

## 1. Bound Pronouns

In addition to deictic pronouns, lazy pronouns, and reflexive pronouns, there are also *bound pronouns*, which are needed to account for the following example,

every man's mother respects his father

which admits the following reading,

$$\forall x \{ \mathbf{M}x \rightarrow \mathbf{R}[\mathbf{m}(x), \mathbf{f}(x)] \}$$

according to which ‘he’ [inside ‘his’] is anaphoric to ‘every man’. Now, ‘he’ cannot be lazy-anaphoric to ‘every man’, since ‘he’ cannot simply be replaced by ‘every man’, and ‘he’ cannot be reflexively-anaphoric to ‘every man’, since ‘every man’ does not fill the subject role.

Thus, we have yet another kind of anaphoric pronoun, which needs to be explained.

## 2. Anaphoric Case-Markers

In order to account for the above anaphoric relation, we introduce **anaphoric case-markers**, which behave *mostly* like ordinary case-markers, and which provide a systematic technique of binding pronouns to their antecedents. We already use non-negative integers to encode the ordinary case-markers, so we propose to use negative integers for anaphoric case-markers. For example, in the following sentence

### 1. Example 1

Jay [-1] respects (-1) his mother

the negative integers indicate that ‘his’<sup>1</sup> is bound by ‘Jay’. Notice the distinction between ‘[-1]’ and ‘(-1)’, which is based on the asymmetry between a pronoun and the phrase that binds it; the antecedent phrase binds the pronoun; not the other way around.<sup>2</sup> There is a corresponding distinction, and asymmetry, in the respective categorial implementations.<sup>3</sup>

type{[ $\alpha$ ]}	=	$D \rightarrow D_\alpha$	type{( $\alpha$ )}	=	$D_\alpha \rightarrow D$
[[ $\alpha$ ]]	=	$\lambda x \{x_\alpha\}$	[( $\alpha$ )]	=	$\lambda x_\alpha \{x\}$
here, $\alpha$ is an anaphoric case-marker (negative integer)					

The tree for Example 1 then is:

Jay	[+1]	[-1]	respects	(-1)	he	's	mother	[+2]
	$\lambda x \{x_1\}$	$\lambda x \{x_{-1}\}$		$\lambda x_{-1} \{x\}$	$\emptyset$	$\lambda x \{x_6\}$		
J	$\lambda x \{x_1 \times x_{-1}\}$			$\lambda x_{-1} \{x_6\}$		$\lambda x_6 \{\mathbf{m}(x)\}$		
				$\lambda x_{-1} \{\mathbf{m}(x)\}$			$\lambda x \{x_2\}$	
			$\lambda y_2 \lambda x_1 \mathbf{R}xy$	$\lambda x_{-1} \{\mathbf{m}(x)_2\}$				
	$J_1 \times J_{-1}$		$\lambda y_{-1} \lambda x_1 \mathbf{R}[x, \mathbf{m}(y)]$					
$\mathbf{R}[J, \mathbf{m}(J)]$								

<sup>1</sup> More technically accurately speaking, ‘e’ [inside ‘he’ inside ‘his’] is bound by ‘Jay’.

<sup>2</sup> By convention, we place the "backward looking" marker (-1) in front of the pronoun it marks.

<sup>3</sup> This is a simplified, and temporary, account of [ $\alpha$ ]. Later, we consider a more general account, which treats [ $\alpha$ ] in a manner that is similar to [ref]. We will retain our account of ( $\alpha$ ).

## 2. Comparison with (variation on) reflexive

Jay	[+1]	[ref <sub>1,-1</sub> ]	respects	(-1)	he	's [+6]	mother	[+2]
	$\lambda x\{x_1\}$	$\lambda x_1\{x_1 \times x_{-1}\}$		$\lambda x_{-1}\{x\}$	$\emptyset$	$\lambda x\{x_6\}$		
J	$\lambda x\{x_1 \times x_{-1}\}$			$\lambda x_{-1}\{x_6\}$		$\lambda x_6\{\mathbf{m}(x)\}$		
				$\lambda x_{-1}\{\mathbf{m}(x)\}$			$\lambda x\{x_2\}$	
			$\lambda y_2 \lambda x_1 \mathbf{R}xy$	$\lambda x_{-1}\{\mathbf{m}(x)_2\}$				
$J_1 \times J_2$			$\lambda y_{-1} \lambda x_1 \mathbf{R}[x, \mathbf{m}(y)]$					
$\mathbf{R}[J, \mathbf{m}(J)]$								

Jay	[+1]	[ref <sub>1,-1</sub> ]	respects	(-1)	he	's	mother	[+2]
				$\lambda x_{-1}\{x\}$	$\emptyset$	$\lambda x\{x_6\}$		
J	$\lambda x\{x_1\}$			$\lambda x_{-1}\{x_6\}$		$\lambda x_6\{\mathbf{m}(x)\}$		
				$\lambda x_{-1}\{\mathbf{m}(x)\}$			$\lambda x\{x_2\}$	
			$\lambda y_2 \lambda x_1 \mathbf{R}xy$	$\lambda x_{-1}\{\mathbf{m}(x)_2\}$				
		$\lambda x_1\{x_1 \times x_{-1}\}$	$\lambda y_{-1} \lambda x_1 \mathbf{R}[x, \mathbf{m}(y)]$					
$J_1$		$\lambda x_1 \mathbf{R}[x, \mathbf{m}(x)]$						
$\mathbf{R}[J, \mathbf{m}(J)]$								

The huge difference is that, according to the latter analysis, ‘respects his mother’ is a true-blue VP [type  $D_1 \rightarrow S$ ], whereas according to the former two analyses, ‘respects his mother’ is an open-VP, since it has an open-place associated with anaphoric binding.

## 3. Example 2

Jay [-1] respects (-1) his mother if (-1) he is virtuous

Jay	[+1]	[-1]	respects	(-1)	he	's	mother	[+2]	if	(-1)	he	[+1]	is	virtuous	
	$\lambda x\{x_1\}$	$\lambda x\{x_{-1}\}$		$\lambda x_{-1}\{x\}$	$\emptyset$	$\lambda x\{x_6\}$				$\lambda x_{-1}\{x\}$	$\emptyset$	$\lambda x\{x_1\}$	$\lambda P_0$	$\mathbf{V}_0$	
J	$\lambda x\{x_1 \times x_{-1}\}$			$\lambda x_{-1}\{x_6\}$		$\lambda x_6$				$\lambda x_{-1}\{x_1\}$		$\{\mathbf{P}_1\}$	$\mathbf{V}_1$		
				$\lambda x_{-1}\{\mathbf{m}(x)\}$			$\lambda x\{x_2\}$		$\lambda P \lambda Q$	$\lambda x_{-1} Vx$					
			$\lambda y_2 \lambda x_1$	$\lambda x_{-1}\{\mathbf{m}(x)_2\}$						$\{\mathbf{P} \rightarrow \mathbf{Q}\}$					
			$\mathbf{R}xy$	$\lambda y_{-1} \lambda x_1 \mathbf{R}[x, \mathbf{m}(y)]$											
$J_1 \times J_{-1} \times J_{-1}$			$\mathbf{R}[J, \mathbf{m}(J)] \times J_{-1}$									$\lambda x_{-1} \lambda Q \{Vx \rightarrow Q\}$			
$\mathbf{V}_J \rightarrow \mathbf{R}[J, \mathbf{m}(J)]$															

Notice that in order to use  $J_{-1}$  to bind/fill both occurrences of ‘he’, we need a further rule that authorizes **anaphoric-duplication**, according to which an anaphorically-marked D-phrase can be repeatedly used. The categorial-logic rule is officially given as follows.

$$\frac{\partial_\alpha}{\partial_\alpha \times \partial_\alpha} \quad [\alpha\text{-duplication}]$$

Here,  $\alpha$  is an anaphoric case-marker (negative integer), and  $\partial$  is an expression of type D.

1. Jay [-1] respects (-1) his mother [-2] if [-2] she is virtuous

Jay	[+1]	[-1]	respects	(-1)	he	's	mother	[+2]	if (-2) she [+1] is virtuous	
	$\lambda x\{x_1\}$	$\lambda x\{x_{-1}\}$		$\lambda x_{-1}\{x\}$	$\emptyset$	$\lambda x\{x_6\}$				
J	$\lambda x\{x_1 \times x_{-1}\}$			$\lambda x_{-1}\{x_6\}$		$\lambda x_6\{\mathbf{m}(x)\}$				
				$\lambda x_{-1}\{\mathbf{m}(x)\}$			$\lambda x\{x_2\}$			
			$\lambda y_2 \lambda x_1 \mathbf{R}xy$	$\lambda x_{-1}\{\mathbf{m}(x)_2\}$						
$J_1 \times J_{-1}$			$\lambda y_{-1} \lambda x_1 \mathbf{R}[x, \mathbf{m}(y)]$							
$\mathbf{R}[J, \mathbf{m}(J)] \times J_{-1}$								(from below)	$\lambda x_{-1} \lambda Q \{Vx \rightarrow Q\}$	
$\mathbf{V}J \rightarrow \mathbf{R}[J, \mathbf{m}(J)]$										

if	(-2)	she	[+1]	is	virtuous
	$\lambda x_{-1}\{x\}$	$\emptyset$	$\lambda x\{x_1\}$	$\lambda P_0\{P_1\}$	$\mathbf{V}_0$
	$\lambda x_{-1}\{x_1\}$		$\mathbf{V}_1$		
$\lambda P \lambda Q \{P \rightarrow Q\}$	$\lambda x_{-1} Vx$				
$\lambda x_{-1} \lambda Q \{Vx \rightarrow Q\}$					

### 3. Applying these Ideas to QPs

In the previous examples, we proposed that items of type D can bind pronouns. This approach can also be applied to QPs, as in the following examples.

#### 1. Example 1

every man [-1] respects (-1) his mother

every	man	[+1]	[-1]	respects	(-1)	he	's [+6]	mother	[+2]
$\lambda P_0 \lambda Q \forall x \{Px \rightarrow Qx\}$	$\lambda x_0 \mathbf{M}x$	$\lambda x\{x_1\}$	$\lambda x\{x_{-1}\}$		$\lambda x_{-1}\{x\}$	$\emptyset$	$\lambda x\{x_6\}$		
$\lambda Q \forall x \{ \mathbf{M}x \rightarrow Qx \}$		$\lambda x\{x_1 \times x_{-1}\}$			$\lambda x_{-1}\{x_6\}$		$\lambda x_6\{\mathbf{m}(x)\}$		
					$\lambda x_{-1}\{\mathbf{m}(x)\}$			$\lambda x\{x_2\}$	
				$\lambda y_2 \lambda x_1 \mathbf{R}xy$	$\lambda x_{-1}\{\mathbf{m}(x)_2\}$				
$\lambda Q_{1-1} \forall x \{ \mathbf{M}x \rightarrow Qxx \}$				$\lambda y_{-1} \lambda x_1 \mathbf{R}[x, \mathbf{m}(y)]$					
$\forall x \{ \mathbf{M}x \rightarrow \mathbf{R}[x, \mathbf{m}(x)] \}$									

In the above tree, we introduce new notation for double-inflection. In particular,

$$Q_{1-1} \stackrel{\text{df}}{=} \lambda x_1 y_{-1} Qxy$$

Thus,  $\llbracket \text{every man} \rrbracket_{+1-1}$  takes a two-place predicate as input, and delivers a sentence as output.

## 2. Example 2

every man's mother respects his father

every	man	's	[-1]	mother	[+1]	respects	(-1)	he	's	father	[+2]
$\lambda P_0 \lambda Q \forall x \{Px \rightarrow Qx\}$	$\mathbf{M}_0$	$\lambda x \{x_6\}$	$\lambda x \{x_{-1}\}$				$\lambda x_{-1} \{x\}$	$\emptyset$	$\lambda x \{x_6\}$		
$\lambda Q \forall x \{ \mathbf{M}x \rightarrow Qx \}$		$\lambda x \{x_6 \times x_{-1}\}$					$\lambda x_{-1} \{x_6\}$		$\lambda x_6 \{\mathbf{f}(x)\}$		
$\lambda Q_{6-1} \forall x \{ \mathbf{M}x \rightarrow Qxx \}$		$\lambda x_6 \{\mathbf{m}(x)\}$					$\lambda x_{-1} \{\mathbf{f}(x)\}$			$\lambda x \{x_2\}$	
$\lambda Q_{\emptyset-1} \forall x \{ \mathbf{M}x \rightarrow Q[\mathbf{m}(x),x] \}$		$\lambda x \{x_1\}$				$\lambda y_2 \lambda x_1 \mathbf{R}xy$	$\lambda x_{-1} \{\mathbf{f}(x)_2\}$				
$\lambda Q_{1-1} \forall x \{ \mathbf{M}x \rightarrow Q[\mathbf{m}(x),x] \}$						$\lambda y_{-1} \lambda x_1 \mathbf{R}[x,\mathbf{f}(y)]$					
$\forall x \{ \mathbf{M}x \rightarrow \mathbf{R}[\mathbf{m}(x),\mathbf{f}(x)] \}$											

In the above tree, we introduce further notation for double-inflection. In particular,

$$Q_{\emptyset-1} \stackrel{\text{df}}{=} \lambda xy_{-1} Qxy$$

Notice the marker ' $\emptyset$ ' which indicates a *lack* of inflection.

## 3. Example 3

every man whose mother respects his father respects his father

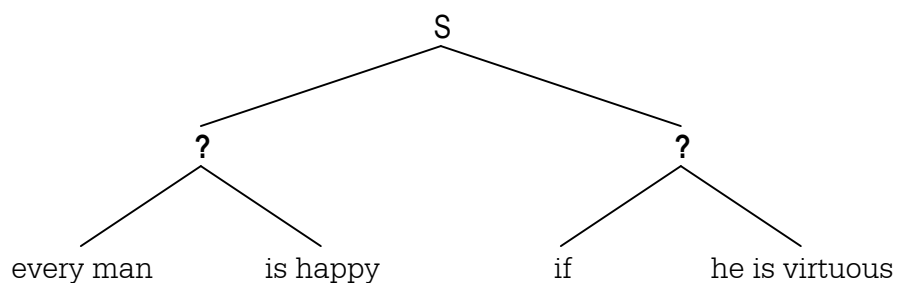
every	man whose mother respects his father	[+1]	[-2]	respects	(-2)	he	's	father	[+2]
$\lambda P_0 \lambda Q \forall x \{Px \rightarrow Qx\}$	(from below) $\lambda x_0 \{ \mathbf{M}x \ \& \ \mathbf{R}[\mathbf{m}(x),\mathbf{f}(x)] \}$	$\lambda x \{x_1\}$	$\lambda x \{x_{-2}\}$		$\lambda x_{-2} \{x\}$	$\emptyset$	$\lambda x \{x_6\}$		
$\lambda Q \forall x \{ \{ \mathbf{M}x \ \& \ \mathbf{R}[\mathbf{m}(x),\mathbf{f}(x)] \} \rightarrow Qx \}$		$\lambda x \{x_1 \times x_{-2}\}$			$\lambda x_{-2} \{x_6\}$		$\lambda x_6 \{\mathbf{f}(x)\}$		
					$\lambda x_{-2} \{\mathbf{f}(x)\}$			$\lambda x \{x_2\}$	
				$\lambda y_2 \lambda x_1 \mathbf{R}xy$	$\lambda x_{-2} \{\mathbf{f}(x)_2\}$				
$\lambda Q_{1-2} \forall x \{ \{ \mathbf{M}x \ \& \ \mathbf{R}[\mathbf{m}(x),\mathbf{f}(x)] \} \rightarrow Qxx \}$				$\lambda y_{-2} \lambda x_1 \mathbf{R}[x,\mathbf{f}(y)]$					
$\forall x \{ \{ \mathbf{M}x \ \& \ \mathbf{R}[\mathbf{m}(x),\mathbf{f}(x)] \} \rightarrow \mathbf{R}[x,\mathbf{f}(x)] \}$									

man	who	se	[-1]	mother	[+1]	respects	(-1)	he	's	father	[+2]
$\lambda Q \lambda P_0 \lambda x_0 \{Px \& Qx\}$		$\lambda x \{x_6\}$	$\lambda x \{x_{-1}\}$				$\lambda x_{-1} \{x\}$	$\emptyset$	$\lambda x \{x_6\}$		
		$\lambda x \{x_6 \times x_{-1}\}$					$\lambda x_{-1} \{x_6\}$		$\lambda x_6 \{f(x)\}$		
		$\lambda Q_{6-1} \lambda P_0 \lambda x_0 \{Px \& Qxx\}$		$\lambda x_6 \{m(x)\}$			$\lambda x_{-1} \{f(x)\}$				$\lambda x \{x_2\}$
		$\lambda Q_{0-1} \lambda P_0 \lambda x_0 \{Px \& Q[m(x),x]\}$		$\lambda x \{x_1\}$	$\lambda y_2 \lambda x_1 Rxy$	$\lambda x_{-1} \{f(x)_2\}$					
		$\lambda Q_{1-1} \lambda P_0 \lambda x_0 \{Px \& Q[m(x),x]\}$		$\lambda y_{-1} \lambda x_1 R[x,f(y)]$							
<b>M<sub>0</sub></b>		$\lambda P_0 \lambda x_0 \{Px \& R[m(x),f(x)]\}$									
$\lambda x_0 \{Mx \& R[m(x),f(x)]\}$											

### 4. Example 4

every man is happy if he is virtuous

There are analyses of this that are computationally *fairly* simple, and are categorially permitted, but do not do justice to what would seem to be the natural syntactic structure. In particular, it seems that the above sentence parses in the following manner.



In particular, a proper semantic analysis should make ‘every man is happy’ a constituent. Unfortunately, such an analysis involves some computational heavy-lifting.

every	man	[+1]	[-1]	is	happy	if	(-1)	he	[+1]	is	virtuous
$\lambda P_0 \lambda Q \forall x \{Px \rightarrow Qx\}$	<b>M<sub>0</sub></b>	$\lambda x \{x_1\}$	$\lambda x \{x_{-1}\}$	$\lambda P_0 \{P_1\}$	<b>H<sub>0</sub></b>		$\lambda x_{-1} \{x\}$	$\emptyset$	$\lambda x \{x_1\}$	$\lambda P_0 \{P_1\}$	<b>V<sub>0</sub></b>
$\lambda Q \forall x \{Mx \rightarrow Qx\}$		$\lambda x \{x_1 \times x_{-1}\}$					$\lambda x_{-1} \{x_1\}$		<b>V<sub>1</sub></b>		
$\lambda Q_{1-1} \forall x \{Mx \rightarrow Qxx\}$				<b>H<sub>1</sub></b>		$\lambda P \lambda Q \{P \rightarrow Q\}$	$\lambda x_{-1} Vx$				
$\lambda Q_{-1} \forall x \{Mx \rightarrow Q[Hx,x]\}$						$\lambda x_{-1} \lambda Q \{Vx \rightarrow Q\}$					
$\forall x \{Mx \rightarrow \{Vx \rightarrow Hx\}\}$											

This derivation introduces another gnarly function –  $Q_{-1}$  – which is officially defined as follows.

$$Q_{-1} =_{df} \lambda(P \times x_{-1})Q\langle P,x \rangle$$

where P has type S, and Q has type  $(S \times D) \rightarrow S$ .

## 5. Example 5 (Counter-Example!)

Even adding powerful anaphoric-binding machinery to our semantic theory does not help us with a large number of examples, including the following seemingly innocuous example, which is nearly identical to the previous example.

no man is happy if he is evil

no	man	[+1]	[-1]	is	happy	if	(-1)	he	[+1]	is	evil	
$\lambda P_0 \lambda Q \forall x \{Px \rightarrow \sim Qx\}$	$\mathbf{M}_0$	$\lambda x \{x_1\}$	$\lambda x \{x_{-1}\}$	$\lambda P_0 \{P_1\}$	$\mathbf{H}_0$			$\lambda x_{-1} \{x\}$	$\emptyset$	$\lambda x \{x_1\}$	$\lambda P_0 \{P_1\}$	$\mathbf{E}_0$
$\lambda Q \forall x \{ \mathbf{M}x \rightarrow \sim Qx \}$		$\lambda x \{x_1 \times x_{-1}\}$						$\lambda x_{-1} \{x_1\}$		$\mathbf{E}_1$		
$\lambda Q_{1-1} \forall x \{ \mathbf{M}x \rightarrow \sim Qxx \}$				$\mathbf{H}_1$		$\lambda P \lambda Q \{P \rightarrow Q\}$		$\lambda x_{-1} \mathbf{E}x$				
$\lambda Q_{-1} \forall x \{ \mathbf{M}x \rightarrow \sim Q \langle \mathbf{H}x, x \rangle \}$						$\lambda x_{-1} \lambda Q \{ \mathbf{E}x \rightarrow Q \}$						
$\forall x \{ \mathbf{M}x \rightarrow \sim \{ \mathbf{E}x \rightarrow \mathbf{H}x \} \}$ $\equiv$ $\forall x \{ \mathbf{M}x \rightarrow \{ \mathbf{E}x \ \& \ \sim \mathbf{H}x \} \}$ $\equiv$ $\forall x \{ \mathbf{M}x \rightarrow \mathbf{E}x \} \ \& \ \forall x \{ \mathbf{M}x \rightarrow \sim \mathbf{H}x \}$												

According to this analysis, the sentence claims that every man is evil and unhappy. This may be a defensible theological position, but it is certainly not what the original sentence claims.

If this were an isolated example, we would not be too alarmed, and probably would be willing to devise a band-aid to cover this theoretical wound. Unfortunately, it is just the tip of the iceberg; there are lots of examples in which our current account of quantification and binding does not pass muster.

## 4. More Examples of Problematic Binding

### 1. Donkey Sentences

There is a class of sentences called "donkey sentences", so called because the original examples go something as follows.

- |  |                      |
|--|----------------------|
| (1) every man owns a donkey                | (no problem, so far) |
| (2) every man who owns a donkey is wealthy | (no problem, so far) |
| (3) every man who owns a donkey beats it   | (problem!)           |

The analyses go as follows.

1. every man owns a donkey

every	man	[+1]	owns	a	donkey	[+2]
$\lambda P_0 \lambda Q \forall x \{Px \rightarrow Qx\}$	$\mathbf{M}_0$	$\lambda x \{x_1\}$		$\lambda P_0 \lambda Q \exists x \{Px \ \& \ Qx\}$	$\mathbf{D}_0$	
$\lambda Q \forall x \{ \mathbf{M}x \rightarrow Qx \}$				$\lambda Q \exists x \{ \mathbf{D}x \ \& \ Qx \}$		$\lambda x \{x_2\}$
			$\lambda y_2 \lambda x_1 \mathbf{O}xy$	$\lambda Q_2 \exists x \{ \mathbf{D}x \ \& \ Qx \}$		
$\lambda Q_1 \forall x \{ \mathbf{M}x \rightarrow Qx \}$		$\lambda x_1 \exists y \{ \mathbf{D}y \ \& \ \mathbf{O}xy \}$				
$\forall x \{ \mathbf{M}x \rightarrow \exists y \{ \mathbf{D}y \ \& \ \mathbf{O}xy \} \}$						

2. every man who owns a donkey is wealthy

every	man	who	[+1]	owns	a	donkey	[+2]	[+1]	is	wealthy
$\lambda P_0 \lambda Q \forall x \{Px \rightarrow Qx\}$	$M_0$	$\lambda Q \lambda P_0 \lambda x_0 \{Px \& Qx\}$	$\lambda x \{x_1\}$	$\lambda y_2 \lambda x_1 Oxy$	$\lambda P_0 \lambda Q \exists x \{Px \& Qx\}$	$D_0$	$\lambda Q \exists x \{Dx \& Qx\}$	$\lambda x \{x_2\}$	$\lambda P_0 \{P_1\}$	$W_0$
					$\lambda Q_2 \exists x \{Dx \& Qx\}$					
		$\lambda Q_1 \lambda P_0 \lambda x_0 \{Px \& Qx\}$	$\lambda x_1 \exists y \{Dy \& Oxy\}$							
		$\lambda P_0 \lambda x_0 \{Px \& \exists y \{Dy \& Oxy\}\}$								
		$\lambda x_0 \{Mx \& \exists y \{Dy \& Oxy\}\}$								
		$\lambda Q \forall x \{ \{Mx \& \exists y \{Dy \& Oxy\}\} \rightarrow Qx \}$			$\lambda x \{x_1\}$					
		$\lambda Q_1 \forall x \{ \{Mx \& \exists y \{Dy \& Oxy\}\} \rightarrow Qx \}$			$W_1$					
$\forall x \{ \{Mx \& \exists y \{Dy \& Oxy\}\} \rightarrow Wx \}$										

3. every man who owns a donkey beats it

The following is the overall tree, which has several missing calculations.

every <sub>1</sub>	man	who <sub>1</sub>	owns	a donkey	[+2]	[-1]	beats	(-1)	it	[+2]
$\lambda P_0 \lambda Q_1 \forall x \{Px \rightarrow Qx\}$	$M_0$	$\lambda Q_1 \lambda P_0 \lambda x_0 \{Px \& Qx\}$	$\lambda y_2 \lambda x_1 Oxy$	$\lambda Q \exists x \{Dx \& Qx\}$	$\lambda x \{x_2\}$	$\lambda x \{x_{-1}\}$	$\lambda y_2 \lambda x_1 Bxy$	$\lambda x_{-1} \{x\}$	$\emptyset$	$\lambda x \{x_2\}$
					$\lambda x \{x_2 \times x_{-1}\}$			$\lambda x_{-1} \{x_2\}$		
					$\lambda Q_{2-1} \exists x \{Dx \& Qxx\}$					
					??					
					??					
					??			$\lambda y_{-1} \lambda x_1 Bxy$		
					??					
??										

This turns out to an exercise in futility (Sisyphus had it easy!) No completion of this tree will yield the correct semantic value. The closest we can get is the following.

$$\forall x \exists y \{ \{Mx \& \{Dy \& Oxy\}\} \rightarrow Bxy \}$$

This is not far off, at least superficially, since the correct translation is (equivalent to):

$$\forall x \forall y \{ \{Mx \& \{Dy \& Oxy\}\} \rightarrow Bxy \}$$

## 2. Other Problematic Examples

There are other examples of "donkey sentences", such as the following.

a donkey is happy if it is well-fed  
 if a donkey is well-fed, then it is happy  
 if it is well-fed, then a donkey is happy

but **not**:

it is happy if a donkey is well-fed

Except for the last one, in which pronoun-binding fails, these are mutually equivalent, which we also have to explain. For the moment, we concentrate on the first one. If we treat 'a' as a variant of 'some', as we did earlier, we have the following semantic analysis.

a	donkey	[+1]	[-1]	is	happy	if	(-1)	it	[+1]	is	well-fed	
$\lambda P_0 \lambda Q \exists x \{P_x \& Q_x\}$	<b>D<sub>0</sub></b>	$\lambda x \{x_1\}$	$\lambda x \{x_{-1}\}$	$\lambda P_0 \{P_1\}$	<b>H<sub>0</sub></b>			$\lambda x_{-1} \{x\}$	$\emptyset$	$\lambda x \{x_1\}$	$\lambda P_0 \{P_1\}$	<b>W<sub>0</sub></b>
$\lambda Q \exists x \{D_x \& Q_x\}$	$\lambda x \{x_1 \times x_{-1}\}$							$\lambda x_{-1} \{x_1\}$	<b>W<sub>1</sub></b>			
$\lambda Q_{1-1} \exists x \{D_x \& Q_x\}$		<b>H<sub>1</sub></b>			$\lambda P \lambda Q \{P \rightarrow Q\}$	$\lambda x_{-1} W_x$						
$\lambda Q_{-1} \exists x \{D_x \& Q(H_x, x)\}$						$\lambda x_{-1} \lambda Q \{W_x \rightarrow Q\}$						
$\exists x \{D_x \& \{W_x \rightarrow H_x\}\}$												

According to this reading, there is at least one donkey who is happy if it is well-fed. This is not the most salient reading; indeed, in some sense, it is absurd, at least for the truth-functional reading of ' $\rightarrow$ '. For then, the final formula is equivalent to the following.

$$\exists x \{D_x \& \sim W_x\} \vee \exists x \{D_x \& H_x\}$$

This means that, according to the proposed account, the original sentence is true if at least one donkey is not well-fed, and also if at least one donkey is happy. This is not what the original sentence says.

The most natural reading of the sentence treats 'a donkey' as a wide-scope universal-quantifier, which gives the following top node.

$$\forall x \{D_x \rightarrow \{W_x \rightarrow H_x\}\}$$

Similarly, we can treat 'a donkey' in the original donkey-sentence as a wide-scope universal-quantifier, as follows.

$$\forall x \{D_x \rightarrow \forall y \{ \{M_y \& Oyx\} \rightarrow Byx \} \}$$

These are the answers I expect my elementary logic students to produce on exams, and the good students have no problem.

But the semanticist has a problem, a very vexing problem – to provide a theoretical account of 'a donkey' that systematically explains how one computationally arrives at these top nodes. Competent speakers "know" what the top nodes are; competent logic students know how to translate the top nodes into loglish; the question is – how do we get there computationally from the pieces we start with?