coherence (defined below).

2.1 Quantifier Raising and QR derivations

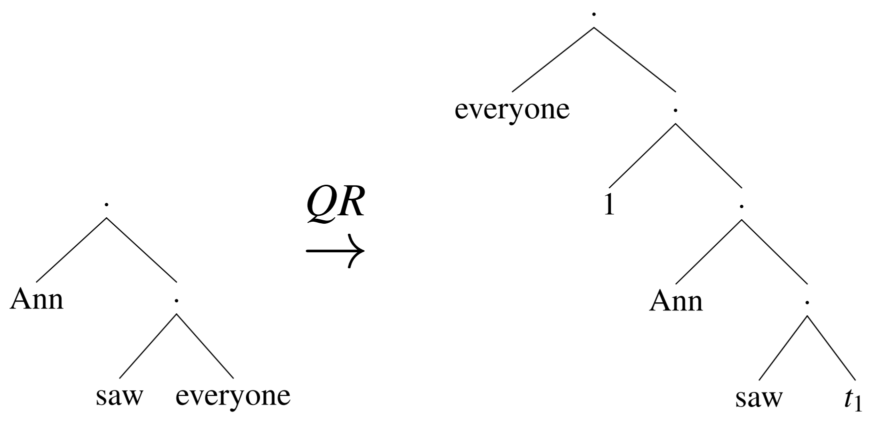
Heim and Kratzer 1998:210 quote May's original 1977:18 proposed rule of QR verbatim:

(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 and Kratzer's 1998:186 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 (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 begin 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.

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) $L := W \mid LL \mid t_i \mid iL$ **(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 $\sigma = [...[...\delta...]_\gamma...]$ be a logical form that contains a logical form γ that contains a logical form δ. Then Quantifier Raising applied to σ produces $[...[\delta[i[...t_i...]_\gamma]]...]$ 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 and 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 standard practice.

I'll keep types as simple as possible, just as in Heim and Kratzer 1998:28 among many others. As usual, there will be a basic type $\mathtt{e}$ for individuals and a basic type $\mathtt{t}$ for truth values, as well as functional types $A \to B$, where A and B are arbitrary types. Following common practice, types can be abbreviated: $A \to B$ 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 $\mathtt{eet}$ is the type $\mathtt{e} \to (\mathtt{e} \to \mathtt{t})$, 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) $T := \mathtt{e} \mid \mathtt{t} \mid T \to T$ **(Types)**

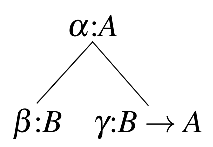
If a logical form α is a member of a type A, I'll say that α *has type* A, and write $\alpha : A$.

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:

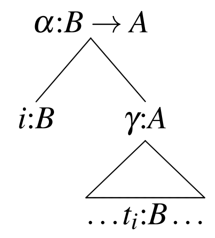
(8) **Typing rules for logical forms:**

T0. 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 $B \to A$. 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 $B \to A$ 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.

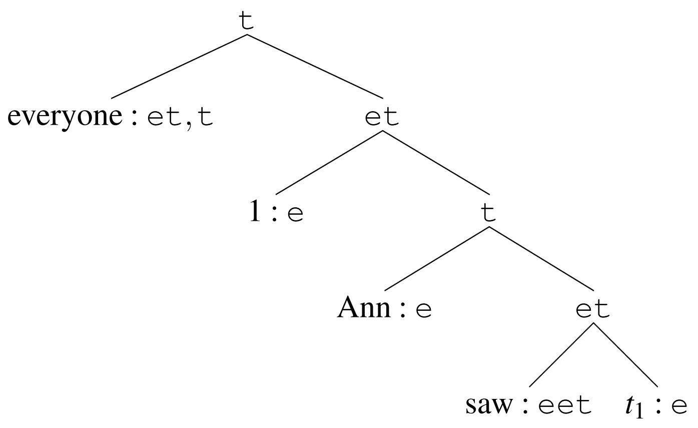


With these preliminaries, we can now say what it means for a QR derivation to be type-coherent:

(9) **Type coherence**: 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. A QR derivation is coherent with respect to types if 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 and 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 2007a), *average* (Kennedy and Stanley 2009), non-canonical coordination (Kubota and 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 $\mathtt{t}$), 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 $\forall y \exists x.\mathbf{saw}\, y\, x$ and $\exists x \forall y.\mathbf{saw}\, y\, x$ are semantically distinct, but $\mathbf{saw}\,\mathbf{ann}$ and $(\forall P.P\,\mathbf{ann})\,\mathbf{saw}$ 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 1934.

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 and Kratzer 1998:43). 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 turnstyle ('$\vdash$') followed by a formula. For our purposes, sequents can be thought of as typing judgments. For instance, the sequent $\mathtt{e}\cdot\mathtt{e}\rightarrow\mathtt{t}\vdash\mathtt{t}$ can be read as "a structure consisting of a type $\mathtt{e}$ and a type $\mathtt{e}\rightarrow\mathtt{t}$ has type $\mathtt{t}$."

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

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

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

$$\Gamma[\Delta]=\Delta\cdot(i\cdot\Gamma[t_i]) \qquad \text{(QR)}$$

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 ('$\rightarrow$'). The left rule ('$\rightarrow L$') 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 $B\rightarrow A$. 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 and Kratzer 1998:96 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.

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 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:

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

Following Barker and Shan 2014 chapter 17 and Barker 2019, I will call $\text{QR}_{\downarrow}$ 'reduction' and $\text{QR}_{\uparrow}$ '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 $\mathtt{t}$:

(17)

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\mathtt{e}\vdash\mathtt{e}}\text{Ax} \qquad \dfrac{\dfrac{}{\mathtt{e}\vdash\mathtt{e}}\text{Ax} \qquad \dfrac{}{\mathtt{t}\vdash\mathtt{t}}\text{Ax}}{\mathtt{e}\cdot\mathtt{et}\vdash\mathtt{t}}\to L}{\mathtt{e}\cdot(\mathtt{eet}\cdot\mathtt{e})\vdash\mathtt{t}}\to L}{1\cdot(\mathtt{e}\cdot(\mathtt{eet}\cdot t_1))\vdash\mathtt{et}}\to R_i \qquad \dfrac{}{\mathtt{t}\vdash\mathtt{t}}\text{Ax}}{\mathtt{et,t}\cdot(1\cdot(\mathtt{e}\cdot(\mathtt{eet}\cdot t_1)))\vdash\mathtt{t}}\to L}{\mathtt{e}\cdot(\mathtt{eet}\cdot\mathtt{et,t})\vdash\mathtt{t}}\text{QR}_{\uparrow}$$

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 $\text{QR}_{\uparrow}$ 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), $\rightarrow L$ corresponds to function application, $\rightarrow R_i$ 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 and 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., $((\lambda xx)(\mathrm{e}))$ and e).

The correspondence between QR and QRT has two parts. The first part 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 part 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 $\mathrm{QR}_\downarrow$; and $\mathrm{QR}_\uparrow$, 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.

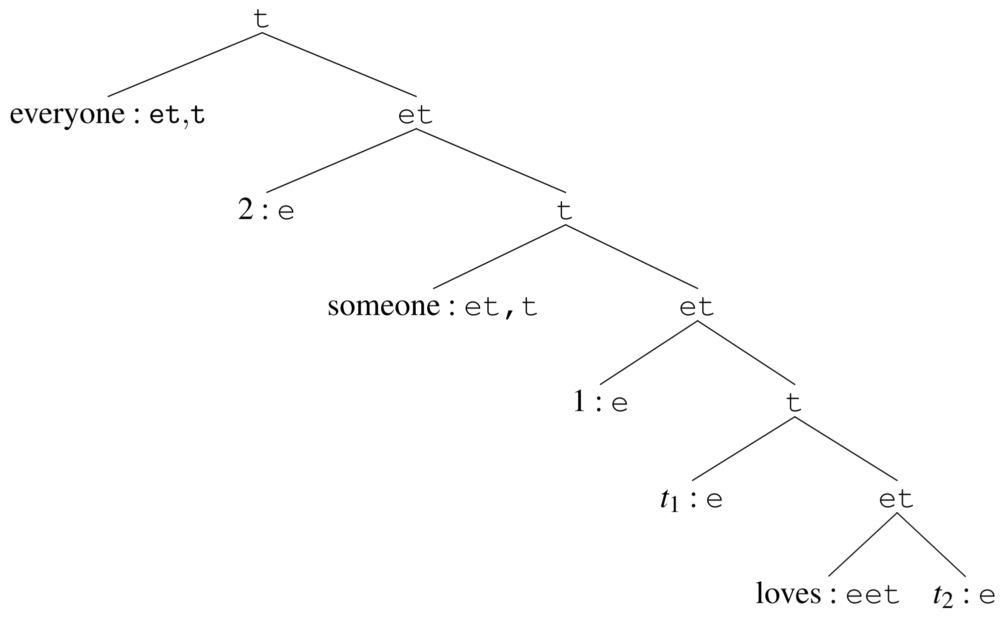
b. QR derivation = σ, σ_1, σ_2, where

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:

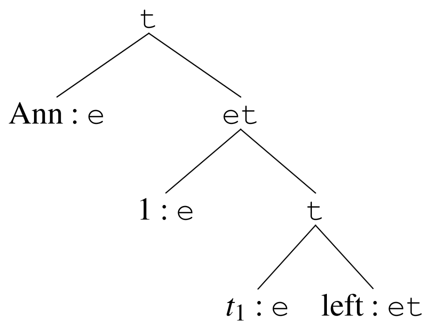
(19)

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

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$_\downarrow$, which corresponds to two nodes. Predicate Abstraction in a QR derivation sometimes corresponds to $\to R_i$, and sometimes to a reduction instance of the QR structural rule (QR$_\downarrow$), 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.



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

QR of names is explicitly allowed by Heim and Kratzer 1998:210. 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 $(\lambda x_1.\mathbf{left}\,x_1)\,\mathbf{ann}$, 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 QRT proofs will never include such a semantically useless 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 track 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: finding a semantically coherent QR derivation, if one exists, is decidable; and for any particular logical form, the number of such derivations that are semantically distinct is finite. Here's the synopsis: assuming 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 1934 Hauptsatz. In any logic, proving a conclusion depends on proving a set of premises from which the conclusion follows. Because the logic is in sequent presentation, 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, the QR$_\downarrow$ inference also has the subformula property, and its premise contains strictly fewer structural connectives than the conclusion.

Thus the only inference that could potentially cause trouble is QR$_\uparrow$. 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. 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$_\uparrow$ are limited as well.

The second case is when the index eliminated by QR$_\uparrow$ was introduced by an instance of $\mathrm{QR}_{\downarrow}$ 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 $\mathrm{QR}_{\downarrow}$ and $\mathrm{QR}_{\uparrow}$, 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 $\mathrm{QR}_{\uparrow}$ in the second case, only the first case remains.

Is there a 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 abandonned the requirement that the final logical 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. As explained above, in order for QR to have a non-null semantic effect, it must interact with one of the logical rules (the only option here 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 an especially relevant example, Kuhlmann et al. 2018 discusses the parsing complexity of Combinatory Categorial Grammar, a formalism with some important similarities to Type Logical Grammar. 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 QRST: Quantifier Raising with Shifty Types

This paper is about Quantifier Raising, and QRT delivers exactly standard Quantifier Raising, no more, no less. Nevertheless, it will be worthwhile to consider a variant logic QRST (Quantifier Raising with Shifty Types). QRST is arguably more natural from a purely logical point of view, in a sense that will become clear shortly. In addition to validating all QRT proofs, QRST also validates certain well-known type shifting principles, including a generalization of Partee's 1987 LIFT operator. Despite enlarging the proof search space, QRST remains decidable with finite readings. This means that we can make unrestricted lifting available without compromising decidabilty or finite readings for Quantifier Raising.

QRST also validates Argument Raising and Value Raising, the two core type shifting principles underlying Hendriks' 1993 Flexible Montague Grammar. Flexible Montague Grammar, of course, is a paradigm example of an in-situ (i.e., non-movement) framework for scope-taking, one that in addition is directly compositional. Comparing QRT with QRST shows that it is not the presence of Quantifier Raising that forces movement in a QR theory, or that renders it non directly compositional (since QRT and QRST both have the same rule of Quantifier Raising), but rather the nature of the rest of the inferential system.

6.1 QRST and generalized lifting

Type shifters are silent operators that shift expressions having one type into expressions having another type. For instance, Partee's 1987 LIFT operator shifts an individual-denoting name of type `e` into a generalized quantifier of type `et,t`, with an appropriate parallel shift in meaning (from, say, **ann** to $\lambda P.P\,\textbf{ann}$). See Partee and Rooth 1983, Partee 1987, and many later works for empirical motivation for lifting.

Lifting is especially relevant in the current context because lifting can force Quantifier Raising. If a name like *Ann* has type `e`, it can appear in a logical form like [Bill (saw Ann)] without needing to undergo QR. But if *Ann* underogoes lifting, then a generalized quantifier sits in direct object position, and QR is forced in order to arrive at a semantically coherent derivation.

If we allow lifting, how will that affect decidability? Do we need to limit lifting in some way, such as forbidding lifting any particular expression more than once? Do we need to allowing it only when it is forced for reasons of type mismatch, as discussed in Winter 2007, among others?

We can settle these questions by making an adjustment to one inference rule of QRT:

(21)

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

The original rule given above for QRT is on the left. The modified rule is the standard right rule for $\rightarrow$ in Gentzen's 1934 system LJ (a sequent presentation of intuitionistic logic), and the same rule used throughout the Type Logical literature. Note that every valid proof in QRT still corresponds to a valid proof in QRST, thanks to the following equivalence:

(22)

$$\frac{\Gamma[B] \vdash A}{i \cdot \Gamma[t_i] \vdash B \rightarrow A} \rightarrow R_i \qquad \equiv \qquad \cfrac{\cfrac{\Gamma[B] \vdash A}{B \cdot (i \cdot \Gamma[t_i]) \vdash A}\,\mathrm{QR}_\downarrow}{i \cdot \Gamma[t_i] \vdash B \rightarrow A} \rightarrow R$$

This equivalence reveals that the QRT rule is a combination of the standard rule with an instance of $\mathrm{QR}_\downarrow$. Because of this conflation, the QRT version limits the availability of $\rightarrow$-right inferences to situations in which an abstraction created by $\mathrm{QR}_\uparrow$ is present. This restriction is precisely what limits QRT to Quantifier Raising with nothing added.

If we restore the standard right rule, new theorems become available, such as this one:

$$\cfrac{\cfrac{\mathtt{e} \vdash \mathtt{e} \qquad \mathtt{t} \vdash \mathtt{t}}{\mathtt{et} \cdot \mathtt{e} \vdash \mathtt{t}} \rightarrow L}{\mathtt{e} \vdash \mathtt{et,t}} \rightarrow R$$

The conclusion sequent is an instance of Partee's 1987 LIFT operator. The conclusion sequent says that any expression in type `e` also has type `et,t`. That is, any expression with the semantic type of an individual (type `e`), such as a proper name, can function semantically in the role of a generalized quantifier (type `et,t`). The Curry-Howard labeling for this proof gives exactly the generalized quantifier semantics mentioned above. 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`.

Making this change does not interfere with either decidability or the finite readings property. One way to see this is to note that the modified logic 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. One simple way to do this would be to add a new typing rule for QR derivations that says that if α is a logical form that has type A, α also has type $(A \to B) \to B$ for any type B, with semantic interpretation $\lambda \kappa.\kappa a$, where a is the semantic interpretation of α when it has type A.

6.2 In-situ scope and direct compositionality

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

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`):

(23)

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\vdots\\ \texttt{et,t}\cdot(1\cdot(\texttt{e}\cdot(\texttt{eet}\cdot t_1)))\vdash\texttt{t}\end{array}}{\texttt{e}\cdot(\texttt{eet}\cdot\texttt{et,t})\vdash\texttt{t}}\,\text{QR}_{\uparrow}}{\texttt{eet}\cdot\texttt{et,t}\vdash\texttt{et}}\,\to R}{\texttt{eet}\vdash\texttt{(et,t)et}}\,\to R$$

Reading from the bottom up, the proof pushes each of the argument types of the conclusion result across the turnstyle, 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 we've seen in derivations given earlier. 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`:

(24)

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \texttt{et,t}\cdot(1\cdot(\texttt{e}\cdot(\texttt{eet}\cdot t_1)))\vdash\texttt{t}\end{array}}{\texttt{e}\cdot(\texttt{eet}\cdot\texttt{et,t})\vdash\texttt{t}}\,\text{QR}_{\uparrow}}{\texttt{eet}\cdot\texttt{et,t}\vdash\texttt{et}}\,\to R$$

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, on which every constituent must always be given a semantic interpretation, since there are plenty of derivations on which the verb phrase does not receive a semantic value, as we have seen. Nevertheless, we can compute the missing semantic interpretations whenever desired. Barker 2007b 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.

What should we make of this situation? There is no difference between QRT and QRST with respect to the structural rule that expresses scope-taking (the QR rule). In this case, the difference between a pure movement theory of scope such as bare QR and an in-situ theory of scope such as Flexible Montague Grammar 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. With the QRT version of $\rightarrow R_i$, scope taking always requires movement and is not directly compositional; with the more general QRST version of the rule, we have in-situ analyses and direct compositionality on demand. Thus the type shifting principles of Flexible Montague Grammar follow from QR combined with the usual rule of hypothetical reasoning ($\rightarrow R$).

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.

A note of caution: it is not the case that Flexible Montague Grammar and QR derivations provide the same expressive power. As noted by Barker and Shan 2014:70, Flexible Montague Grammar does not handle parasitic scope, among other limitations. A fully general argument that QRST provides direct compositionality on demand requires an interpolation theorem similar to that in Barker 2019, which in turn requires adding a fusion operator to the logical formulas. This is straightforward, but I will not provide those details here.

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 $QR_{\uparrow}$ to create unbound traces will never lead to a complete and valid proof.

7.2 Complex traces

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

7.3 Quantifier Raising is syntactic

The bi-directional structural equation in QRT that characterizes Quantifier Raising is a structural inference rule. This means that an in-situ quantifier in its surface position and the corresponding logical form created by QR are not just semantically related, they are *fully syntactically equivalent*. 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 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 2007a. As shown in Barker and 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.

9 References

Alonso-Ovalle, Luis. 2006. *Disjunction in Alternative Semantics*. UMass Amherst PhD dissertation.

Barker, Chris. 2007a. Parasitic Scope. *Linguistics and Philosophy* **30.4**: 407–444.

Barker, Chris. 2007b. Direct Compositionality on Demand. In Chris Barker and Pauline Jacobson (eds). *Direct Compositionality*. Oxford University Press. 102–131.

Barker, Chris. 2019. NL_λ as the logic of scope and movement. *Journal of Logic, Language and Information*. 1–21.

Bumford, Dylan and Chris Barker. 2013. Association with distributivity and the problem of multiple antecedents for singular different. *Linguistics and Philosophy* **36.5**: 355–369.

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

Barker, Chris and Pauline Jacobson. Introduction: Direct Compositionality. In Chris Barker and Pauline Jacobson (eds), *Direct Compositionality*. Oxford. 1–19.

Bumford, Dylan. 2017. Split-scope definites: Relative superlatives and Haddock descriptions. *Linguistics and Philosophy* **40.6**: 549–593k.

Charlow, Simon. 2020. The scope of alternatives: indefiniteness and islands *Linguistics and Philosophy* **43**: 427–472.

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

Cresti, Diana. 1995. Extraction and reconstruction. *Natural Language Semantics* **3**: 79–122.

Dayal, Veneeta. 2013. The syntax of scope and quantification. In *The Cambridge Handbook of Generative Syntax*. 827–859.

Gentzen, Gerhard. 1934. Untersuchungen über das logische Schließen. I. *Mathematische Zeitschrift* **39.2**: 176–210. Translated by M.E. Szabo. 1969. *The Collected Papers of Gerhard Gentzen* Studies in Logic and the Foundations of Mathematics, Volume 55: 68–131.

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

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

Jacobs, Joachim. 1980. Lexical decomposition in Montague Grammar. *Theoretical Linguistics* **7**:121–136.

Jäger, Gehard. 2005. *Anaphora and Type Logical Grammar.* Springer.

Kennedy, Christopher, and Jason Stanley. 2009. On *average*. *Mind* **118.471**: 583–646.

Kratzer, Angelika. 1998. Scope or Pseudoscope? Are There Wide-Scope Indefinites? In Susan Rothstein (ed). *Events and Grammar*. Kluwer, Dordrecht. 163–196.

Kratzer, Angelika and Junko Shimoyama. 2002. Indeterminate pronouns: the view from Japanese. In Hituzi Syobo (ed). *Proceedings of the 3rd Tokyo Conference on Psycholinguistics*: 1–24.

Kubota, Yusuke, and Robert Levine. 2015. Against ellipsis: arguments for the direct licensing of 'noncanonical' coordinations. *Linguistics and Philosophy*. **38.6**: 521–576.

Kuhlmann, M., Satta, G., and Jonsson, P. 2018. On the complexity of CCG parsing. *Computational Linguistics* **44.3**: 447–482.

Lambek, Joachim. 1958. The mathematics of sentence structure. *The American Mathematical Monthly* **60.3**: 154–170.

Lechner, Winfried. 2017. Phrasal comparatives and Parasitic Scope. *Wiener Linguistische Gazette* 82:181–191.

May, Robert. 1977. *The Grammar of Quantification.* MIT PhD dissertation. Reprinted by Garland, 1991.

Moortgat, Michael. 1997. Categorial Type Logics. In *The Handbook of Logic and Language*, J. van Benthem and Alice G. B. ter Meulen (eds).

Partee, Barbara. 1987. Noun Phrase Interpretation and Type-Shifting Principles. In Jeroen Groenendijk, Dick de Jongh, and Martin Stokhof (eds). *Studies in Discourse Representation Theory and the Theory of Generalized Quantifiers*. Foris, Dordrecht. 115–143.

Partee, Barbara and Mats Rooth. 1983. Generalized conjunction and type ambiguity. In Bauerle, R., Schwarze, C., and von Stechow, A., editors, *Meaning, Use and Interpretation of Language*. De Gruyter, Berlin.

Reinhart, Tanya. 1997. Quantifier Scope: How Labor is Divided Between QR and Choice Functions. *Linguistics and Philosophy*. **20.4**: 335–397.

Restall, Greg. 2000. *An Introduction to Substructural Logics*. Routledge.

Szabolcsi, Anna. 2010. *Quantification*. Cambridge University Press.

Weir, A. 2015. A Robust Non-transitive Logic. *Topoi* **34.1**: 99–107.

White, M., Charlow, S., Needle, J. and Bumford, D., 2017. Parsing with Dynamic Continuized CCG. *In Proceedings of the 13th International Workshop on Tree Adjoining Grammars and Related Formalisms*. 71–83.

Winter, Yoad. 2007. Type shifting with semantic features: A unified perspective. In C. Barker and P. Jacobson (eds). *Direct compositionality*: 164–187.

10 Appendix

This Appendix proves cut elimination for QRT, and gives details of the mappings from QR derivations to QRT and back again.

10.1 Cut elimination

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

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

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, 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.

(26) **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 1934, Restall 2000, Jäger 2005:41). 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.

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. This means in particular that neither QR↑ nor QR↓ impedes cut elimination.

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 wrinkle in the QRT case compared to the usual proof is when the cut formula is the active formula in both of the premises (a *principal cut*).

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

The cut formula is $B \to A$, which is also the active formula for both the $\to R_i$ inference and the $\to L$ inference.

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

The original cut has been transformed into a pair of smaller cuts using the same initial premises and arriving at the same final conclusion. Here, 'smaller' refers to the total number of base types and logical connectives in the premises; see, e.g., Jäger 2005:43 for a precise definition. Because this transformation eliminates the $\to R_i$ instance, and because the $\to R_i$ rule in effect incorporates an instance of reduction, it is necessary to add a compensating reduction inference in the replacement proof fragment.

10.2 Every QR derivation has an equivalent QRT proof

Assume we have a semantically coherent QR derivation $\sigma, \sigma_1, ..., \sigma_n$ 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 two parameters begin with internal structures that are isomoprhic 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:

1. 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.

2. 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 $\Pi = \Gamma$.

3. 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 $\Gamma[B] \vdash A$.

Thus 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, so it is necessary to eliminate cuts according to the method sketched above in order to arrive at a cut-free QRT proof. This is why the examples in the main text are cut-free. If the cut rule is added to QRT, the correspondence between QR derivations and QRT proofs becomes even closer: the corresponding proof is not only beta-equivalent, it is fully semantically equivalent.

10.3 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.

(27) 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:

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

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.

10.4 Every QRT proof has an equivalent QR derivation

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

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 every QR$_\uparrow$ inference follows every logical inference, and in which every QR$_\downarrow$ inference has been eliminated (along with its coindexed QR$_\uparrow$ inference). After these adjustments, all QR$_\uparrow$ inferences will be gathered together as the final n inferences of p'. Then the QR$_\uparrow$ inferences in p' induce a QR derivation $\sigma, \sigma_1, ..., \sigma_n$ 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 $\Sigma, \Sigma_1, ..., \Sigma_n$.

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 D. Because σ_n and Σ_n are isomorphic (up to sibling order), it will be convenient to associate labels with the structure of Σ, relying on the isomorphism to map the labels onto the corresponding nodes of σ_n.

The argument involves induction on the structure of p'. The base case of the inductive argument is an axiom inference of the form $A \vdash A$. Any logical form labeled with the left hand side (trivially) has the result type.

For the inductive case of the argument, consider a non-axiom inference, and assume that we have a labeling for each premise such that the labeling satisfies the typing rules, and the label of the root of the structure matches the type on the result side of the premise conclusion. Since the subproof of p' whose final conclusion sequent is $\Sigma_n \vdash A$ does not contain any expansion inferences (by construction), the inductive part of the argument has three subcases: $\rightarrow L$, $\rightarrow R_i$, and QR$_\downarrow$. We reason as follows:

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

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 structure. 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.

$\mathrm{QR}_\downarrow$:

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

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 D.

Here is a hint for the problem posed in (13): one of the 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 and Barker 2013 for a discussion of how the type given to *same* accounts for the ambiguity in the presence of Quantifier Raising.