Anaphora resolved

Roelofsen, F.

Publication date
2008

Chapter 1

Framework

This part of the dissertation evaluates and refines some of the most prominent theories of pronominal anaphora that have been developed within the framework of Generative Grammar. These theories are particularly concerned with third person singular pronouns such as *he* and *she*, which I will henceforth simply refer to as *pronouns*. The most important common characteristic of the theories to be discussed is that they all assume a fundamental distinction between *bound* and *referential* pronouns. The remainder of this introductory chapter is dedicated to motivating this distinction, and defining the formal syntax and semantics of a fragment of English in which bound and referential pronouns are clearly distinguished. Chapter 2 will discuss several accounts of how binding and coreference are constrained, and chapter 3 will attempt to resolve the issues that are raised and/or left open by these accounts.

1.1 Bound and Referential Pronouns

Pronouns can be interpreted in at least two distinct ways.[1] First, they can be interpreted as *bound variables*. For example, sentence (1.1) below has a reading which says that every man has the property of being an x such that x thinks that x will win. Or in slightly more formal terms, that every man has the property [λx. x thinks that x will win]. On this reading, *he* is interpreted as a variable x which is bound by a λ-operator.

(1.1) Every man thinks he will win.

Second, pronouns can be taken to *refer* to some contextually salient entity. In (1.2) for example, *he* may be taken to refer to John.

(1.2) John is in good shape. I think he will win.

[1] For an early discussion of this distinction, see Partee (1978).

Further motivation for the distinction between bound and referential pronouns comes from the fact that it naturally explains certain ambiguities that arise when pronouns occur in focus constructions and in elliptical constructions.

Ambiguity in focus constructions. Consider the following sentence (capital letters are used here to indicate a pitch accent):

(1.3) MAX called his mother.

Suppose that the pronoun is anaphorically related to *Max*. Then the sentence has two readings. The first says that Max called his mother, and suggests that nobody else called Max's mother. That is, Max has the property [$\lambda x.$ x called Max's mother], and nobody else does. The second reading says that Max called his mother and suggests that other people didn't call their mother. That is, Max has the property [$\lambda x.$ x called x's mother] and the other people don't. This ambiguity is naturally explained in terms of the distinction between bound and referential pronouns. Interpreting *his* as referring to a contextually salient individual, in this case Max, yields the first reading, while interpreting the pronoun as a bound variable gives us the second reading.[2]

A similar ambiguity arises in constructions which involve focus-sensitive operators such as *only* and *even*. Consider the following example:

(1.4) Only MAX called his mother.

Suppose that the pronoun is anaphorically related to *Max*. Then the sentence has two readings. The first says that only Max has the property [$\lambda x.$ x called x's mother] (nobody else called his own mother); the second reading says that only Max has the property [$\lambda x.$ x called Max's mother] (nobody else called Max's mother). The distinction between bound and referential pronouns provides a natural explanation of this ambiguity. On the first reading, *his* is interpreted as a bound variable; on the second reading, it is interpreted as referring to Max. Of course, similar examples can be constructed with other focus-sensitive operators.

Ambiguity in elliptical constructions. Consider (1.5), a simple case of VP ellipsis.

(1.5) Max called his mother and Bob did too.

a. ...Bob called his own mother too. [sloppy]
b. ...Bob called Max's mother too. [strict]

[2]Even more readings are obtained, of course, if the pronoun in (1.3) is not taken to refer to Max but to some other contextually salient individual. Such readings are left out of consideration here and in the examples below, because they are not really relevant for the point being made.

Suppose that the pronoun in the source clause (*Max called his mother*) is anaphorically related to *Max*. Then, as first observed by Ross (1967), the target clause (*Bob did too*) has two readings: (1.5a) and (1.5b). The first reading is called *sloppy*; the second is called *strict*.

Keenan (1971) first suggested that this ambiguity can be explained in terms of the distinction between bound and referential pronouns. If the pronoun in the source clause is interpreted as a bound variable, then the source clause as a whole says that Max has the property [$\lambda x.$ x called x's mother], and the target clause says that Bill has that property too. This gives us the sloppy reading in (1.5a). If the pronoun is taken to refer to the most salient individual in the utterance context—here, plausibly Max—then the source clause says that Max has the property [$\lambda x.$ x called Max's mother], and the target clause, again, says that Bill has that property too. This gives us the strict reading in (1.5b).

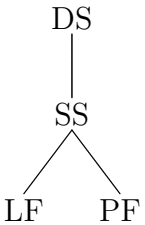
This concludes the informal characterization of and motivation for the distinction between bound and referential pronouns. In the next section, I will formally define the syntax and the semantics of a fragment of English in which bound and referential pronouns are clearly distinguished.

1.2 Basic Framework

The fragment to be defined here will include most of the example sentences to be discussed. To keep the framework as simple as possible, the syntax will be allowed to overgenerate considerably. I will not discuss any syntactic constraints that could be deployed to combat this overgeneration. My aim here is merely to set up a precise and convenient terminology, so that the discussion below will be clear and my claims falsifiable.

1.2.1 Syntax

To facilitate the discussion below, I will assume the old Government and Binding architecture (Chomsky, 1981), in which there are four levels of syntactic representation: Deep Structure (DS), Surface Structure (SS), Logical Form (LF), and Phonological Form (PF):



(PS 1)	S	→	NP VP	[sentences]
(PS 2)	CP	→	C S	[complement phrases]
(PS 3)	VP	→	IV	[verb phrases]
(PS 4)	VP	→	TV NP	[verb phrases]
(PS 5)	VP	→	AV CP	[verb phrases]
(PS 6)	NP	→	POS RN	[noun phrases]
(PS 7)	NP	→	DET CN	[noun phrases]
(PS 8)	CN	→	CN S	[common nouns]
(PS 9)	POS	→	NP 's	[possessives]
(LI 1)	DET	→	a, the, every, some, no, ...	[determiners]
(LI 2)	CN	→	man, girl, ...	[common nouns]
(LI 3)	RN	→	mother, friend, ...	[relational nouns]
(LI 4)	IV	→	sing, walk, ...	[intransitive verbs]
(LI 5)	TV	→	call, love ...	[transitive verbs]
(LI 6)	AV	→	know, say ...	[attitude verbs]
(LI 7)	C	→	that	[complementizer]
(LI 8)	NP	→	John, Mary, Max, Lucie, ...	[noun phrases]
			who, whom,	
			he_n, she_n, it_n, ...	
			he, she, it, ...	

Table 1.1: Phrase structure rules and lexical insertion rules for the generation of deep structures.

The DS component of our fragment consists of all trees (or labeled bracketings) that can be generated by the phrase structure rules (PS 1-9) and the lexical insertion rules (LI 1-8) in table 1.1.

Bound pronouns come with a binding index, which is adjoined to the pronoun in subscript (e.g., [she_2]). Referential pronouns do not have an index. The general form of possessives is [NP 's]. This generates instances such as [John 's], [every girl 's], and [he_1 's]. I will often write [his_1] instead of [he_1 's], and similarly for other pronominal possessives. Also, I will often simply refer to such pronominal possessives as pronouns, as I already did in the informal discussion above.

I will assume that surface structures are obtained from deep structures by wh-movement, and that logical forms are obtained from surface structures by quantifier raising. If a wh-element moves it receives a binder index n, which is adjoined to it in superscript (e.g., [who]^3). It also leaves behind a trace which has that same index n as its binding index (e.g., the trace of [who]^3 would be t_3).

(1.6) $[_S\ X\ [_{NP}\ wh]\ Y] \Rightarrow [_S\ [_{NP}\ wh]^n\ [_S\ X\ t_n\ Y]]$ (wh-movement)

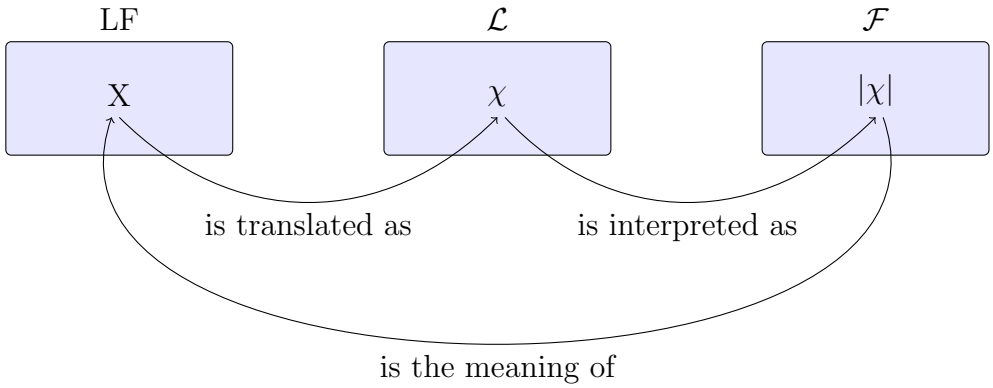
The same goes for quantifier raising: if a noun phrase undergoes QR it receives a binder index n and leaves behind a trace which has that same index n as its binding index.

(1.7) $[_{\text{S}}\ \text{X}\ [_{\text{NP}}\ \text{Z}]\ \text{Y}] \quad \Rightarrow \quad [_{\text{S}}\ [_{\text{NP}}\ \text{Z}]^n\ [_{\text{S}}\ \text{X}\ \text{t}_n\ \text{Y}]]$ (quantifier raising)

Finally, phonological forms are obtained from surface structures by contracting pronominal possessives (e.g. [he_1 's] becomes [his_1]) and deleting all indices, brackets, and traces.

1.2.2 Semantics

The semantic component of our framework associates logical form constituents with their meaning. A standard way of doing so consists in the following three steps. First, a space of possible meanings is defined. Such a space of meanings is called a *frame* $\mathcal{F}$. Second, a formal language $\mathcal{L}$ is defined, and each expression in $\mathcal{L}$ is assigned a meaning in $\mathcal{F}$. Finally, logical form constituents are translated into $\mathcal{L}$-expressions. This is pictured below. Each logical form constituent X is translated into an $\mathcal{L}$-expression χ, which in turn is assigned a meaning $|\chi|$ in $\mathcal{F}$. $|\chi|$ is then called the meaning of X.



I will take $\mathcal{F}$ and $\mathcal{L}$ to be a frame and a language of two-sorted type theory (Ty2) (Gallin, 1975).[3] Below, I will first define Ty2 in general, and then specify the particular Ty2 frame $\mathcal{F}$ and the particular Ty2 language $\mathcal{L}$ that we will use.

Two-sorted Type Theory

We start with the basis: a definition of the *types* in two-sorted type theory. In n-sorted type theory there are $n + 1$ basic types and infinitely many complex types. Thus, in the particular case of 2-sorted type theory there are 3 basic types and infinitely many complex types.

[3] In general, $\mathcal{F}$ and $\mathcal{L}$ are taken to be a frame and a language of n-sorted type theory, where n depends on the complexity of the fragment of natural language that is being described.

1.1. Definition. [Types]
The set Ω of Ty2 types is the smallest set of strings such that:

1. $e, s, t \in \Omega$
2. If $\tau, \sigma \in \Omega$, then $(\tau\sigma) \in \Omega$

Outer brackets of complex types will often be omitted. For example, $(s(et))$ will often be abbreviated as $s(et)$.

Given Ω, we can define the class of Ty2 frames and the class of Ty2 languages.

1.2. Definition. [Frames]
A Ty2 frame F is a set of objects $\bigcup_{\tau\in\Omega} D_\tau^F$ such that:

- $D_e^F \neq \emptyset$
- $D_s^F \neq \emptyset$
- $D_t^F = \{0, 1\}$
- $D_{\tau\sigma}^F = \{f \mid f : D_\tau \to D_\sigma\}$ for every complex type $\tau\sigma$

Note that the letter F *ranges over* Ty2 frames here. In particular, it should not be confused with the letter $\mathcal{F}$, which denotes the particular frame whose elements will be associated with the logical form constituents in our fragment of English (this particular frame will be defined below).

For every Ty2 frame F and every Ty2 type τ, D_τ^F is the set of objects of type τ in F. Table 1.2 lists some names that are customarily used for objects of certain types in Ty2 frames.

Objects of type	are called
t	truth values
s	possible worlds
e	individuals
et	properties
$e(et)$	binary relations
se	individual concepts
$s(et)$	property concepts
$s(e(et))$	binary relation concepts
st	propositions

Table 1.2: Names for objects of certain types.

1.3. Definition. [Languages]
A Ty2 language L is a set of expressions $\bigcup_{\tau\in\Omega} E_\tau^L$ such that:

- For every Ty2 type τ, E_τ^L contains a countable set of constants of type τ and a countable set of variables of type τ.

- If φ and ψ are expressions of type t (*formulas*) then $\neg\varphi$ and $(\varphi \wedge \psi)$ are also formulas;
- If φ and ψ are expressions of the same type, then $\varphi = \psi$ is a formula;
- If φ is a formula and x is a variable of any type, then $\forall x.\varphi$ is a formula;
- If φ is a formula and x is a variable of type e, then $\iota x.\varphi$ is an expression of type e.
- If φ is an expression of type σ and x is a variable of type τ, then $\lambda x.\varphi$ is an expression of type $\tau\sigma$;
- If φ is an expression of type $(\tau\sigma)$ and ψ is an expression of type τ, then $\varphi(\psi)$ is an expression of type σ;

Other logical operators ($\exists$, $\vee$, $\rightarrow$, $\leftrightarrow$) are used as abbreviations:

- $\exists x.\varphi$ abbreviates $\neg\forall x.\neg\varphi$
- $\varphi \vee \psi$ abbreviates $\neg(\neg\varphi \wedge \neg\psi)$
- $\varphi \rightarrow \psi$ abbreviates $\neg(\varphi \wedge \neg\psi)$
- $\varphi \leftrightarrow \psi$ abbreviates $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$

Expressions are sometimes subscripted with their type. For example, we may write φ_t to indicate that φ is of type t. Finally, note the difference between the letters L and $\mathcal{L}$. L is used here to range over TY2 languages, whereas $\mathcal{L}$ denotes the particular TY2 language whose elements will be associated with the logical form constituents in our fragment of English ($\mathcal{L}$ will be defined below).

Thus, we have defined what TY2 frames and TY2 languages are. Now, given a certain TY2 language L and a certain TY2 frame F, we must specify how the expressions in L are assigned a meaning in F. This is done by means of *interpretation functions* and an *assignment functions*.

1.4. DEFINITION. [Interpretation functions and assignment functions]
Let L be a TY2 language and F a TY2 frame. Then, an *interpretation function* I for L and F is a function that maps every *constant* in L to an object in F, such that for every type τ and every constant $c_\tau \in E^L_\tau$ we have $I(c_\tau) \in D^F_\tau$. That is, I maps every constant of type τ to an object of type τ. Similarly, an *assignment function* g for L and F maps every *variable* of type τ to an object of type τ. If g is an assignment function, we write $g[d/x]$ for the assignment function g' defined by $g'(x) = d$ and $g'(y) = g(y)$ if $y \neq x$.

1.5. DEFINITION. [Interpretation]
Let L be a TY2 language, F a TY2 frame, I an interpretation function for L and F, g an assignment function for L and F, and φ an expression in L. Then the

interpretation $|\varphi|^{F,I,g}$ of φ in F given I and g is recursively defined as follows:[4]

$$
\begin{array}{lcl}
|c| & = & I(c) \text{ if } c \text{ is a constant} \\
|x| & = & g(x) \text{ if } x \text{ is a variable} \\
|\neg\varphi| & = & 1 \text{ iff } |\varphi| = 0 \\
|\varphi \wedge \psi| & = & 1 \text{ iff } |\varphi| = 1 \text{ and } |\psi| = 1 \\
|\varphi = \psi| & = & 1 \text{ iff } |\varphi| = |\psi| \\
|\forall x_\tau.\varphi|^{F,I,g} & = & 1 \text{ iff } |\varphi|^{F,I,g[d/x]} = 1 \text{ for all } d \in D_\tau^F \\
|\iota x_e.\varphi|^{F,I,g} & = & \begin{cases} \text{the unique object } d \in D_e^F \text{ such that } |\varphi|^{F,I,g[d/x]} = 1 \\ \text{undefined if such a unique object does not exist} \end{cases} \\
|\lambda x_\tau.\varphi|^{F,I,g} & = & \text{the function } f \text{ with domain } D_\tau^F \text{ such that} \\
 & & \text{for all } d \in D_\tau^F\text{: } f(d) = |\varphi|^{F,I,g[d/x]} \\
|\varphi(\psi)| & = & |\varphi|(|\psi|)
\end{array}
$$

Notice that the interpretation of expressions of the form $\iota x_e.\varphi$ may be undefined. To keep things simple, I have not specified how this may affect the definedness of more complex expressions which contain expressions of this kind as a subexpression. This problem is a particular instance of a more general problem, which is known as the problem of *presupposition projection*. This is an important problem in itself, but I will not go into it here. The reader is referred to (Beaver, 1997; Geurts, 1999) and the references given there.

Next, we define what it means for two Ty2 expressions to be *equivalent*.

1.6. Definition. [Equivalence]
Let L be a Ty2 language and let φ and ψ be expressions in L. Then:

- φ and ψ are equivalent iff $|\varphi|^{F,I,g} = |\psi|^{F,I,g}$ for all F, I, and g.[5]
- φ and ψ are equivalent given a particular frame F' iff $|\varphi|^{F',I,g} = |\psi|^{F',I,g}$ for all I, and g.
- φ and ψ are equivalent given a particular frame F' and a particular interpretation function I' iff $|\varphi|^{F',I',g} = |\psi|^{F',I',g}$ for all g.

We may also define what it means for one Ty2 expression to *entail* another. We are especially interested in entailment between expressions of type st, because these are the expressions that will be associated with *sentential* logical form constituents.

1.7. Definition. [Entailment]
Let L be a Ty2 language and let φ and ψ be expressions of type (st) in L. Then:

[4] Whenever possible, I simply write $|\varphi|$ instead of $|\varphi|^{F,I,g}$.

[5] Provided, of course, that I is an interpretation function for L and F, and g is an assignment function for L and F. Henceforth, this qualification will be left implicit.

- φ entails ψ iff
  for all F, I, and g, and for all $w \in D_s^F$ such that
  $|\varphi|^{F,I,g}(w) = 1$ we also have $|\psi|^{F,I,g}(w) = 1$

- φ entails ψ given a particular frame F' iff
  for all I and g, and for all $w \in D_s^{F'}$ such that
  $|\varphi|^{F',I,g}(w) = 1$ we also have $|\psi|^{F',I,g}(w) = 1$

- φ entails ψ given a particular frame F'
  and a particular interpretation function I' iff
  for all g and for all $w \in D_s^{F'}$ such that
  $|\varphi|^{F',I',g}(w) = 1$ we also have $|\psi|^{F',I',g}(w) = 1$

Finally, it should be remarked that TY2 expressions can be converted into other TY2 expressions by α-conversion and β-reduction. α-conversion can be thought of as re-naming of bound variables and β-reduction as applying λ-expressions to their arguments. For example:

$(\lambda x.x = y)$	can be α-converted into	$(\lambda z.z = y)$
$(\lambda x.x = y)(z)$	can be β-reduced to	$(z = y)$

If ψ can be obtained from φ by (repeatedly) applying α-conversion and/or β-reduction, then we will simply say that φ can be *reduced* to ψ. If φ can be reduced to ψ, then φ and ψ are always equivalent (for a proof of this fact, as well as proper definitions of α-conversion and β-reduction see Andrews, 1986). This means that the picture we started out with in the beginning of this section is in fact a little bit more complicated. Each logical form constituent X is translated into an $\mathcal{L}$-expression χ. This expression may be *reducible* to other $\mathcal{L}$-expressions $\chi', \chi'', \ldots$. In any case, $\chi, \chi', \chi'', \ldots$ will be equivalent, that is, they will all be associated with the same meaning $|\chi|$. Thus, $|\chi|$ will be called *the* meaning of X, and $\chi, \chi', \chi'', \ldots$ will all be called possible translations of X.

This concludes my presentation of TY2. For more detail, I refer to Gallin (1975) and Andrews (1986).

Fixing $\mathcal{F}$, $\mathcal{L}$, and $\mathcal{I}$

Let me now specify $\mathcal{F}$ and $\mathcal{L}$, the particular TY2 frame and TY2 language whose elements will be associated with the LF constituents in our fragment of English. We will take $\mathcal{F}$ to be the most general frame, containing all possible meanings. This means, in particular, that $D_e^{\mathcal{F}}$ will consist of all possible individuals and that $D_s^{\mathcal{F}}$ will consist of all possible worlds. Next let us define $\mathcal{L}$. To do so we must fix its inventory of constants and its inventory of variables. The constants in $\mathcal{L}$ correspond to the content words in our fragment of English.[6] Some of the

[6]There is a traditional distinction between content words and function words. Names, nouns, verbs, adjectives, and most adverbs are considered to be content words, while determiners,

constants in $\mathcal{L}$ are listed in table 1.3, and some of the variables in $\mathcal{L}$ are listed in table 1.4. Notice that $\mathcal{L}$ contains two kinds of variables ranging over individuals: $x_1, x_2, \ldots$ will be used in the translation of traces, whereas $x, x', \ldots$ will be used for all other purposes (see, for example, the translation of *every* in table 1.5 below).

JOHN	se	individual concept
SING	$s(et)$	property concept
MAN	$s(et)$	property concept
MOTHER	$s(e(et))$	binary relation concept
LOVE	$s(e(et))$	binary relation concept
SAY	$s((st)(et))$	

Table 1.3: Some constants in $\mathcal{L}$.

$w, w', \ldots$	s	worlds
$x, x', \ldots$	e	individuals
$x_1, x_2, \ldots$	e	individuals
$P, P', \ldots$	et	properties
$R, R', \ldots$	$e(et)$	binary relations
$p, p', \ldots$	st	propositions

Table 1.4: Some variables in $\mathcal{L}$.

Apart from $\mathcal{F}$ and $\mathcal{L}$, we will also fix the interpretation function $\mathcal{I}$ that maps all the constants in $\mathcal{L}$ onto appropriate meanings in $\mathcal{F}$. $\mathcal{I}$ is taken to be the most general interpretation function that respects the way in which different content words are conventionally related. For example, $\mathcal{I}$ must be such that for every world w in $D_s^{\mathcal{F}}$, $\mathcal{I}(\text{TIGER})(w)$ (the set of tigers in w) is a subset of $\mathcal{I}(\text{ANIMAL})(w)$ (the set of animals in w).

From Logical Form Constituents to TY2 Expressions

Now we are ready to specify how logical form constituents are translated into $\mathcal{L}$-expressions. This is done in two steps. First, the translation of terminal nodes is defined and second, the translation of non-terminal nodes is defined in terms of the translations of their daughter nodes. The translation function $[\![\ ]\!]^{\text{C}}$ will have a context-parameter C, which reflects the idea that the interpretation of some words, in particular referential pronouns, depends on the context of use.

pronouns, complementizers, auxiliaries, expletives, etc. are considered to be function words. The only content words in our fragment are names, nouns, and verbs.

Node		Translation	Type
$[\![\text{man}]\!]^{\text{C}}$	=	$\lambda w.\lambda x.\text{MAN}(w)(x)$	$s(et)$
$[\![\text{sing}]\!]^{\text{C}}$	=	$\lambda w.\lambda x.\text{SING}(w)(x)$	$s(et)$
$[\![\text{mother}]\!]^{\text{C}}$	=	$\lambda w.\lambda x.\lambda y.\text{MOTHER}(w)(x)(y)$	$s(e(et))$
$[\![\text{love}]\!]^{\text{C}}$	=	$\lambda w.\lambda x.\lambda y.\text{LOVE}(w)(x)(y)$	$s(e(et))$
$[\![\text{say}]\!]^{\text{C}}$	=	$\lambda w.\lambda p.\lambda x.\text{SAY}(w)(p)(x)$	$s((st)(et))$
$[\![\text{John}]\!]^{\text{C}}$	=	$\lambda w.\text{JOHN}(w)$	se
$[\![\text{he}]\!]^{\text{C}}$	=	$[\![\text{ANT}^{\text{C}}(\text{he})]\!]^{\text{C}}$	se
$[\![\text{he}_n]\!]^{\text{C}}$	=	$\lambda w.x_n$	se
$[\![\text{t}_n]\!]^{\text{C}}$	=	$\lambda w.x_n$	se
$[\![\text{that}]\!]^{\text{C}}$	=	$\lambda w.\lambda p.p(w)$	$s((st)t)$
$[\![\text{who}]\!]^{\text{C}}$	=	$\lambda w.\lambda P.P$	$s((et)(et))$
$[\![\text{every}]\!]^{\text{C}}$	=	$\lambda w.\lambda P.\lambda P'.\forall x(P(x) \rightarrow P'(x))$	$s((et)((et)t))$
$[\![\text{the}]\!]^{\text{C}}$	=	$\lambda w.\lambda P.\iota x.P(x)$	$s((et)e)$
$[\![\text{'s}]\!]^{\text{C}}$	=	$\lambda w.\lambda x.\lambda R.\iota x'.R(x)(x')$	$s(e((e(et))e))$

Table 1.5: Translations of some terminal LF nodes.

The Translation of Terminal Nodes. Table 1.5 specifies the translation of some terminal LF nodes. Other terminal LF nodes are translated analogously.

A referential pronoun [he] occurring in a context C is translated as $[\![\text{ANT}^{\text{C}}(\text{he})]\!]^{\text{C}}$ where $\text{ANT}^{\text{C}}(\text{he})$ is the *antecedent* of [he] in C. The antecedent of a referential pronoun must always be a referential expression itself: an expression of type (se) whose translation does not contain any free variables (traces, in particular, do *not* count as referential expressions). If a referential expression A is the antecedent of a pronoun P in a context C, then I will say that P is *resolved* to A in C and write P = A next to the LF under consideration. For example, if (1.8) is considered in a context in which [she] is resolved to [Mary] I will write she = Mary next to it, as in (1.9).

(1.8) [Mary] [says that she likes John]

(1.9) [Mary] [says that she likes John] she = Mary

Notice that apart from the translation of referential pronouns, all the other translations in table 1.5 are context-independent. This is a simplification, which I permit myself here in order to focus exclusively on the interpretation of pronouns. In general, the translation of other nodes may also be context-dependent (I am thinking, for example, of the domain restrictions of determiners).

Finally, notice that every terminal LF node X is translated into a type-theoretical expression χ of type $s\tau$ (where τ may be different in each case). This means that X is always associated with a function $|\chi|$ from possible worlds to objects of type τ. Such objects (functions from possible worlds to other objects) are called *intensional objects*. Accordingly, $|\chi|$ is called the *intension* of X. For every

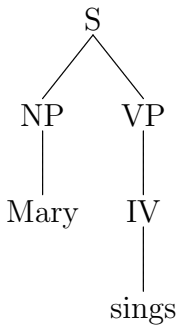
particular world w, $|\chi|(w)$ will be an object of type τ. This object is called the *extension* or the *denotation* of X in w. We will say that X *expresses* its intension $|\chi|$ and that X *denotes* its extension $|\chi|(w)$ in each particular world w.

For example, [John] expresses the individual concept $|\lambda w.\textsc{john}(w)|$ and in each particular world w', it denotes the individual $|\lambda w.\textsc{john}(w)|(w')$. An intransitive verb like [sing] expresses the property concept $|\lambda w.\lambda x.\textsc{sing}(w)(x)|$, and in each particular world w' it denotes the property $|\lambda w.\lambda x.\textsc{sing}(w)(x)|(w')$. This terminology extends in a natural way to the other terminal nodes and, as we will see right below, also to all non-terminal nodes.

The Translation of Non-Terminal Nodes. The composition rules in table 1.6 specify how the translation of a non-terminal LF node can be constructed from the translations of its daughter nodes. Notice that the composition rules assign to every non-terminal node X a translation χ of type $s\tau$. Thus, the meaning associated with a non-terminal node is always an *intensional* object. As in the case of terminal nodes, $|\chi|$ is called the intension of X, and in every particular world w, $|\chi|(w)$ is called the denotation or the extension of X in w.

Let me go through a few examples to illustrate how the composition rules work. First, consider the logical form in (1.10).

(1.10) $[_{\text{S}}[_{\text{NP}}$ Mary$]$ $[_{\text{VP}}[_{\text{IV}}$ sings$]]]$



This example illustrates the workings of COPY and EFA (extensional function application). First, COPY tells us that the translation of $[_{\text{VP}}[_{\text{IV}}$ sings$]]$ is identical to the translation of $[_{\text{IV}}$ sings$]$, which is defined in the lexicon:

(1.11) $\lambda w.\lambda x.\textsc{sing}(w)(x)$

The translation of $[_{\text{NP}}$ Mary$]$ is also defined in the lexicon:

(1.12) $\lambda w.\textsc{mary}(w)$

Now EFA tells us that the translation of $[_{\text{S}}$ Mary sings$]$ is:

COPY
If a non-terminal node X only has one daughter node Y then:

$$[\![\mathrm{X}]\!]^{\mathrm{C}} = [\![\mathrm{Y}]\!]^{\mathrm{C}}$$

EFA (extensional function application)
If a non-terminal node X has two daughters Y and Z such that $[\![\mathrm{Y}]\!]^{\mathrm{C}} = \gamma$ and $[\![\mathrm{Z}]\!]^{\mathrm{C}} = \zeta$ with γ of type $s(\tau\sigma)$ and ζ of type $s\tau$ for some τ and σ, then:

$$[\![\mathrm{X}]\!]^{\mathrm{C}} = \lambda w.\gamma(w)(\zeta(w))$$

IFA (intensional function application)
If a non-terminal node X has two daughters Y and Z such that $[\![\mathrm{Y}]\!]^{\mathrm{C}} = \gamma$ and $[\![\mathrm{Z}]\!]^{\mathrm{C}} = \zeta$ with γ of type $s((s\tau)\sigma)$ and ζ of type $s\tau$ for some τ and σ, then:

$$[\![\mathrm{X}]\!]^{\mathrm{C}} = \lambda w.\gamma(w)(\zeta)$$

QINP (quantifying in noun phrases of type se)
If a non-terminal node X has two daughters Y^{n} (notice the binder index) and Z such that $[\![\mathrm{Y}]\!]^{\mathrm{C}} = \gamma$ and $[\![\mathrm{Z}]\!]^{\mathrm{C}} = \zeta$ with γ of type se and ζ of type st, then:

$$[\![\mathrm{X}]\!]^{\mathrm{C}} = \lambda w.(\lambda x_n.\zeta(w))(\gamma(w))$$

QIGQ (quantifying in generalized quantifiers of type $s((et)t)$)
If a non-terminal node X has two daughters Y^{n} (notice the binder index) and Z such that $[\![\mathrm{Y}]\!]^{\mathrm{C}} = \gamma$ and $[\![\mathrm{Z}]\!]^{\mathrm{C}} = \zeta$ with γ of type $s((et)t)$ and ζ of type st, then:

$$[\![\mathrm{X}]\!]^{\mathrm{C}} = \lambda w.(\gamma(w))(\lambda x_n.\zeta(w))$$

QIWH (quantifying in wh-elements of type $s((et)(et))$)
If a non-terminal node X has two daughters Y^{n} (notice the binder index) and Z such that $[\![\mathrm{Y}]\!]^{\mathrm{C}} = \gamma$ and $[\![\mathrm{Z}]\!]^{\mathrm{C}} = \zeta$ with γ of type $s((et)(et))$ and ζ of type st, then:

$$[\![\mathrm{X}]\!]^{\mathrm{C}} = \lambda w.(\gamma(w))(\lambda x_n.\zeta(w))$$

PM (predicate modification)
If a non-terminal node X has two daughters Y and Z such that $[\![\mathrm{Y}]\!]^{\mathrm{C}} = \gamma$ and $[\![\mathrm{Z}]\!]^{\mathrm{C}} = \zeta$ with γ and ζ both of type $s(et)$, then:

$$[\![\mathrm{X}]\!]^{\mathrm{C}} = \lambda w.\lambda x.\gamma(w)(x) \wedge \zeta(w)(x)$$

FC (function composition)
If a non-terminal node X has two daughters Y and Z such that $[\![\mathrm{Y}]\!]^{\mathrm{C}} = \gamma$ and $[\![\mathrm{Z}]\!]^{\mathrm{C}} = \zeta$ with γ of type $s(\tau\sigma)$ and ζ of type $s(\sigma\rho)$ for some τ, σ and ρ, then:

$$[\![\mathrm{X}]\!]^{\mathrm{C}} = \lambda w.\lambda y_{\tau}.(\zeta(w))(\gamma(w)(y))$$

Table 1.6: Rules which determine the translation of non-terminal LF nodes.

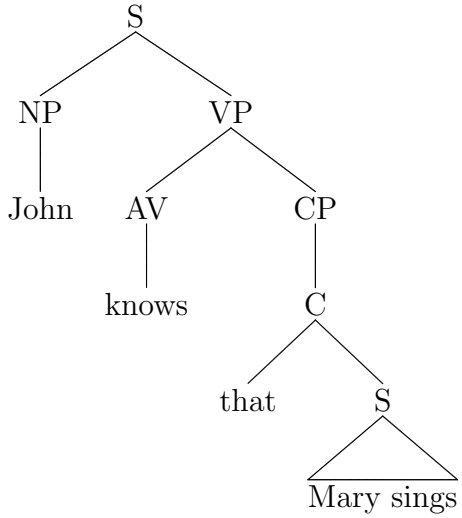
(1.13) $\lambda w.(\overbrace{\boxed{\lambda w'.\lambda x.\textsc{sing}(w')(x)}}^{\text{[sings]}})(w)\left((\overbrace{\boxed{\lambda w''.\textsc{mary}(w'')}}^{\text{[Mary]}})(w)\right)$

which can be reduced to:

(1.14) $\lambda w.(\textsc{sing}(w)(\textsc{mary}(w)))$

The next example, (1.15), illustrates how IFA (intensional function application) works. It also shows how the complementizer *that* and CP-embedding verbs like *know* are treated.

(1.15) [$_{\text{S}}$[$_{\text{NP}}$ John] [$_{\text{VP}}$[$_{\text{AV}}$ knows] [$_{\text{CP}}$[$_{\text{C}}$ that] [$_{\text{S}}$ Mary sings]]]]



Let us first determine the translation of the embedded CP. Notice that the translation of [$_{\text{S}}$ Mary sings] was derived above. The translation of [$_{\text{C}}$ that] can be found in the lexicon:

(1.16) $\lambda w.\lambda p.p(w)$

Now IFA tells us how to combine the translations of [$_{\text{C}}$ that] and [$_{\text{S}}$ Mary sings] to get the translation of [$_{\text{CP}}$ that Mary sings]:

(1.17) $\lambda w.(\overbrace{\boxed{\lambda w'.\lambda p.p(w')}}^{\text{[that]}})(w)(\overbrace{\boxed{\lambda w''.(\textsc{sing}(w'')(\textsc{mary}(w'')))}}^{\text{[Mary sings]}})$

which can be reduced to:

(1.18) $\lambda w.(\textsc{sing}(w)(\textsc{mary}(w)))$

Notice that this is identical to the translation of [$_{\text{S}}$ Mary sings]. So the complementizer [$_{\text{C}}$ that] has no semantic effect. Now let us determine the translation of the matrix clause. The translation of [$_{\text{PV}}$ knows] can be found in the lexicon:

(1.19) $\lambda w.\lambda p.\lambda x.\textsc{knows}(w)(p)(x)$

IFA tells how to combine this with the translation of the embedded clause to get the translation of [$_{\text{VP}}$ knows that Mary sings]:

(1.20) $\lambda w.(\overbrace{\boxed{\lambda w'.\lambda p.\lambda x.\textsc{knows}(w')(p)(x)}}^{\text{[knows]}})(w)(\overbrace{\boxed{\lambda w''.(\textsc{sing}(w'')(\textsc{mary}(w'')))}}^{\text{[that Mary sings]}})$

which can be reduced to:

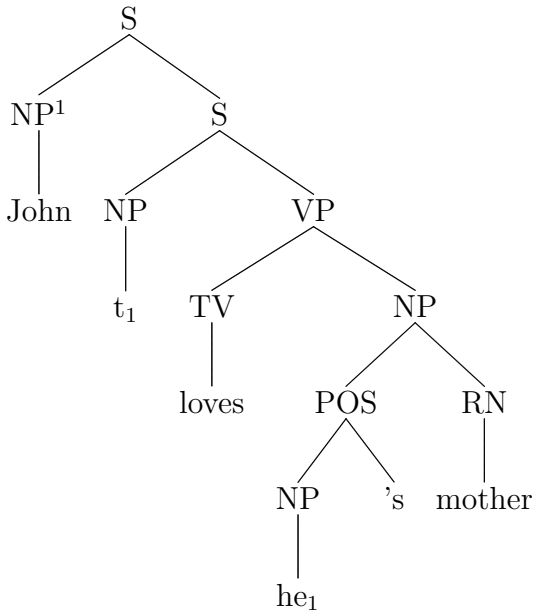
(1.21) $\lambda w.\lambda x.\textsc{knows}(w)(\lambda w'.\textsc{sing}(w')(\textsc{mary}(w')))(x)$

Finally, this is combined with the translation of [$_{\text{NP}}$ John], which can be found in the lexicon, to get the translation of (1.15):

(1.22) $\lambda w.\textsc{knows}(w)(\lambda w'.\textsc{sing}(w')(\textsc{mary}(w')))(\textsc{john}(w))$

The next example, (1.23), illustrates how a noun phrase of type *se* is "quantified in" with the help of QINP. It also shows how possessives like [$_{\text{POS}}$ he$_1$'s] and relational nouns like [$_{\text{RN}}$ mother] are treated.

(1.23) [$_{\text{S}}$ [$_{\text{NP}}$ John]^1 [$_{\text{S}}$ [$_{\text{NP}}$ t$_1$] [$_{\text{VP}}$ [$_{\text{TV}}$ loves] [$_{\text{NP}}$ [$_{\text{POS}}$ he$_1$'s]] [$_{\text{RN}}$ mother]]]]]



Let us first derive the translation of [$_{\text{POS}}$ he$_1$'s]. The translation of its elements can be found in the lexicon and are composed using EFA to get:

(1.24) $\lambda w.\lambda R.\iota x.R(x_1)(x)$

This can be composed with the translation of [$_{\text{RN}}$ mother], again using EFA, to obtain the translation of [$_{\text{NP}}$ his$_1$ mother]:

(1.25) $\lambda w.\iota x.\textsc{mother}(w)(x_1)(x)$

Two more applications of EFA give us the translation of [$_{\text{S}}$ t_1 loves his$_1$ mother]:

(1.26) $\lambda w.\textsc{love}(w)(\iota x.\textsc{mother}(w)(x_1)(x))(x_1)$

Finally, QINP tells us how to compose this with the translation of [$_{\text{NP}}$ John]^1 to get the translation of (1.23):

$$\lambda w.(\lambda x_1.(\overbrace{\boxed{\lambda w'.\textsc{love}(w')(\iota x.\textsc{mother}(w')(x_1)(x))(x_1)}}^{[t_1 \text{ loves his}_1 \text{ mother}]})(w))\left((\overbrace{\boxed{\lambda w''.\textsc{john}(w'')}}^{[\text{John}]})(w)\right)$$

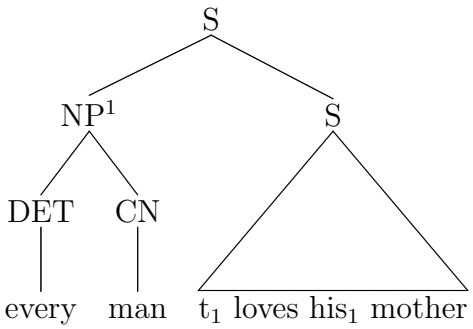
which can be reduced to:

(1.27) $\lambda w.\textsc{love}(w)(\iota x.\textsc{mother}(w)(\textsc{john}(w))(x))(\textsc{john}(w))$

If the bound pronoun [he$_1$] in (1.23) were replaced by a referential pronoun [he] with [John] as its antecedent, we would get exactly the same end result.

The example in (1.28) is very much like the one in (1.23). Only, instead of showing how noun phrases of type *se* are quantified in, it shows how *generalized quantifiers* of type $s((et)t)$ are quantified in, and also how determiners like [$_{\text{DET}}$ every] work.

(1.28) [$_{\text{S}}$ [$_{\text{NP}}$ [$_{\text{DET}}$ every] [$_{\text{CN}}$ man]]^1 [$_{\text{S}}$ t_1 loves his$_1$ mother]]



The translation of [$_{\text{S}}$ t_1 loves his$_1$ mother] was given in (1.26). The translation of [$_{\text{DET}}$ every] can be found in the lexicon:

(1.29) $\lambda w.\lambda P.\lambda P'.\forall x(P(x) \rightarrow P'(x))$

This can be composed with the translation of [$_{\text{CN}}$ man] using EFA to get the translation of [$_{\text{NP}}$ every man]:

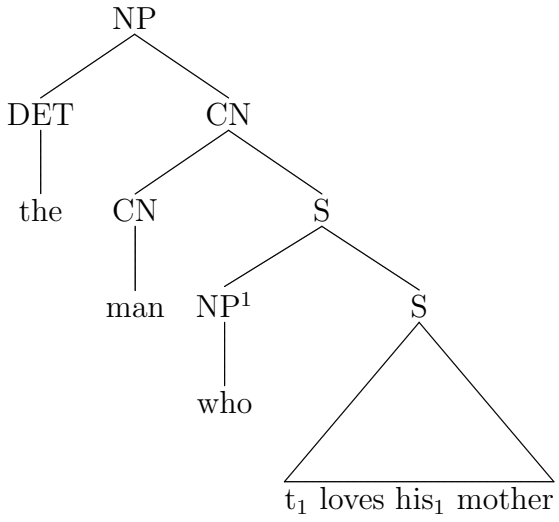
(1.30) $\lambda w.\lambda P'.\forall x(\textsc{man}(w)(x) \rightarrow P'(x))$

Now QIGQ tells us how to combine the translation of [$_{\text{NP}}$ every man]^1 with that of [$_{\text{S}}$ t_1 loves his$_1$ mother] to get the translation of (1.28):

(1.31) $\lambda w.\forall x(\textsc{man}(w)(x) \rightarrow \textsc{love}(w)(\iota x'.\textsc{mother}(w)(x)(x'))(x))$

We will do two more examples. One to illustrate how QIWH and PM deal with relative clauses, and one to show how FC deals with quantifiers in object position. First consider (1.32).

(1.32) $[_{\text{NP}}\ [_{\text{DET}}\ \text{the}]\ [_{\text{CN}}\ [_{\text{CN}}\ \text{man}]\ [_{\text{S}}\ [_{\text{NP}}\ \text{who}]^1\ [_{\text{S}}\ \text{t}_1\ \text{loves his}_1\ \text{mother}]]]]$



The translation of $[_{\text{S}}\ \text{t}_1\ \text{loves his}_1\ \text{mother}]$ was given in (1.26). The translation of $[_{\text{NP}}\ \text{who}]$ can be found in the lexicon:

(1.33) $\lambda w.\lambda P.P$

Now, QIWH tells us how to compose the translation of $[_{\text{NP}}\ \text{who}]$ with the translation of $[_{\text{S}}\ \text{t}_1\ \text{loves his}_1\ \text{mother}]$ to get the translation of the relative clause:

$$\lambda w.((\overbrace{\boxed{\lambda w'.\lambda P.P}}^{\text{[who]}})(w))(\lambda x_1.(\overbrace{\boxed{\lambda w''.\textsc{love}(w'')(\iota x.\textsc{mother}(w'')(x_1)(x))(x_1)}}^{\text{[t}_1\text{ loves his}_1\text{ mother]}})(w))$$

which reduces to:

(1.34) $\lambda w.\lambda x_1.\textsc{love}(w)(\iota x.\textsc{mother}(w)(x_1)(x))(x_1)$

The next step is to derive the translation of $[_{\text{CN}}\ \text{man who}^1\ \text{t}_1\ \text{loves his}_1\ \text{mother}]$. The translation of $[_{\text{CN}}\ \text{man}]$ is given in the lexicon:

(1.35) $\lambda w.\lambda x.\textsc{man}(w)(x)$

and PM (predicate modification) tells us how to compose this with the translation of the relative clause to get:

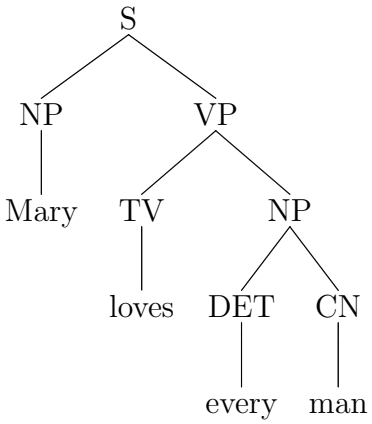
(1.36) $\lambda w.\lambda x.\textsc{man}(w)(x) \wedge \textsc{love}(w)(\iota x'.\textsc{mother}(w)(x)(x'))(x)$

Finally, EFA tells us how to compose this with the translation of [$_{\text{DET}}$ the] to get the translation of (1.32):

(1.37) $\lambda w.\iota x.\text{MAN}(w)(x) \wedge \text{LOVE}(w)(\iota x'.\text{MOTHER}(w)(x)(x'))(x)$

The last example, (1.38), shows how FC deals with quantifiers in object position.

(1.38) [$_{\text{S}}$ [$_{\text{NP}}$ Mary] [$_{\text{VP}}$ [$_{\text{TV}}$ loves] [$_{\text{NP}}$ [$_{\text{DET}}$ every] [$_{\text{CN}}$ man]]]]



Such constructions cannot be dealt with by standard function application, because quantifiers are of type $s((et)t)$ while transitive verbs are of type $s(e(et))$. Thus, transitive verbs combine, in every world, with something of type e to yield something of type et. Quantifiers don't provide something of type e but something of type $(et)t$ in every world, so function application is impossible.

But notice that the *input type* of generalized quantifiers, et, matches the *output type* of transitive verbs. If the transitive verb could just get its input elsewhere, then the generalized quantifier would know what to do with its output. This is the idea of function composition: a function f of type $s((et)t)$ and a function f' of type $s(e(et))$ are composed into a function $\lambda w.\lambda x.f(w)(f'(w)(x))$ of type $s(et)$ which, in every world w, takes an individual of type e as its input and gives as its output the result of first applying $f'(w)$ to x and then applying $f(w)$ to $f'(w)(x)$. In our concrete example, the ingredients of function composition are the translation of [$_{\text{TV}}$ loves] and the translation of [$_{\text{NP}}$ every man]:

(1.39) $\lambda w.\lambda x.\lambda y.\text{LOVES}(w)(x)(y)$

(1.40) $\lambda w.\lambda P.\forall x(\text{MAN}(w)(x) \rightarrow P(x))$

FC tells us how to compose these two functions in order to get the translation of [$_{\text{VP}}$ loves every man]:

(1.41) $\lambda w.\lambda x'.\forall x(\text{MAN}(w)(x) \rightarrow \text{LOVES}(w)(x)(x'))$

And EFA tells us how to combine (1.41) with the translation of [$_{\text{NP}}$ Mary] to get the translation of (1.38):

(1.42) $\lambda w.\forall x(\textsc{man}(w)(x) \rightarrow \textsc{loves}(w)(x)(\textsc{mary}(w)))$

This concludes the illustration of the translation of non-terminal LF nodes. Let me remark that there are many alternative ways to set up the lexicon and the composition rules. For example, if we assume more complex types in the lexicon, add *type raising* to our inventory of composition rules, and/or make quantifier raising obligatory, we could possibly do without function composition and unify the rules for quantifying in (see Heim and Kratzer, 1998, chapter 7, for some discussion). However, such adaptations would, as far as I can see, not have any significant consequences for the particular issues that are to be discussed in this dissertation. For practical convenience I have chosen here to use simple types in the lexicon and a relatively large inventory of composition rules.

We have now completely filled in the picture we started out with in the beginning of this section. First, we specified $\mathcal{F}$, $\mathcal{L}$, and the interpretation function $\mathcal{I}$ which associates expressions in $\mathcal{L}$ to meanings in $\mathcal{F}$. Then we specified how logical form constituents are translated into $\mathcal{L}$-expressions. Putting everything together, we end up with a system that assigns a meaning to every logical form constituent in our fragment.

1.3 Contextual and Conventional Meaning

Let me take a step back at this point and observe that the framework laid out above allows us to make a distinction between two kinds of meaning: contextual meaning and conventional meaning. To appreciate this distinction, notice that several factors are involved in the interpretation of linguistic expressions. First of all, to interpret (a particular usage of) an expression it is necessary to assume that that expression belongs to the vocabulary of a particular language (e.g. some dialect of English) and that it is to be interpreted according to the linguistic *conventions* to which speakers of that language adhere. In the case of English, such conventions determine, for example, how words like *chair* and *sing* are interpreted.

In addition, the interpretation of an expression often depends on the *context* in which it is used (e.g. what has been said before, what is the topic of the conversation, what is the question that is being addressed, etcetera). This is especially clear in the case of referential pronouns—their interpretation is not fixed by general conventions, but depends on the context of use.

This distinction is captured by the formal machinery developed above. We may define the following two notions of meaning:

1.8. Definition. [Contextual Meaning]
The contextual meaning of a logical form constituent X in a context C is $[\![\mathrm{X}]\!]^{\mathrm{C}}$.

1.9. Definition. [Conventional Meaning]
The conventional meaning of a logical form constituent X is the function which maps every context C to the contextual meaning of X in C.

Similarly, we can define the following notions of equivalence and entailment:

1.10. Definition. [Equivalence]
Let X and Y be two logical form constituents and let C_X and C_Y be the respective contexts in which they are used. Then:

- X and Y are contextually equivalent relative to C_X and C_Y iff $[\![X]\!]^{C_X}$ and $[\![Y]\!]^{C_Y}$ are equivalent given $\mathcal{F}$ and $\mathcal{I}$;
- X and Y are conventionally equivalent iff there are two contexts C_X and C_Y such that X and Y are contextually equivalent relative to C_X and C_Y.

1.11. Definition. [Entailment]
Let X and Y be two sentential logical forms and let C_X and C_Y be the respective contexts in which they are used. Then:

- X contextually entails Y relative to C_X and C_Y iff $[\![X]\!]^{C_X}$ entails $[\![Y]\!]^{C_Y}$ given $\mathcal{F}$ and $\mathcal{I}$;
- X conventionally entails Y iff there are two contexts C_X and C_Y such that X contextually entails Y relative to C_X and C_Y.

Intuitively, X is conventionally equivalent with Y iff they are equivalent *as far as their conventional meaning is concerned*. A similar intuition holds for conventional entailment. These fine-grained notions of meaning, equivalence, and entailment will play a significant role below, especially in section 1.8.

We now turn to the formal definition of anaphoric relations such as binding and coreference.

1.4 Anaphoric Relations

The grammatical framework laid out above allows us to formally define notions such as binding and coreference. In doing so I will try to stay as close as possible to the notions that have been discussed in the literature (be it formally or informally). Let me start with binding. The definition of binding requires the definition of one auxiliary notion, namely that of c-command.

1.12. Definition. [C-command]
One node A c-commands another node B iff (i) A does not dominate B and (ii) all branching nodes that dominate A also dominate B.

1.13. Definition. [Binding]
Let X be a logical form constituent, A a noun phrase in X with a binder index, and B a pronoun or trace in X with a binding index. Then A binds B in X iff:

i A's binder index matches B's binding index;

ii A c-commands B in X;

iii A does not c-command any other NP in X which satisfies i and ii.

This notion of binding is what Heim and Kratzer (1998) and Büring (2005a) call *semantic binding* and what Reinhart (2006) calls *A-binding*. To get a feel for what the notion amounts to consider the following examples:

(1.43) [John]^1 [t_1 loves his$_1$ mother]

(1.44) [every man]^1 [t_1 thinks that he$_1$ will win]

In (1.43), [John] binds [t_1] and [his$_1$]; in (1.44), [every man] binds [t_1] and [he$_1$].
In terms of binding we may define the following notion of *cobinding*.

1.14. Definition. [Cobinding]
Two nodes A and B in a logical form constituent X are cobound iff there is a third node which binds both A and B in X.

In (1.43), [t_1] and [his$_1$] are cobound, and in (1.44), [t_1] and [he$_1$] are cobound.

Finally, consider coreference. This relation only involves *referential* noun phrases: expressions of type *se* whose translation does not contain any free variables. The notion of coreference that is generally assumed in the literature does not require that two expressions denote the same individual in *all* possible worlds, but merely that they denote the same individual in those worlds that are consistent with the speech participants' common assumptions in a given utterance context.[7] Stalnaker (1978) called this set of possible worlds the *context set*.[8] To appreciate the idea that coreference only requires denoting the same individual in each world in the context set, consider the name *Zapatero* and the description *the President of Spain*. In a conversation between two people from Madrid, the context set will probably only include worlds in which the name and the description denote exactly the same individual. As a consequence, whenever one of the

[7]This notion of coreference is sometimes called *presupposed* coreference (cf. Büring, 2005a, p.153).

[8]The term *common ground* is often used synonymously with the term *context set*. However, as Kai von Fintel pointed out to me, Stalnaker used these terms for distinct notions. The common ground, in his terminology, is a set of presupposed *propositions*, whereas the context set is a set of *possible worlds* recognized by the speaker to be the "live options" relevant to the conversation (Stalnaker, 1978, p.84–85).

speech participants uses the name, he may just as well have used the description to convey the same message. Thus, intuitively, the name and the description corefer in such a context.

In a conversation between two people from Melbourne, the context set will probably include worlds in which *Zapatero* and *the President of Spain* do not denote the same individual. Even if both speech participants know that Zapatero is the President of Spain, they may not take for granted that their interlocutor knows this as well. Therefore, if one of them uses the description, he cannot be sure that using the name instead would convey the same message. Thus, intuitively, the name and the description do not corefer in such a context. This idea can be formalized as follows:

1.15. Definition. [Coreference]
Let C be a context, and let S_C be the context set in C. Then, two referential noun phrases A and B corefer in C iff for every $w \in S_C$, $[\![\mathrm{A}]\!]^{\mathrm{C}}(w)$ is equivalent to $[\![\mathrm{A}]\!]^{\mathrm{C}}(w)$ given $\mathcal{F}$ and $\mathcal{I}$.

This concludes the definition of anaphoric relations. Let us now return to the examples discussed at the very beginning of this chapter, repeated here:

(1.3) MAX called his mother.

(1.4) Only MAX called his mother.

(1.5) Max called his mother and Bob did too.

It was suggested that the distinction between bound and referential pronouns would yield a natural explanation of the ambiguities exhibited by these examples. We are almost ready to spell out this explanation in detail. The final ingredient we need is a basic theory of focus.

1.5 Focus

Theories of focus (cf. Rooth, 1985) generally assume that constituents may or may not be F-marked at surface structure. For example, the surface structure in (1.45) has an F-marked subject NP.

(1.45) $[\text{the dog}]_{\mathrm{F}}$ [destroyed the vase]

F-marking is interpreted both phonologically (at PF) and semantically (at LF). The phonological interpretation of F-features consists in *accenting* certain syllables within each F-marked constituent. For example, the surface structure in (1.45) will be pronounced as:

(1.46) the DOG destroyed the vase

Which of the syllables in an F-marked constituent are accented depends on various factors, which are not directly relevant here (cf. Büring, 2007).

The semantic import of F-features is their role in determining the *focus alternatives* of LF constituents. The focus alternatives of an LF constituent X are obtained from X by replacing its F-marked sub-constituents with contextually salient alternatives. For example, some of the focus alternatives of (1.45) may be:

(1.47) a. [the cat] [destroyed the vase]
b. [the burglar] [destroyed the vase]
c. [a friend of my father] [destroyed the vase]

Let us write $\text{ALT}^{\text{C}}(\text{X})$ for the set of focus alternatives of X in C, and let us call:

$$[\![\text{X}]\!]^{\text{C,F}} = \{[\![\text{Y}]\!]^{\text{C}} \mid \text{Y} \in \text{ALT}^{\text{C}}(\text{X})\}$$

the *focus value* of X in C (to avoid confusion, $[\![\text{X}]\!]^{\text{C}}$ and $[\![\text{X}]\!]^{\text{C,F}}$ are sometimes called the *ordinary* semantic value and the *focus* semantic value of X in C, respectively).[9] Focus alternatives play a role in a variety of linguistic phenomena. Notorious examples are the computation of *implicatures*, which figure in examples like (1.3) and will be discussed in section 1.6, the interpretation of focus-sensitive operators, which play a role in examples like (1.4) and will be discussed in section 1.7, and the interpretation of VP ellipsis, which is relevant for examples like (1.5) and will be discussed in section 1.8.

1.6 Implicatures

Consider the following scenario, taken from Rooth (1992b). Mats, Steve, and Paul are taking an exam, which is graded right away. When Mats comes home, his brother George asks how it went. Mats answers:

(1.48) Well, I PASSED.

Given this answer, George will probably conclude that Mats did not do better than passing, that he did not, for example, ace the exam.

Now consider another answer Mats could have given:

(1.49) Well, STEVE passed.

Given this answer, George would probably conclude that Mats and Paul did not pass. The general line of reasoning that leads to this conclusion could be the

[9] For simplicity, I assume here that focus alternatives are determined at a *syntactic* level. Rooth (1985) assumed that they are determined at a *semantic* level. For the particular phenomena to be discussed here, it does not really matter which of these assumptions is adopted. I have adopted the first just to keep things as simple as possible.

following: Mats said that Steve passed. If he or Paul had passed as well, he would have said so. He didn't, so he and Paul probably didn't pass.

In the case of (1.48), George's reasoning is similar: Mats said that he passed. If he had aced he would have said so. He didn't, so he probably didn't ace.

Grice (1975) called the conclusions that arise from such reasoning patterns *implicatures*. The role of focus in the computation of implicatures is to determine the appropriate set of comparison. A given logical form LF is always compared with its focus alternatives. For example, (1.48) is compared with its focus alternatives [I failed] and [I aced]. The Gricean reasoning, then, amounts to taking every focus alternative of LF to be false, unless it is contextually entailed by LF itself. For example, (1.48) implicates that I did not ace (if I had, I would have said so).

Similarly, (1.49) is compared with its focus alternatives [nobody passed], [Mats passed], [Paul passed], [Steve and Mats passed], [Steve and Paul passed], [Mats and Paul passed], and [Steve, Mats and Paul passed]. The alternatives that are not contextually entailed by (1.49) are taken to be false, resulting in the implicature that Mats and Paul did not pass.[10]

We are now ready to consider the ambiguity in (1.3), repeated below:

(1.3) MAX called his mother.

The pronoun can either be bound or referential. Let us first take it to be referential, with [Max] as its antecedent:

(1.50) $[\text{Max}]_{\text{F}}$ [called his mother] his = Max

Now suppose that [John], [Bill], and [Fred] are the contextually salient alternatives of [Max]. Then the focus alternatives of (1.50) are:

(1.51)
a. [John] [called his mother] his = Max
b. [Bill] [called his mother] his = Max
c. [Fred] [called his mother] his = Max

These alternatives are not contextually entailed by (1.50), so they are taken to be false. In other words, (1.50) implicates that John, Bill, and Fred did not call Max's mother. This is indeed one of the possible readings of (1.3).

Now suppose that the pronoun in (1.3) is bound by [Max]:

(1.52) $[\text{Max}]^{1}_{\text{F}}$ [t_1 called his_1 mother]

Suppose again that [John], [Bill], and [Fred] are the contextually salient alternatives of [Max]. Then the focus alternatives of (1.52) are:

[10] This is a simplified picture of course. For a more complete story about implicatures see Davis (2005) and the references given there.

(1.53) a. [John]^1 [t_1 called his$_1$ mother]
b. [Bill]^1 [t_1 called his$_1$ mother]
c. [Fred]^1 [t_1 called his$_1$ mother]

These alternatives are not contextually entailed by (1.52), so they are taken to be false. In other words, (1.52) implicates that John, Bill, and Fred did not call their own mother. This is the second possible reading of (1.3).

Thus, we may conclude that the ambiguity in (1.3) is explained in a natural way if the basic framework presented here is combined with a standard theory of focus and implicature.

1.7 Only

Next, let us consider the interpretation of focus-sensitive operators. In fact, I will add one of them, *only*, to our basic fragment. But let me first illustrate why operators like *only* are called focus-sensitive. Consider the following sentences:

(1.54) John only introduced BILL to Sue.

(1.55) John only introduced Bill to SUE.

These sentences illustrate that different intonation patterns in the scope of *only* lead to different interpretations: (1.54) says that John introduced Bill, and no one else, to Sue, while (1.55) says that John introduced Bill to Sue, and to no one else. This is why *only* is called a focus-sensitive operator.

Next, let us consider the syntactic distribution of *only*.

(1.56) Bill called only SUE.

(1.57) Bill only called SUE.

(1.58) Bill only CALLED Sue.

(1.59) Only BILL called Sue.

(1.60) *Only Bill CALLED Sue.

(1.61) *Only Bill called SUE.

(1.56), (1.57), and (1.58) show that *only* can adjoin both to NP and to VP. In (1.59), *only* could be analyzed as adjoined to NP or as adjoined to S. (1.60) and (1.61) seem to suggest that *only* cannot be adjoined to S. Together with the assumption that *only* must associate with some focused element in the phrase to which it adjoins, this would explain why (1.60) and (1.61) are ungrammatical. I don't know of any alternative explanation for this fact.

However, other examples seem to suggest that it *is* possible for *only* to adjoin to S. Consider the following scenario, adapted from (Jacobson, 2007): every year, I have a large number of people over for Thanksgiving. I am very grumpy about

the fact that in general people don't help out enough and don't bring enough food. I turn to you and ask:

(1.62) Do you think anyone will help out this year? Will anyone bring some extra turkey? Some salad or some wine? Or at least some extra chairs?

You answer:

(1.63) I'm afraid only SUE will bring some SALAD this year.

This sentence does not mean that Sue is the only one who will bring some salad this year, but rather that Sue will bring some salad and that nobody else will bring anything else. To accommodate such cases I will assume that *only* may adjoin to S as well as to NP and VP, and that there is an alternative explanation for the ungrammaticality of (1.60) and (1.61). Thus let us add the following rules to the syntax of our fragment:

(PS 10) S → only S
(PS 11) VP → only VP
(PS 12) NP → only NP

Next let us consider the semantics of *only*. First, consider the case of [only S]. Intuitively, a phrase like [only Bill_F loves Mary] is true iff [Bill_F loves Mary] is true and all the focus alternatives of [Bill_F loves Mary] are false. Formally:

(1.64) $[\![\text{only S}]\!]^\text{C} = \lambda w.\varphi(w) \wedge \neg\psi_1(w) \wedge \ldots \wedge \neg\psi_n(w)$

where φ is $[\![\text{S}]\!]^\text{C}$ and $\psi_1, \ldots, \psi_n$ are all the elements of $[\![\text{S}]\!]^{\text{C,F}}$. If the S in question is [Bill_F loves Mary], and the alternatives of [Bill] are [John] and [Fred], then:

(1.65) a. $\varphi = \lambda w.\textsc{loves}(w)(\textsc{mary}(w))(\textsc{bill}(w))$
b. $\psi_1 = \lambda w.\textsc{loves}(w)(\textsc{mary}(w))(\textsc{john}(w))$
c. $\psi_2 = \lambda w.\textsc{loves}(w)(\textsc{mary}(w))(\textsc{fred}(w))$

This yields the following translation of [only Bill_F loves Mary]:

(1.66) $\lambda w.\textsc{loves}(w)(\textsc{mary}(w))(\textsc{bill}(w))$
$\wedge \neg\textsc{loves}(w)(\textsc{mary}(w))(\textsc{john}(w))$
$\wedge \neg\textsc{loves}(w)(\textsc{mary}(w))(\textsc{fred}(w))$

which indeed matches our intuitions about the meaning of *only*.[11]

Now consider the case of [only VP]. Intuitively, [only likes Bill_F] expresses a property which holds of all individuals who like Bill, and no other contextually salient individuals. In other words, [only likes Bill_F] expresses a property which

[11] Again, this is of course a simplified picture. For more details on the meaning of *only* see (Ippolito, 2007; van Rooij and Schulz, 2007) and the references given there.

holds of all individuals who have the property expressed by [likes Bill_F] and who do not have the properties expressed by the focus alternatives of [likes Bill_F]. Formally:

(1.67) $[\![\text{only VP}]\!]^\text{C} = \lambda w.\lambda x.\varphi(w)(x) \wedge \neg\psi_1(w)(x) \wedge \ldots \wedge \neg\psi_n(w)(x)$

where φ is $[\![\text{VP}]\!]^\text{C}$ and $\psi_1, \ldots, \psi_n$ are all the elements of $[\![\text{VP}]\!]^{\text{C,F}}$. If the VP in question is [likes Bill_F], and the alternatives of [Bill] are [John] and [Mary] then:

(1.68) a. $\varphi = \lambda w.\lambda x.\textsc{likes}(w)(\textsc{bill}(w))(x)$
b. $\psi_1 = \lambda w.\lambda x.\textsc{likes}(w)(\textsc{john}(w))(x)$
c. $\psi_2 = \lambda w.\lambda x.\textsc{likes}(w)(\textsc{mary}(w))(x)$

which results in the following translation of [only likes Bill_F]:

(1.69) $\lambda w.\lambda x.\textsc{likes}(w)(\textsc{bill}(w))(x)$
$\wedge \neg\textsc{likes}(w)(\textsc{john}(w))(x)$
$\wedge \neg\textsc{likes}(w)(\textsc{mary}(w))(x)$

Finally let us consider the case of [only NP]. As an example, consider [only Sue_F sleeps]. Intuitively, this means that Sue sleeps, and that other contextually salient individuals do not sleep. Or in other words, that the individual denoted by $[\text{Sue}]_\text{F}$ sleeps, while the individuals denoted by all the focus alternatives of $[\text{Sue}]_\text{F}$ do not sleep. Formally:

(1.70) $[\![\text{only NP}]\!]^\text{C} = \lambda w.\lambda P.P(w)\varphi(w) \wedge \neg P(w)\psi_1(w) \wedge \ldots \wedge \neg P(w)\psi_n(w)$

where φ is $[\![\text{NP}]\!]^\text{C}$ and $\psi_1, \ldots, \psi_n$ are all the elements of $[\![\text{NP}]\!]^{\text{C,F}}$. Notice that the translation of [only NP] is not of type *se* but of type $s((et)t)$ (it's a generalized quantifier). If the NP in question is $[\text{Sue}]_\text{F}$, and the alternatives of [Sue] are [Fred] and [Bill], then:

(1.71) a. $\varphi = \lambda w.\textsc{sue}(w)$
b. $\psi_1 = \lambda w.\textsc{fred}(w)$
c. $\psi_2 = \lambda w.\textsc{bill}(w)$

which results in the following translation of [only Sue_F sleeps]:

(1.72) $\lambda w.\textsc{sleeps}(w)(\textsc{sue}(w))$
$\wedge \neg\textsc{sleeps}(w)(\textsc{fred}(w))$
$\wedge \neg\textsc{sleeps}(w)(\textsc{bill}(w))$

Given this treatment of *only* we may now turn to the ambiguity in (1.4), repeated below:

(1.4) Only MAX called his mother.

Suppose that the alternatives of [Max] are [John] and [Bill]. Now, if the pronoun in (1.4) is referential, with [Max] as its antecedent, then we obtain the following translation:

(1.73) $\lambda w.\textsc{called}(w)(\iota x.\textsc{mother}(w)(\textsc{max}(w))(x))(\textsc{max}(w))$
$\wedge\neg\textsc{called}(w)(\iota x.\textsc{mother}(w)(\textsc{max}(w))(x))(\textsc{john}(w))$
$\wedge\neg\textsc{called}(w)(\iota x.\textsc{mother}(w)(\textsc{max}(w))(x))(\textsc{bill}(w))$

In words: Max called his mother, and the others did not call Max's mother. If the pronoun in (1.4) is bound by [only Max], then we obtain the following translation:

(1.74) $\lambda w.\textsc{called}(w)(\iota x.\textsc{mother}(w)(\textsc{max}(w))(x))(\textsc{max}(w))$
$\wedge\neg\textsc{called}(w)(\iota x.\textsc{mother}(w)(\textsc{john}(w))(x))(\textsc{john}(w))$
$\wedge\neg\textsc{called}(w)(\iota x.\textsc{mother}(w)(\textsc{bill}(w))(x))(\textsc{bill}(w))$

In words: Max called his mother, and the others didn't call their own mother.

Thus we may conclude that the ambiguity that arises in (1.4) is explained in a natural and straightforward way if our basic framework is combined with a simple theory of the interpretation of focus-sensitive operators like *only*.

Finally, let us turn to the ambiguity in constructions such as (1.5):

(1.5) Max called his mother and Bob did too.

In order to explain this ambiguity, we need a basic theory of VP ellipsis.

1.8 VP ellipsis

It is often assumed that ellipsis is the result of deleting certain material at PF (cf. Sag, 1976; Heim and Kratzer, 1998; Merchant, 2001). The exact conditions under which such deletion is licensed is subject to an ongoing debate. The present framework may shed some new light on this debate.

LF Identity. Sag (1976) and Williams (1977) proposed that a constituent may only be deleted at PF if it is identical to some other constituent at LF (which itself is *not* deleted at PF). This constraint, which is known as the LF Identity condition, can still be found in many textbooks (cf. Heim and Kratzer, 1998).

1.16. Definition. [LF Identity]
A constituent may be deleted at PF only if it is identical to another constituent at LF, which itself is not deleted at PF.

Semantic Identity. However, Sag and Hankamer (1984) already observed that the following examples are problematic for LF Identity:

(1.75) Do you think they will like me? - Of course they will.

(1.76) Could you come over here, please? - Of course I could.

The elided VP in (1.75) is [like you], while its antecedent is [like me]. Similarly, the elided VP in (1.76) is [come over there], while its antecedent is [come over here]. Both are legitimate cases of ellipsis, even though the LF representation of the elided VP differs from the LF representation of its antecedent VP. Sag and Hankamer concluded from this data that LF Identity is not really what is at stake. Rather, the relevant identity constraint must be *semantic*: the elided VP and the antecedent VP must be semantically equivalent.

Now recall that in the present framework, there are two notions of semantic equivalence: contextual equivalence (relative to the given context) and conventional equivalence (relative to *some* context). Thus we may define the following two Semantic Identity conditions.

1.17. Definition. [Strong Semantic Identity]
A constituent may be deleted at PF only if it is contextually equivalent to another constituent at LF, which itself is not deleted at PF.

1.18. Definition. [Weak Semantic Identity]
A constituent may be deleted at PF only if it is conventionally equivalent to another constituent at LF, which itself is not deleted at PF.

To see which of these conditions is more adequate consider the following example:

(1.77) a. Sue: You won't believe what Sam just told me.
b. Ann: What?
c. Sue: John wants to marry his sister, and Bill does too.

Suppose that the pronoun in the antecedent VP in (1.77c) is resolved to Sam. Then the antecedent VP as a whole is interpreted as *wants to marry Sam's sister*, and the elided VP must also be interpreted as *wants to marry Sam's sister*. This is correctly predicted if the elided VP is required to be contextually equivalent with the antecedent VP (Strong Semantic Identity). If mere conventional equivalence were required (Weak Semantic Identity), then the elided VP could just as well be interpreted as *wants to marry John's sister* or *wants to marry Bill's sister*. So Strong Semantic Identity seems more adequate than Weak Semantic Identity. Or in other words, the notion of semantic equivalence relevant for VP ellipsis seems to be *contextual* equivalence rather than *conventional* equivalence.

Strong Semantic Identity accounts for Sag and Hankamer's examples if Kaplan's (1989) semantics for indexicals is adopted. It also solves another problem for LF Identity, which was discussed by Fiengo and May (1994):

(1.78) Mary loves John, and he thinks that Sally does too.

This sentence has a reading on which John thinks that Sally loves him too. Two possible LFs that would correspond to this reading are:

(1.79) a. Mary [loves John], and he thinks that Sally [loves John], too.
he = John

b. Mary [loves John], and he thinks that Sally [loves him], too.
he = him = John

LF Identity only admits (1.79a) as a possible LF of (1.78), because in (1.79b) the elided VP and its antecedent are not identical. The problem with (1.79a) is that, if the elided VP were *not* elided, then *he* could never be interpreted as referring to John.

(1.80) He thinks that Sally loves John, too. $\Rightarrow$ he $\neq$ John

It would be hard to explain how VP ellipsis suddenly makes this interpretation available. This problem does not arise for Strong Semantic Identity, which accepts (1.79b) as a possible LF of (1.78).

Focus Match. However, Strong Semantic Identity on its own is sometimes not strong enough to rule out illegitimate cases of VP ellipsis. To see this, consider the following example, adapted from Rooth (1992a):

(1.81) John's sister thinks he might have a chance, and Bill does too.

(1.82) [John]^1 [[t_1's sister] [thinks he$_1$ might have a chance]], and
[Bill]^1 [t_1 [thinks he$_1$ might have a chance]] too.

The VP in gray has an identical antecedent, so as far as Strong Semantic Identity is concerned, ellipsis is licensed. The problem is that the second conjunct of (1.81) can only be taken to mean that Bill thinks that John might have a chance, *not* that Bill thinks he himself might have a chance, which is the reading represented by (1.82). This observation can be accounted for by adopting the following additional constraint on VP ellipsis (cf. Rooth, 1992a; Tancredi, 1992; Heim, 1997; Tomioka, 1997; Fox, 1999b; Merchant, 2001):

1.19. Definition. [Focus Match]
VP ellipsis is licensed only if the elided VP is dominated by some sentential constituent S_E which focus-matches some other sentential constituent S_A (its *antecedent*). S_E focus-matches S_A iff S_A contextually entails an element of the focus value of S_E.

Intuitively, Focus Match says that an elided VP must always be contained in a clause that *contrasts appropriately* with another clause in the discourse. Let us see how this idea accounts for the fact that ellipsis is ruled out in (1.82). The first sentential constituent dominating the gray VP is [t_1 [thinks he$_1$ might have a chance]]. This phrase does not focus-match any other phrase in the discourse,

so it does not license ellipsis. Another sentential constituent which dominates the gray VP is the entire second conjunct of (1.82):

(1.83) [Bill]^1 [t_1 [thinks he$_1$ might have a chance]]

But again, this phrase does focus-matches any other phrase in the discourse, even if we take [Bill] to be F-marked. Conclusion: Focus Match indeed predicts that ellipsis is not licensed in (1.82).

This is all very well, but still we may ask whether Rooth's example really justifies the stipulation of an additional condition on VP ellipsis such as Focus Match. Doesn't the fact that (1.81) does not have a sloppy reading simply follow from the meaning of the particle *too* (which has been ignored so far)? In fact it does: the use of a phrase [S too] roughly requires that some other phrase in the discourse contextually entails one of the focus alternatives of S. This requirement is not fulfilled in (1.82). So to account for Rooth's example we do not need to stipulate Focus Match. However, there are many parallel examples which do not involve the particle *too*:

(1.84) John's sister thinks he might have a chance. Bill$_F$ doesn't$_F$.

(1.85) John's sister thinks he might have a chance, because Bill$_F$ does.

(1.86) John's sister talked to his coach before Bill$_F$ did.

These sentences do not have sloppy readings either, and something like Focus Match is indeed required to account for this fact. In many of the examples below I will ignore the particle *too*, given that it is always possible to construct parallel examples without *too*.

It is important to point out that, even though Focus Match is stated here as a special condition on VP ellipsis, it should really be thought of as a *corollary* of a much more general theory about the encoding of *information structure*. In English, and in many other languages, information structure is encoded by means of intonation (especially accentuation) and by means of word order. There are also languages in which information structure is encoded by means of special morphemes (cf. Büring, 2007). How a theory of information structure should be formulated exactly, and how something like Focus Match should follow from it, is of course subject to a large ongoing debate (cf. Schwarzschild, 1999; Tomioka, 1997). Here I will abstract away from this debate and simply assume that VP ellipsis must comply with Focus Match.

Semantic Identity Reconsidered. Roughly put, the difference between Strong and Weak Semantic Identity is this: Strong Semantic Identity forces a referential pronoun in an elided VP to refer to the same individual as the corresponding pronoun in the antecedent VP; Weak Semantic Identity does not force this, it allows referential pronouns in the elided VP to "shift" their reference to another

individual. Example (1.77) was meant to show that such shifts in reference are generally not allowed. This lead us to the conclusion that Strong Semantic Identity should be adopted rather than Weak Semantic Identity. But once Focus Match is adopted, this conclusion should be reconsidered: Focus Match seems to prohibit exactly those kind of shifts in reference that Weak Semantic Identity by itself wrongly permits. For example, in the case of (1.77), the problem with Weak Semantic Identity was that it allows readings like:

(1.87) John wants to marry Sam's sister, and Bill wants to marry John's sister too.

(1.88) John wants to marry Sam's sister, and Bill wants to marry Bill's sister too.

But these readings are ruled out by Focus Match, because the two conjuncts do not contrast appropriately. Thus, once Focus Match is in place, we could reconsider Weak Semantic Identity as an alternative for Strong Semantic Identity. I will not attempt to tease these two options apart. In any case, for all the examples discussed below it does not really matter whether Strong or Weak Semantic Identity is adopted alongside Focus Match. Let me therefore simply say from now on that VP ellipsis is subject to a condition called *VP Identity*, meaning that it is subject to Focus Match and either Strong or Weak Semantic Identity.

Strict and Sloppy Readings. Let us now finally turn to the ambiguity in (1.5).

(1.5) Max called his mother and Bob did too.
  a. ... Bob called his own mother too. [sloppy]
  b. ... Bob called Max's mother too. [strict]

This ambiguity is now straightforwardly accounted for. First notice that the pronoun in the source clause may be either bound or referential. Suppose it is bound. Then the source clause has the following LF:

(1.89) [Max]^1 [t_1 called his$_1$ mother]

By VP Identity, the LF of the target clause must then be:

(1.90) [Bob]^1 [t_1 called his$_1$ mother] too

This gives us the sloppy reading in (1.5a). Now suppose the pronoun in the source clause is referential to Max:

(1.91) [Max]^1 [t_1 called his mother] his = Max

Then, by VP Identity, the target clause must have either one of the following LFs:

(1.92) [Bob]^1 [t_1 called his mother] too his = Max

(1.93) [Bob]^1 [t_1 called Max's mother] too

Both LFs represent the strict reading in (1.5b). Thus, we may conclude that the present framework accounts for all the ambiguities that we started out with at the beginning of this chapter.

1.9 Summary

Let me briefly summarize what has been established in this chapter. We started out by motivating the distinction between bound and referential pronouns. The basic motivation was that pronouns seem to be interpreted as bound variables in constructions like:

(1.1) Every man thinks he will win.

whereas they seem to be interpreted as referential expressions in other constructions such as:

(1.2) John is in good shape. I think he will win.

Furthermore, it was suggested that a distinction between bound and referential pronouns would naturally explain the ambiguity of constructions like:

(1.3) MAX called his mother.

(1.4) Only MAX called his mother.

(1.5) Max called his mother and Bob did too.

Next, we set out to formulate a formal framework in which bound and referential pronouns are clearly distinguished: a bound pronoun comes with an index and is translated as a variable with that same index, a referential pronoun inherits its translation from its antecedent, which is determined contextually. To explain the ambiguities in (1.3)-(1.5), a basic theory of focus, implicature, focus-sensitive operators like *only*, and VP ellipsis was presented. The ambiguities were accounted for in a straightforward way, and in passing it was observed that the present framework may shed some new light on the identity condition that is supposed to govern VP ellipsis.

The next chapter will start to explore constraints on binding and coreference.