SOME NOTES ON LOGIC AND QUANTIFIER RAISING

MARCUS KRACHT

1. Predicate Logic

This note clarifies some of the concepts used in the lecture. I continue to use typewriter fonts for actual symbols; this saves me the quotes. In addition to typewriter font also boldface serves that purpose. This will allow me to distinguish a token word like `green` from its formal translation, which is **green**. Otherwise it is not possible to see that the translation actually does something. Arbitrary strings are written using different fonts. Also, I prefer to write '**every** `x`' in place of '$\mathbf{every}_{\mathtt{x}}$' for the fact that the letter 'x' has an actual occurrence as a variable rather than being part of the string that constitutes the quantifier. The syntax of formulae is crucial, so I do not adopt a convention to drop brackets. This means that I do not write $\varphi\to\psi$ but always $(\varphi\to\psi)$, even when the brackets seem superfluous. They constitute parts of the string that we have to write down. No convention on dropping brackets is in place during this note! However, there will be a transition in languages betwen Sections 3 and 4.

2. Truth and Evaluation

The language of predicate logic consists of the following symbols:

① **Auxiliary symbols**: `(` and `)`.
② **Variables**: typically `x`, `y`; often other conventions are employed, such as $\mathtt{x}_i$, where i is a number. This helps to prevent running out of symbols. Computer languages actually allow to use anything as a variable as long as it has no predefined meaning.
③ **Connectives**: $\&$, $\neg$, $\vee$, $\to$,
④ **Quantifiers**: $\forall$ and $\exists$ (and the generalised quantifiers).
⑤ **Constants**: (any set we please)
⑥ **Unary predicate letters**: (any set we please)
⑦ **Binary predicate letters**: (any set we please)

This set above is the **alphabet**, called A. The language is defined as follows.

Definition 1 (Formulae)**.** *The set of* ***formulae over*** A *is defined as follows.*

① *If* F *is a unary predicate letter and* x *a variable or a constant, then* $F^\frown x$ *is a formula.*
② *If* R *is a binary predicate letter and* x *and* y *variables or constants, then* $x^\frown R^\frown y$ *is a formula.*
③ *If* $\vec{x}$ *and* $\vec{y}$ *are formulae, then so are* $\neg^\frown\vec{x}$, $(\vec{x}\&\vec{y})$, $(\vec{x}\wedge\vec{y})$, $(\vec{x}\vee\vec{y})$, *and* $(\vec{x}\to\vec{y})$.
④ *If* $\vec{x}$ *is a formula and* y *a variable, then* $\exists y\vec{x}$ *and* $\forall y\vec{x}$ *are formulae.*

Notice that there are no brackets in combination with quantifiers and negation. This is not an oversight. This is not very user friendly: I have used $^\frown$ where we ordinarily write nothing. Erase it if it disturbs you. Also, we often do not write $\vec{x}$ for formulae but φ, ψ and so on. This is for the eye only. Writing $\vec{x}$ I remind you of the fact that they are strings. Assuming that the symbols of A are letters far from practical. In actual practice we use all kinds of strings as primitive symbols. In this case, what I write as simple concatenation will have to be replaced by concatenation with a blank inserted. So detail will be exposed in Section 4. (If you have typed on a keyboard of a computers or a typewriter you will actually understand that the blank is the space bar and not simply nothing.)

Notice that there is not a single language but an infinite set of them. This is because we may choose the constants and predicate letters. The theory does not specify what they are. In predicate logic it is customary to use *structures* to interpret formulae. A structure contains (a) a **domain**, and (b) a function that assigns an interpretation to every constant and every function. We write as follows: a structure is a pair $\mathfrak{M} = \langle D, I\rangle$, where D is a set and I a function such that

(1) for every unary predicate letter F: $I(F) \subseteq D$.
(2) for every binary predicate letter R: $I(F) \subseteq D \times D$.

The phrase 'what F stands for' can be given a precise definition. First, it is *relative to a structure*; second, given a structure $\mathfrak{M} = \langle D, I\rangle$, 'what F stands for' is $I(F)$. Thus, if F is a constant (say, `Maria`), then I(`Maria`) is some specific individual. If F is a unary predicate letter, say `runs` (watch my use of 'letter'!), then it stands for I(`runs`), which is a set of individuals; and finally, if it is a binary predicate letter, say `sees`, then it stands for a relation between individuals, here I(`sees`).

Finally, an **assignment** is a function σ from the set of variables to D. A **model** consists of a structure and an assignment. We write $\langle\mathfrak{M}, \sigma\rangle$ for the model. Given a formula φ and a model $\langle\mathfrak{M}, \sigma\rangle$, we write

$\langle\mathfrak{M},\sigma\rangle \vDash \varphi$ to say that φ is true in the model; or that it is true$_\sigma$ in $\mathfrak{M}$. Relative to a model we can say the following:

Definition 2 (Atomic Formulae). *Let F be a unary predicate letter, R a binary predicate letter, x, y variables, c and d constants, and $\langle\mathfrak{M},\sigma\rangle$ a model.*

(1) *Fx is true$_\sigma$ in $\mathfrak{M}$ iff $\sigma(x) \in I(F)$.*
(2) *Fc is true$_\sigma$ in $\mathfrak{M}$ iff $I(c) \in I(F)$.*
(3) *xRy is true$_\sigma$ in $\mathfrak{M}$ iff $\langle\sigma(x),\sigma(y)\rangle \in I(R)$.*
(4) *xRd is true$_\sigma$ in $\mathfrak{M}$ iff $\langle\sigma(x),I(d)\rangle \in I(R)$.*
(5) *cRy is true$_\sigma$ in $\mathfrak{M}$ iff $\langle I(c),\sigma(y)\rangle \in I(R)$.*
(6) *cRd is true$_\sigma$ in $\mathfrak{M}$ iff $\langle I(c),I(d)\rangle \in I(R)$.*

We say that φ is false$_\sigma$ in $\mathfrak{M}$ iff it is not true$_\sigma$ in $\mathfrak{M}$.

Definition 3 (Connectives). *Let φ, ψ be formulae and $\langle\mathfrak{M},\sigma\rangle$ a model.*

(1) *$\neg\varphi$ is true$_\sigma$ in $\mathfrak{M}$ iff φ is not true$_\sigma$ in $\mathfrak{M}$;*
(2) *$(\varphi\&\psi)$ is true$_\sigma$ in $\mathfrak{M}$ iff both φ and ψ are true$_\sigma$ in $\mathfrak{M}$;*
(3) *$(\varphi\vee\psi)$ is true$_\sigma$ in $\mathfrak{M}$ iff either φ or ψ is true$_\sigma$ in $\mathfrak{M}$;*
(4) *$(\varphi\rightarrow\psi)$ is true$_\sigma$ in $\mathfrak{M}$ iff φ is not true$_\sigma$ in $\mathfrak{M}$ or ψ is true$_\sigma$ in $\mathfrak{M}$.*

Definition 4 (Quantifiers). *Let φ be a formula, x a variable and $\langle\mathfrak{M},\sigma\rangle$ a model.*

(1) *$\exists x\varphi$ is true$_\sigma$ in $\mathfrak{M}$ iff there is an $o \in D$ such that φ is true$_{\sigma[x/o]}$ in $\mathfrak{M}$;*
(2) *$\forall x\varphi$ is true$_\sigma$ in $\mathfrak{M}$ iff for every an $o \in D$: φ is true$_{\sigma[x/o]}$ in $\mathfrak{M}$.*

It is possible that if $\mathfrak{M}'$ was another structure, or σ' another assignment, φ is true$_\sigma$ in $\mathfrak{M}$ and not true$_{\sigma'}$ in $\mathfrak{M}'$. What is absolutely crucial for these definitions that they do not yield contradictory results. In other words, what we want is that φ is not both true$_\sigma$ in $\mathfrak{M}$ and false$_\sigma$ in $\mathfrak{M}$. To show that this is the case we establish what is known as unique readability.

Lemma 5 (Unique Readability). *Let φ be a formula. Then exactly one of the following cases obtains:*

(1) *φ is atomic,*
(2) *$\varphi = \neg\chi$ for some χ;*
(3) *$\varphi = (\chi\&\chi)$ for some χ, χ';*
(4) *$\varphi = (\chi\vee\chi)$ for some χ, χ';*
(5) *$\varphi = (\chi\rightarrow\chi)$ for some χ, χ';*
(6) *$\varphi = \forall x\chi$ for some x, χ;*
(7) *$\varphi = \exists x\chi$ for some x, χ.*

Moreover, the formulae χ, χ', and the variable x are unique in the arising case.

This means that when given φ, either it is atomic and it is uniquely determined whether or not it is true. Or it is not atomic, and then it is uniquely composed from immediate subformulae which are uniquely determined. This guarantees that the structure of φ is unique even though it is technically a string. The claim is not self-evident and needs proof. I skip the proof in the interest of more urgent matters.

First, we shall say that a formula φ is a **tautology** if for every $\mathfrak{M}$ and every valuation: φ is true$_\sigma$ in $\mathfrak{M}$. φ is **satisfiable** if there is a $\mathfrak{M}$ and a σ such that φ is true$_\sigma$. It is quite possible that φ may not be satisfiable in a given $\mathfrak{M}$, but that does not exclude that it is satisfiable in some other $\mathfrak{M}'$! Similarly, given $\mathfrak{M}$, φ may be false$_\sigma$ and true$_{\sigma'}$ for some σ', which must be different from σ (by unique readability).

Tow formulae φ and χ are **equivalent** if for every $\mathfrak{M}$ and every σ: φ is true$_\sigma$ iff χ is true$_\sigma$. Equivalently, $(\varphi \vee \chi)$ as well as $(\chi \vee \varphi)$ are tautologies. (Can you see why?)

3. Quantification and Variables

I define what it means for a variable to occur **free** and **bound**. The definition is based on the formula qua string. In a string, an occurrence can be defined as some sequence of positions (see also my note on c-command and scope).

Definition 6. *Let φ be a formula. An occurrence of a variable x is said to be* ***free*** *if is not contained in a formula of the form $Qx\psi$ where Q is either $\exists$ or $\forall$. If it does not occur free it occurs bound.*

Here is a formula:

$$\forall\mathtt{x}(\mathtt{x0y}\to\exists\mathtt{y}\mathtt{Cy}) \quad (1)$$

In this string, x occurs 2 times, and y occurs 3 times. The first occurrence of x is this one:

$$\forall\underline{\mathtt{x}}(\mathtt{x0y}\to\exists\mathtt{y}\mathtt{Cy}) \quad (2)$$

The entire string contains the underlined occurrence, and it has the property that it is of the form $\forall\mathtt{x}\varphi$ where φ is a formula. So, the first occurrence of x is bound. For the same reason the second occurrence is also bound. Now, the first occurrence of y is this:

$$\forall\mathtt{x}(\mathtt{x0}\underline{\mathtt{y}}\to\exists\mathtt{y}\mathtt{Cy}) \quad (3)$$

It is not contained in formula of the form $\exists\mathtt{y}\varphi$, so it is free. The next two occurrences of y on the other hand are bound. Thus, one and the same variable can occur both bound and free. Let $\mathrm{fr}(\varphi)$ denote the set of variables that have a free occurrence in φ. Then for this formula, $\mathrm{fr}(\varphi) = \{\mathtt{y}\}$.

There is an additional fact that one sometimes needs to take care of, namely to see which occurrence of a variable is bound by which occurrence of a quantifier. This may arise exactly when there are several occurrences of Qx where x is a variable and Q a quantifier. For example, in the following formula $\forall\mathtt{x}$ occurs twice.

(4) $$\forall\mathtt{x(xOy}\to\exists\mathtt{xCx)}$$

We say that a given occurrence of a variable x is bound by the occurrence of Qx whhich has smallest scope among those that contain our fixed occurrence. Thus, the leftmost quantifier binds the first two occurrences of `x` since it is the only one that contains them. However, the two rightmost occurrences are not bound by the leftmost quantifier but rather by the one following it.

Lemma 7 (Coincidence Lemma). *Let φ be a formula, $\mathfrak{M}$ a structure and σ and σ' two valuations into $\mathfrak{M}$ such that $\sigma(x) = \sigma'(x)$ whenever $x \in \mathrm{fr}(\varphi)$. Then φ is true$_\sigma$ iff φ is true$_{\sigma'}$.*

I shall prove this claim. First, I show that it is true for all atomic formulae. To this end, let Fx be an atomic formula. Then $\mathrm{fr}(Fx) = \{x\}$. Let σ and σ' be assignments such that $\sigma'(x) = \sigma(x)$. Then Fx is true$_\sigma$ iff $\sigma(x)$ satisfies what F stands for iff $\sigma'(x)$ satisfies what F stands for iff Fx is true$_{\sigma'}$. If the formula is xRy the proof is essentially the same. Now, let the formula be $(\varphi\to\psi)$. Then it turns out that $\mathrm{fr}((\varphi\to\psi)) = \mathrm{fr}(\varphi) \cup \mathrm{fr}(\psi)$. This is seen as follows: every occurrence of a variable in $(\varphi\to\psi)$ can be uniquely traced to an occurrence of that variable in either φ or ψ. If that occurrence is free in $(\varphi\to\psi)$ there is no quantifier occurrence that binds the corresponding occurence in φ or ψ. If that occurrence is bound in $(\varphi\to\psi)$ the corresponding occurrence is also bound. Now let σ and σ' be valuations such that $\sigma(x) = \sigma(x')$ for all variables in $\mathrm{fr}((\varphi\to\psi))$. Then $\sigma(x) = \sigma'(x)$ for all $x \in \mathrm{fr}(\varphi)$ and for all $x \in \mathrm{fr}(\psi)$. Hence, φ is true$_\sigma$ iff it is true$_{\sigma'}$; and ψ is true$_\sigma$ iff it is true$_{\sigma'}$. Putting this together we get: $(\varphi\to\psi)$ is true$_\sigma$ iff φ is true$_\sigma$ and ψ is true$_\sigma$ iff φ is true$_{\sigma'}$ and ψ is true$_{\sigma'}$ iff $(\varphi\to\psi)$ is true$_{\sigma'}$. Similarly for $(\varphi\&\psi)$, $(\varphi\vee\psi)$ and $\neg\varphi$. Finally we treat the case of $\exists x\varphi$. It is easily seen that $\mathrm{fr}(\exists x\varphi) = \mathrm{fr}(\varphi) - \{x\}$. Assume that σ' is such that $\sigma(y) = \sigma'(y)$ for all $y \in \mathrm{fr}(\exists x\varphi)$. Suppose that $\exists x\varphi$ is true$_\sigma$. Then there is o such that φ is true$_{\sigma[x/o]}$. It turns out that $\sigma[x/o](y) = \sigma'[x/o](y)$ for every $y \in \mathrm{fr}(\varphi)$. Hence, by assumption on φ, φ is true$_{\sigma'[x/o]}$, and thus that $\exists x\varphi$ is true$_{\sigma'}$. The roles of σ and σ' are interchangeable, whence the equivalence holds in this case, too.

Let φ be such that $\mathrm{fr}(\varphi) = \varnothing$. Such formulae are called **sentences**.

Lemma 8. *Let φ be a sentence and $\mathfrak{M}$ a structure. Then if there is an assignment σ such that φ is true$_\sigma$ in $\mathfrak{M}$ then for all valuations σ' into $\mathfrak{M}$ φ is true$_{\sigma'}$.*

This is easy to see. Under the assumptions, let σ' be any valuation. Then for all variables $x \in \mathrm{fr}(\varphi)$ it is true that $\sigma(x) = \sigma'(x)$, since there aren't anyt such variables. By the Concidence Lemma, it follows now that if σ' is an arbitrary valuation then φ is true$_{\sigma'}$.

Another important fact is this. First, some notation: $\varphi[y/x]$ denotes the result of replacing every free occurrence of x by y. Since x may occur also bound, this will not necessarily remove every occurrence of x. Also, be aware of the fact that $[y/x]$ is not part of our language. It just denotes some formula when given one. (For assignments, the notation $\sigma[x/o]$ means something different!)

If x does not occur free in φ then obviously $\varphi[y/x] = \varphi$, otherwise they are different. Now, the formula $\varphi[y/x]$ has y where x used to occur freely. The two formulae are therefore not much different. We are interested in the case where y does not occur free in φ. Now suppose σ is an assignment, and let $o = \sigma(x)$. Now $\sigma[y/o]$ is such that φ is true$_{\sigma[y/o]}$ (by the Coincidence Lemma) and, moreover, $\varphi[y/x]$ is true$_{\sigma[y/o]}$, since now x and y receive the same value and so it does not matter whether you have x occur freely or y.

Lemma 9. *Let φ be such that y does not occur free in φ. Then φ is true$_\sigma$ in $\mathfrak{M}$ iff $\varphi[y/x]$ is true$_{\sigma[y/\sigma(x)]}$ in $\mathfrak{M}$.*

Lemma 10 (Renaming of Bound Variables)**.** *Let φ a formula in which y does not occur free and $\mathfrak{M}$ a structure. Let $Q = \forall$ or $Q = \exists$. Then for every assignment σ into $\mathfrak{M}$, $Qx\varphi$ is true$_\sigma$ in $\mathfrak{M}$ iff $Qy(\varphi[y/x])$ is true$_\sigma$ in $\mathfrak{M}$.*

Before we begin, let us note that the roles of x and y are completely symmetrical. x does not occur free in $\varphi[y/x]$, and it turns out that $\varphi[y/x][x/y] = \varphi$. (Can you see this?) Hence, we need to show only one direction. We treat $Q = \forall$. Now suppose σ is an assignment and $Qx\varphi$ is true$_\sigma$. Then there is an o such that φ is true$_{\sigma[x/o]}$. Then $\varphi[y/x]$ is true$_{\sigma[x/o][y/o]}$. But does not occur free in $\varphi[y/x]$, so by the Coincidence Lemma, $\varphi[y/x]$ is true$_{\sigma[y/o]}$. Hence $Qy(\varphi[y/x])$ is true$_\sigma$.

4. Quantifier Raising and Translation

Rather than formulating the truth conditions of a sentence, we provide a translation procedure from sentential structures into a language formulae. It is different from the language used so far. It has generalised quantifiers in place of just $\exists$ and $\forall$. Moreover, it has a different bracketing convention. I shall not rehearse the exact definition of the language here; this is done in the manuscript. I only depart from the manuscript by enclosing negated statements into brackets, to ensure unique readability.

The translation function is called τ. However, it will become apparent below that in addition to τ we need auxiliary functions τ^x, for each variable x. When we write τ° this will denote either τ or any of the τ^x.

The base cases are provided by the atomic formulae. We assume for the next definition we assume that F is an intransitive verb, G a transitive verb, m, n variables or proper names. The convention for proper names and verbs is as follows: they will be translated by τ by the corresponding bold face version, with all inflections removed, which is to say effectively that we leave open what exactly will go in their place. This is up to lexical semantics. Variables will be translated by themselves. Since the primitive symbols of the alphabet are actually strings, we insert blanks to obtain unique segmentation. The blank is denoted by $\Box$. Thus, we have

① $\tau^\circ(x) := x$.
② $\tau^\circ([_{\mathrm{VP}}[_{\mathrm{DP}}m][_{\mathrm{V}'}F]]) := \tau^\circ(F)^\frown\Box^\frown\tau^\circ(m)$
③ $\tau^\circ([_{\mathrm{VP}}[_{\mathrm{DP}}m][_{\mathrm{V}'}F][_{\mathrm{DP}}n]]) := \tau^\circ(m)^\frown\Box^\frown\tau^\circ(F)^\frown\Box^\frown\tau^\circ(n)$

Thus,

(5)
$$\begin{aligned}\tau([_{\mathrm{VP}}[_{\mathrm{DP}}\texttt{Maria}][_{\mathrm{V}'}\texttt{runs}]]) &= \tau(\texttt{runs})^\frown\Box^\frown\tau(\texttt{Maria})\\ &= \mathbf{run}^\frown\Box^\frown\mathbf{maria}\\ &= \mathbf{run\ maria}\end{aligned}$$

This is either true or false in a given structure (independently of the assignment). It is true if $I(\mathbf{maria}) \in I(\mathbf{run})$. Next we specify the translation for negation as follows. Negation can be at the left periphery of a sentence, as follows:

(6)
$$[_{\mathrm{S}}[_{\mathrm{Neg}}\texttt{not}]T]$$

Here, T abbreviates a constituent of category S. In this case, we put

(7)
$$\tau^\circ([_{\mathrm{S}}[_{\mathrm{Neg}}\texttt{not}]\ T]) := \texttt{(}\neg^\frown\tau^\circ(T)^\frown\texttt{)}$$

This says the following: if we choose τ on the left, then on the right we also have τ, and if we have τ^x on the left then we also have τ^x on the right.

The rule of quantifier raising (QR) is formulated on structures. Its input is an ordered tree with labels on the nodes, and it returns another such tree. I draw the trees linearly. The input to QR is this structure:

$$...[_{\mathrm{S}}...[_{\mathrm{DP}}T]...]... \tag{8}$$

Here T is the tree dominated by the node with label DP. The output is

$$...[_{\mathrm{S}}[_{\mathrm{DP},x}T][_{\mathrm{S}}...[_{\mathrm{DP}}x]...]... \tag{9}$$

where x is a variable not already occurring in the input tree. The output tree now contains occurrences of variables. These were not part of the original sentence; they have been 'smuggled in' by applying QR. The translation into logical form will translate the variables by themselves. However, we also need to see what happens to the structure labelled 'DP, x'. Here, the variable is distributed over all nodes by using τ^x in place of τ. (In the following, X may be either N or N$'$.)

$$\tau^\circ([_{\mathrm{S}}[_{\mathrm{DP},x}T][_{\mathrm{S}}U]]) :=\tau^\circ([_{\mathrm{DP},x}T])^\frown\Box^\frown\tau^\circ([_{S}U]) \tag{10}$$

$$\tau^\circ([_{\mathrm{S}}[_{\mathrm{VP}}U]]) :=\tau^\circ([_{\mathrm{VP}}U]) \tag{11}$$

$$\tau^\circ([_{\mathrm{DP},x}[_{\mathrm{D}}Q][_{X}F]]) :=\tau^x([_{\mathrm{D}}Q])^\frown\{^\frown\tau^x([_{X}F])^\frown\} \tag{12}$$

The base cases are:

$$\begin{aligned}\tau^x([_{\mathrm{D}}V]) &= \tau(V)^\frown\Box^\frown x\\ \tau^x([_{\mathrm{N}}W]) &= \tau(W)^\frown\Box^\frown x\end{aligned} \tag{13}$$

The case where we have to apply τ in place of τ^x does not arise in well-formed structures. Thus it is safe to declare that τ is undefined for such input. Thus

$$\begin{aligned}&\tau^{\mathtt{x}}([_{\mathrm{DP},\mathtt{y}}[_{\mathrm{D}}\mathtt{every}][_{\mathrm{N}}\mathtt{dog}]])\\ =&\tau^{\mathtt{y}}([_{\mathrm{D}}\mathtt{every}])^\frown\{^\frown\tau^{\mathtt{y}}([_{\mathrm{N}}\mathtt{dog}])^\frown\}\\ =&\mathbf{every}^\frown\Box^\frown\mathtt{y}^\frown\{^\frown\mathbf{dog}^\frown\Box^\frown\mathtt{y}^\frown\}\\ =&\mathbf{every}\ \mathtt{y}\{\mathbf{dog}\ \mathtt{y}\}\end{aligned} \tag{14}$$

(Notice that the spaces are inserted by the translation. Thus everything except the linebreaks is predetermined!) This is an incomplete expression. The remainder of that expression is supplied by the other part of the tree.

Adjectives are treated as follows (where X can be either N or N′):

(15) $\tau^\circ([_{\mathrm{N}'}[_{\mathrm{Adj}}A][_X F]]) := \tau^\circ([_{\mathrm{Adj}}A])^\frown\square^\frown\texttt{\&}^\frown\square^\frown\tau^\circ([_X F])$

If the structure is well-formed the function τ will not appear here, so it is enough to specialize to τ^x:

(16) $\tau^x([_{\mathrm{N}'}[_{\mathrm{Adj}}A][_X F]]) := \tau^x([_{\mathrm{Adj}}A])^\frown\square^\frown\texttt{\&}^\frown\square^\frown\tau^x([_X F])$

Finally, the translation of adjectives is the same as for nouns. We agree here that $\tau(A)$ simply renders the typewriter font into boldface: $\tau(\texttt{brown}) = \mathbf{dog}$.

(17) $\tau^x([_{\mathrm{Adj}}A]) := \tau(A)^\frown\square^\frown x$

The last thing we need to address is the relative clauses. Here, matters are somewhat more complicated. We assume that movement, even before S-structure, does essentially the same as quantifier raising: it moves a constituent, indexes it with a variable and leaves that variable where the constituent has been. (Notice that we use variables where syntactic theory uses indices. This is for all intents and purposes the same thing. Variables are just a little easier to use here.) In particular, the word `who` is assumed to have been moved from somewhere else. The surface structure therefore looks more like this:

(18) $\texttt{boy}\ [_{\mathrm{RC}}[_{\mathrm{C,x}}\texttt{who}]\ \texttt{Maria likes x}]$

It has been derived from

(19) $\texttt{boy}\ [_{\mathrm{RC}}[_{\mathrm{C}}\varnothing][_{\mathrm{S}}\ \texttt{Maria likes who}]]$

Now, since the translation applies to LF it need not be concerned with questions such as whether movement happened before or after S-structure. It gets the structure in (18) rather than the structure in (19). However, in case the relative clause sits in a quantified expression such as `every boy who Maria likes owns a house` we need to see to it that the association between variables and quantifiers is not broken. Every quantifier sends down a variable to be put where one is needed. Now, `who Maria likes` is like an adjective, but it has the form `who Maria likes x`, with an occurrence of the variable already put in. The translation does not perform substitutions of any kind, so it will not rewrite the variable. Instead, what it will do is to assume the following translation for `who`:

(20) $\tau^y([_{\mathrm{C},x}\texttt{who}]) := (^\frown y^\frown\square^\frown\texttt{=}^\frown\square^\frown x^\frown)$

This translation does the following. When it enters the clause in the form of τ^y it will add the condition that x is the same as y upon which it is effectively the same whether we write x or y. In particular the occurrence of x at the trace position will not get replaced. This replacement is now needless. We need one more set of rules (X either N or N′):

(21) $\tau^\circ([_{\mathrm{N}'}[_X A][_{\mathrm{RC}} F]]) := \tau^\circ([_X A])^\frown\Box^\frown\&^\frown\Box^\frown\tau^\circ([_{\mathrm{RC}} F])$

(22) $\tau^\circ([_{\mathrm{RC}}[_{\mathrm{C},x} A][_{\mathrm{S}} F]]) := \tau^\circ([_{\mathrm{C},x} A])^\frown\Box^\frown\&^\frown\Box^\frown\tau^\circ([_{\mathrm{S}} F])$

It is certainly possible to unify these rules into a couple of all purpose rules, but I shall not perform this reduction.

Let us perform the translation on the sentence

(23) `Every boy who two girls like owns a house.`

First, `who` is assumed to have moved from object position, so we actually have the surface structure

(24) $\texttt{every boy } [_{\mathrm{RC}}[_{\mathrm{C,x}}\texttt{who}][_{\mathrm{S}}\texttt{two girls like x}]]\texttt{owns a house}.$

Next we perform quantifier raising. This gives us (among other) this structure:

(25) $[_{\mathrm{S}}[_{\mathrm{DP,y}}\texttt{ every boy } [_{\mathrm{S}}[_{\mathrm{RC}}[_{\mathrm{C,x}}\texttt{who}][_{\mathrm{S}}[_{\mathrm{DP,z}}\texttt{two girls}][_{\mathrm{S}}\texttt{z like x}]]]]]$
$[_{\mathrm{S}}[_{\mathrm{DP,u}}\texttt{a house}]\texttt{y owns u}]].$

Call this structure U. We have $U = [_{\mathrm{S}}[_{\mathrm{DP,y}}V]\ W]$, where $[_{\mathrm{DP,y}}V]$ is on the first line of (25) and W on the second. This is what we translate using τ:

(26) $\tau(U) = \tau([_{\mathrm{DP,y}}V])^\frown\Box^\frown\tau(W)$

Now, $W = [_{\mathrm{S}}[_{\mathrm{DP,u}}\texttt{a house}]\texttt{y owns u}]]$.

(27)
$$\begin{aligned}
&\tau([_{\mathrm{S}}[_{\mathrm{DP,u}}\texttt{a house}]\texttt{y owns u}]])\\
=&\tau([_{\mathrm{DP,u}}\texttt{a house}])^\frown\Box^\frown\tau(\texttt{y owns u})\\
=&\tau^{\mathtt{u}}(\mathrm{a})^\frown\{^\frown\tau^{\mathtt{u}}(\texttt{house})^\frown\}^\frown\Box^\frown\texttt{y } \mathbf{own}\texttt{ u}\\
=&\mathbf{a}\texttt{ u}\{\mathbf{house}\texttt{ u}\}\texttt{ y }\mathbf{own}\texttt{ u}
\end{aligned}$$

Next we turn to the translation of V:

$$\tau([_{\mathrm{DP},\mathtt{y}}\ \texttt{every boy}\ [_{\mathrm{S}}[_{\mathrm{RC}}[_{\mathrm{C},\mathtt{x}}\texttt{who}][_{\mathrm{S}}[_{\mathrm{DP},\mathtt{z}}\texttt{two girls}][_{\mathrm{S}}\ \texttt{z like x}]]]]])$$

$$=\tau^{\mathtt{y}}(\texttt{every})^\frown\{^\frown\tau^{\mathtt{y}}([_{\mathrm{N}'}\texttt{boy})[_{\mathrm{S}}[_{\mathrm{RC}}[_{\mathrm{C},\mathtt{x}}\texttt{who}][_{\mathrm{S}}[_{\mathrm{DP},\mathtt{z}}\texttt{two girls}][_{\mathrm{S}}\ \texttt{z like x}]]]]])^\frown\}$$

$$=\textbf{every}\ \mathtt{y}\{\tau^{\mathtt{y}}(\texttt{boy})^\frown\Box^\frown\texttt{\&}^\frown\Box^\frown\tau^{\mathtt{y}}([_{\mathrm{RC}}[_{\mathrm{C},\mathtt{x}}\texttt{who}][_{\mathrm{S}}[_{\mathrm{DP},\mathtt{z}}\texttt{two girls}][_{\mathrm{S}}\ \texttt{z like x}]]])^\frown\}$$

$$=\textbf{every}\ \mathtt{y}\{\textbf{boy}\ \mathtt{y}\ \texttt{\&}\ \tau^{\mathtt{y}}([_{\mathrm{C},\mathtt{x}}\texttt{who}])^\frown\Box^\frown\texttt{\&}^\frown\Box^\frown\tau^{\mathtt{y}}([_{\mathrm{S}}[_{\mathrm{DP},\mathtt{z}}\texttt{two girls}][_{\mathrm{S}}\texttt{z like x}]])^\frown\} \tag{28}$$

$$=\textbf{every}\ \mathtt{y}\{\textbf{boy}\ \mathtt{y}\ \texttt{\&}\ \texttt{(y = x)}^\frown\Box^\frown\texttt{\&}^\frown\Box^\frown\tau^{\mathtt{y}}([_{\mathrm{S}}[_{\mathrm{DP},\mathtt{z}}\texttt{two girls}][_{\mathrm{S}}\texttt{z like x}]])^\frown\}$$

$$=\textbf{every}\ \mathtt{y}\{\textbf{boy}\ \mathtt{y}\ \texttt{\&}\ \texttt{(y = x)}\ \texttt{\&}\ \tau^{\mathtt{y}}([_{\mathrm{DP},\mathtt{z}}\texttt{two girls}])^\frown\Box^\frown\tau^{\mathtt{y}}([_{\mathrm{S}}\texttt{z like x}])^\frown\}$$

$$=\textbf{every}\ \mathtt{y}\{\textbf{boy}\ \mathtt{y}\ \texttt{\&}\ \texttt{(y = x)}\ \texttt{\&}\ \textbf{two}\ \mathtt{z}\{\textbf{girl}\ \mathtt{z}\}\ \texttt{z like x}\}$$

The overall translation is now

$$\begin{aligned}\tau(U) &=\tau(V)^\frown\Box^\frown\tau(W)\\ &=\textbf{every}\ \mathtt{y}\{\textbf{boy}\ \mathtt{y}\ \texttt{\&}\ \texttt{(y = x)}\ \texttt{\&}\ \textbf{two}\ \mathtt{z}\{\textbf{girl}\ \mathtt{z}\}\ \texttt{z like x}\}^\frown\Box^\frown\textbf{a}\ \mathtt{u}\{\textbf{house}\ \mathtt{u}\}\ \mathtt{y}\ \textbf{own}\ \mathtt{u}\\ &=\textbf{every}\ \mathtt{y}\{\textbf{boy}\ \mathtt{y}\ \texttt{\&}\ \texttt{(y = x)}\ \texttt{\&}\ \textbf{two}\ \mathtt{z}\{\textbf{girl}\ \mathtt{z}\}\ \texttt{z like x}\}\ \textbf{a}\ \mathtt{u}\ \{\textbf{house}\ \mathtt{u}\}\ \mathtt{y}\ \textbf{own}\ \mathtt{u}\end{aligned} \tag{29}$$

This is the translation, obtained in a completely mechanical manner. It can be simplified (using logical equivalences) to

$$\textbf{every}\ \mathtt{y}\{\textbf{boy}\ \mathtt{y}\ \texttt{\&}\ \textbf{two}\ \mathtt{z}\{\textbf{girl}\ \mathtt{z}\}\ \texttt{z like y}\}\ \textbf{a}\ \mathtt{u}\ \{\textbf{house}\ \mathtt{u}\}\ \mathtt{y}\ \textbf{own}\ \mathtt{u} \tag{30}$$

However, remember that this latter form is not what you get from the translation, it is only *equivalent* to the result.

We need to address two questions: (a) Does it matter which variable we choose when applying QR? Answer: No, as long as we always choose an unused variable (or at least one that has not free occurrences). This is due to the result above called 'Renaming of Bound Variables'. (b) Does it matter which quantifier we raise first? Answer: No, because the competing QR can always be performed.

Notice finally that there is a mismatch in constituent structure between the linguistic structures and the logical structures. The manuscript spells out **every** $x\{\varphi\}$ as a constituent. However, constituent receive interpretation, and this string is not interpreted. Only the string **every** $x\{\varphi\}\psi$ is interpreted. Yet, at this point the distinction is irrelevant. The translation procedure is such that it always yields a formula at the end, one which can be interpreted.

Department of Linguistics, UCLA, 3125 Campbell Hall, Los Angeles, CA 90095-1543