Ambiguous Noun Phrases in Logical Form

Mary P. Harper
School of Electrical Engineering
Purdue University
West Lafayette, IN 47907

October 1, 1990

Abstract

In this paper, we describe how to represent pronouns, singular definite noun phrases, and singular indefinite noun phrases in logical form given their parse trees, while obeying our computational constraints. Initially, the noun phrases are represented using a composite representation for their possible meanings, in keeping with the compactness constraint. The initial representation is specified using only syntactic and sentence level information, obeying our modularity constraint. Finally, when ambiguity is resolved, the precise behavior of the noun phrases is pinpointed in a way compatible with the initial representation in order to obey the formal consistency constraint.

1 Introduction

A goal of natural language research is to provide a computer model capable of understanding English sentences. This model must be able to generate an unambiguous internal representation for each sentence processed. The building of this representation can be approached in two different ways.

The first requires the generation of an unambiguous internal representation for each sentence before attempting to represent subsequent sentences. A problem arises when the current sentence is ambiguous and the information contained in subsequent sentences is required to resolve the ambiguity. One way to handle this problem is to make a best guess at the meaning of the current sentence, but if the guess turns out to be incorrect then the system would need some way of recovering, which would most likely require storing information about the ambiguous sentence and subsequent sentences. Another way to handle this problem is to look ahead at subsequent sentences (without representing them) for clues to help in resolving the ambiguity, but this approach fails because some representation of the current sentence and succeeding sentences is needed to reason about the ambiguity.

A second approach, the one adopted in this paper, is to avoid immediately committing to a single meaning of an ambiguous sentence by using an intermediate representation called logical form (Schubert

*This paper contains results from the author’s thesis in the Computer Science Department at Brown University. The paper has benefited from discussions with Eugene Charniak, Paul Harper, Kate Sanders, Leora Morgenstern, Tom Dean, and Frederic Evans. The work was supported in part by the NSF grants IST 8416034, IST 8515005, and IST 9011179 and ONR grant N00014-79-C-0529, and AFOSR grant F49620-88-c-0132.
and Pelletier [SP84], Allen [All87], Harper [Har88]). A logical form representation partially specifies the meaning of a sentence based on syntactic and sentence-level information, without considering the affects of pragmatics and context. This partial specification of meaning allows us to process additional sentences before picking a single meaning of the current ambiguous sentence. Later, as we process the subsequent sentences, the intermediate representation of the current sentence can be incrementally updated. This process would continue until all ambiguities are resolved and the final unambiguous representation of the sentence is generated.

By using logical form, the process of deriving the meaning of a sentence is divided into several modules, as shown in Figure 1. Notice that three levels of representation are needed in this architecture, as well as three algorithms to map one representation into the next. A parse tree, provided by a sentence parsing algorithm, indicates only the structural properties of a sentence. Logical form, on the other hand, provides information about the logical roles of noun phrases while preserving some syntactic information (e.g., the surface subject could be identified). In this paper, we determine how a sentence and its components (in particular, the logical form for pronouns, singular definite noun phrases, and singular indefinite noun phrases) should be represented in logical form given its parse tree. Once logical form is combined with contextual information, a single internal representation for the sentence's meaning should be provided. We consider how logical form is updated into a final unambiguous internal representation. However, because the effect of context on language comprehension is not fully understood, mapping logical form to internal representation is not completely specified.

Because logical form is a component of a computer model for language comprehension, it must be designed with two goals in mind. First, it should accurately model the linguistic behavior of language. Second, it should be compatible with the goal of devising a computationally tractable model of language comprehension. In fact, we have defined three constraints for using logical form in a computational framework ([Har88] and [Har90]).

1. **Compactness Constraint:** Logical form should compactly represent ambiguity.

2. **Modularity Constraint:** Logical form should be initially computable from syntax and local
(sentence-level) semantics. In particular, logical form should not be dependent on pragmatics, which requires inference and hence internal representation.

3. **Formal Consistency Constraint:** Further processing of logical form should only disambiguate or further specify logical form. Logical form has a meaning. Any further processing must respect that meaning.

The compactness constraint captures the spirit of logical form as presented by Allen [All87]. Compactness is important in a computer model of language comprehension because of the need to postpone decisions about ambiguity without imposing large storage requirements. Logical form should compactly represent the underdetermined meaning of a sentence, underdetermined in that ambiguities and anaphora are not resolved. The modularity constraint is important because it allows us to consider local semantics independently of ambiguities like scoping and anaphora, which require contextual information to resolve. To obey the modularity constraint while providing the logical form for a sentence, a system must use only the information available in the lexicon about the meanings of the words in the sentence along with the syntactic structure of that sentence. If the modularity constraint is ignored, the value of computing logical form is greatly diminished. Finally, the formal consistency constraint is important because it limits the way that logical form can be updated, making the system easier to understand.

Initially, logical form provides a composite representation for a sentence. However, as more information becomes available, then the meaning of the sentence is incrementally updated (without contradicting its original meaning) until all ambiguity is resolved.

In the rest of this paper, we describe how logical form can be used to postpone decisions about the meanings of anaphoric noun phrases and quantifier scope ambiguity. First, we describe how logical form was first used to postpone decisions about sentences with quantifier scope ambiguity. Then, we introduce our logical form representation for a sentence and its components. In particular, we introduce logical form representations for pronouns, singular definite noun phrases, and singular indefinite noun phrases, consider how to update these initial representations, discuss our implementation of this approach, and review related work.

### 2 Quantifier Scope Ambiguity in Logical Form

A major advantage of using logical form is that it allows a language understanding system to postpone decisions about the final meaning of ambiguous sentences. Thus, it has been used by several researchers to handle quantifier scope ambiguities (see Schubert and Pelletier [SP84] and Allen [All87]) like the one in example 1.

**Example 1**

Someone loves everyone.

**Meanings:**

1. $\exists x \forall y (\text{love } x \ y)$
2. $\forall y \exists x (\text{love } x \ y)$

This sentence has two meanings depending on whether the universal has scope over the existential or vice versa. Schubert and Pelletier [SP84] and Allen [All87] avoid committing to a single meaning for a sentence with quantifier scope ambiguity by placing quantifiers in the predicate-argument structure.
representing the meaning of the sentence. Hence, the sentence in example 1 is represented as shown in 2.

Example 2
Someone loves everyone.
(love [∃x x] [∀y y])
Meaning:
(or ∃x ∀y (love x y)
∀y ∃x (love x y))

Because the quantifiers are stored in the predicate argument-structure with no scoping information indicated, this representation does not commit to either meaning indicated in 1; it is simply a compact way of expressing the disjunction of possible readings. Later, when more information is available, the initial logical form for the sentence is modified to indicate quantifier scope information. Schubert and Pelletier [SP84] indicate quantifier scoping by extracting and ordering the quantifiers to the left of the predicate-argument structure. Allen indicates for each pair of quantifiers which has scope over the other, a method which is not limited to a linear sequence of operators when expressing quantifier scoping information1.

Both Schubert and Pelletier [SP84] and Allen [All87] avoid committing to a single meaning for a sentence with quantifier scope ambiguity by placing quantifiers in the predicate-argument structure representing the meaning of the sentence; a representation which is consistent with our compactness constraint. Each approach obeys the modularity constraint by using only syntactic and lexical knowledge to provide the logical form for a sentence. Later, when the information necessary to pick a single meaning has been processed, they provide a way to indicate which quantifier has scope over the other. Because the modification of logical form simply commits to one of the meanings encoded in the logical form, each approach obeys the formal consistency constraint. Hence, their approaches are compatible with quantifier behavior and consistent with our constraints.

Other semantic ambiguities can be maintained in logical form until information is available for selecting a single meaning. For example, the precise meaning of the sentence Every man showed every boy a picture of his mother cannot be specified until we have the information necessary to select the intended antecedent for the pronoun and to determine the quantifier scoping. The pronoun his can have many different antecedents, for example: every man, every boy, some entity introduced in other sentences, or some individual in the environment of the speaker or hearer. The meaning of the singular definite noun phrase his mother is ambiguous because it depends on the antecedent chosen for the embedded pronoun. Also, the meaning of the singular indefinite a picture of his mother is ambiguous because its meaning cannot be pinpointed until we determine the meaning of his mother and decide whether the two universally quantified noun phrases in the sentence have scope over it. In this paper, we consider how to compactly represent these three types of ambiguous noun phrases in logical form.

1Hintikka [Hin79b, Hin79a] has noted that the linear ordering of quantifiers is not sufficient to capture all possible meanings of a sentence when four or more quantifiers occur in the sentence. Consider a sentence with four quantifiers, two universal and two existential: Every boy, wanted every girl, to introduce a friend of his, to a friend of hers. It should be possible for every boy to have scope over a friend of his, without having scope over a friend of hers. Similarly, every girl could have scope over a friend of hers, but not a friend of his. There is no way to express this in a linear quantifier scoping string, but Allen has no trouble indicating non-linear scoping using his approach.
3 Pronouns in Logical Form

Pronouns are a source of underspecification in a sentence. The antecedent of a pronoun cannot be determined using syntactic information alone; contextual and syntactic constraints combine to allow a listener/reader to decide on the antecedent for a certain pronoun. The precise meaning of a pronoun can be postponed by adding a level of logical form, dividing the process of determining the meaning of a pronoun into two phases. First, the representation for the pronoun is determined using only syntactic and sentence-level information. Then, once the antecedent is determined, a feat which often requires pragmatic and contextual information available in subsequent sentences, we provide a way to update our logical form to indicate this information.

In the rest of this section, we discuss the linguistic behaviors of pronouns to model, introduce the logical form representation for a pronoun, discuss how that representation is updated once contextual information isolates an antecedent for the pronoun, and describe how the approach handles examples of verb phrase ellipsis.

3.1 Pronouns: Linguistic Evidence

Pronouns are always anaphoric. Either they have linguistic antecedents or depend on salient individuals in the environment of the speaker or hearer (deictic use). Pronouns with linguistic antecedents can be categorized in two ways; either their antecedents occur in the same sentence (intrasentential reference) or in other sentences (intersentential reference).

When the antecedent of a pronoun occurs in the same sentence and is represented as a universally quantified variable, that pronoun takes on the behavior of that variable. Consider the sentence *Fred showed every girl her picture*. Given that the antecedent for *her* is *every girl*, the pronoun adopts the behavior of a variable bound by the universal quantifier of its antecedent, as shown in example 3.

Example 3
Fred showed every girl$_i$ her$_i$ picture$_i$. 
$\forall x \text{ (if (girl } x) \text{ (show Fred (} x \text{'s picture) } x))$.

In contrast, if a pronoun's antecedent occurs in another sentence, then that pronoun cannot act like a bound variable. Quantifiers do not have scope across sentences in English. Consider the following two sentences: *Fred likes everyone*. But, *he doesn't return the sentiment*. The pronoun in the second sentence cannot be bound by the quantifier corresponding to *everyone* in the first sentence. However, a quantified noun phrase in one sentence can be the antecedent for a pronoun in another sentence, as in *Fred likes everyone*. But, *they don't return the sentiment*. In this case, *they* can take *everyone* as its antecedent without acting like a quantified variable; it adopts the discourse entity for the group of individuals that *everyone* quantifies over. Webber [Web78] discusses how to construct discourse entities.

---

2 By reference, we do not mean that the pronoun denotes its linguistic antecedent, rather that it adopts the behavior of its antecedent.

3 A discourse entity is created for each non-anaphoric noun phrase in a sentence once that sentence has been disambiguated. A discourse entity is a designator for the entity or set of entities the noun phrase evokes in the discourse model of the speaker or hearer.
for noun phrases in a sentence (both for quantified noun phrases and noun phrases that are quantified over). If a noun phrase in one sentence is the antecedent for a pronoun in another, the pronoun is replaced with the antecedent’s discourse entity.

Pronouns have also been classified as bound variable or referential pronouns. The antecedent for a bound variable pronoun occurs in the same sentence as the pronoun and the meaning of the pronoun is represented as the variable bound by the operator associated with its antecedent. In contrast, the meaning of a referential pronoun is the discourse entity evoked by its antecedent. The bound versus referential dichotomy divides the world of pronouns in a slightly different way than the intersentential-intrasentential dichotomy. Pronouns with intersentential antecedents are typically referential. However, pronouns with intrasentential antecedents cannot always be classified as bound variable or referential pronouns. There is another category of pronouns which Evans [Eva80] dubs E-type pronouns. This type of pronoun appears to be a bound variable, but on closer inspection is not. Donkey sentences (originally noticed by Geach [Gea62]) can be used to demonstrate this difficulty. A typical donkey sentence is: *Every miner who owns donkey beats it.* Given that *every miner* has scope over *a donkey*, the indefinite cannot be referential and the existential operator is blocked from binding the pronoun because of the scope island; quantified noun phrases embedded in a relative clause attached to a noun phrase cannot bind pronouns outside of the relative clause environment. However, *a donkey* can be the antecedent for the pronoun. Donkey sentences provide evidence that all of the following cannot be simultaneously true (adapted from Heim [Hei82], p. 102):

1. Indefinites should be represented using existential quantifiers.
2. Indefinites obey the same scope-island restriction as universals.
3. Pronouns are either bound variables or referential.

Many researchers have attacked one or more of these assumptions. We prefer, for simplicity, to question only the third. We assume that indefinites are existentially quantified and that scope island constraints hold. These two assumptions will allow us to build a simpler logical form for a sentence. Later, in section 5, we will discuss a way to handle donkey sentences without requiring the pronoun to be bound by the existential operator.

Consider also how pronouns behave in verb phrase ellipsis. To signal verb phrase ellipsis, the full verb phrase is replaced with an auxiliary, as in the second sentence in 4. A sentence with verb phrase ellipsis is also called an *elided sentence*.

**Example 4**

*Trigger Sentence:* Fred *loves his wife.*
*Elided Sentence:* George *does too.*

*Meanings:*  
1. George loves Fred’s wife.  
   *(strict meaning)*  
2. George loves George’s wife.  
   *(sloppy meaning)*

---

4The only exception are pronouns like those in *paycheck sentences* (first noticed by Karttunen [Kar69]). Consider the sentence *Fred gave his paycheck to his wife. George gave it to his mistress.* The pronoun it is not referential. For that matter, it is not bound. The pronoun seems to take *his paycheck* as its antecedent where the pronoun is instantiated to a different individual than in the original sentence.

5This is related to the Complex Noun Phrase Constraint introduced by Ross [Ros67] which prevents wh-movement out of a relative clause attached to a noun phrase.
An elided sentence has little meaning independent of the first sentence, called a trigger sentence. In this example, the index on Fred and his indicates that they are co-referential. Given that the antecedent for his is the subject of the trigger sentence, and the meaning of the elided verb phrase depends on the meaning assigned to the trigger verb phrase, then the elided sentence is ambiguous. This ambiguity complicates the task of providing representations for pronouns. Example 4 demonstrates that the representation of pronouns is crucial for handling verb phrase ellipsis. Though the meaning of the elided sentence is ambiguous, it cannot mean George loves some other person's wife (other than Fred's or George's). The meaning assigned to the trigger verb phrase limits the possible meanings of the elided sentence, providing evidence that the meaning of an elided verb phrase should be derived from the possible trigger verb phrase representations. Hence, we must be prepared to provide two ways for a pronoun in a trigger sentence to refer to a syntactic subject.

Because we are concerned with building a computational model for language understanding, we develop a representation to capture the different meanings of a pronoun and to obey our computational constraints. Since pronouns have a range of behaviors between variables on the one hand and constants on the other, the initial logical form for a pronoun must be compatible with both extremes (to model the range of pronoun behaviors and to be consistent with the compactness and formal consistency constraints). Hence, we provide a composite representation for a pronoun, one that is compatible with any possible antecedent it can have given its position in a sentence.

3.2 Pronouns: An Initial Representation

Pronouns in a sentence are represented as part of the process of providing logical form for that sentence. So before we introduce the logical form for pronouns, we briefly describe the logical form for the rest of a sentence. A sentence is represented as a predicate-argument structure, with subjects lambda abstracted to handle verb phrase ellipsis (following Sag [Sag76], Williams [Wil77], Webber [Web78], and Partee and Bach [PB81]). By lambda abstracting syntactic subjects in logical form, a pronoun whose antecedent is a syntactic subject can refer to that subject in two different ways, either indirectly by using the subject's lambda variable or directly by using a value depending on the type of the subject noun phrase. The logical roles of all noun phrases in a sentence are indicated by position in logical form (logical subject first, logical object second, logical indirect object third, etc.). Following Webber [Web78], we represent universal noun phrases as universally quantified and restricted variables (as in 5a) and existentially quantified noun phrases as existentially quantified and restricted variables (as in 5b).

Example 5
a. Sentence: Every man is happy.
   Representation: ∀x: (man x) (happy x)
   Meaning: ∀x (if (man x) (happy x))

b. Sentence: A man is happy.
   Representation: ∃x: (man x) (happy x)
   Meaning: ∃x (and (man x) (happy x))

The colon between the quantifier and its restriction is syntactic sugar which expands differently depending on the type of the quantifier. Compare the meaning for the universal restriction, shown on
the third line of 5a, with the meaning for the existential, shown on the third line of 5b. Quantifier
scoping is handled in the same way as in Allen [All87] and Schubert and Pelletier [SP84]. Initially,
quantifiers are placed in the predicate-argument structure for the sentence (except for subjects, which
are necessarily abstracted⁶). Later, when information becomes available for making scoping decisions,
quantifier scoping is indicated using a method similar to Allen's [All87] (to be introduced in section
5). In this section, we do not introduce a general representation for definite noun phrases (see section
4), but only provide representations for proper nouns and possessive noun phrases. Possessive noun
phrases are represented as functions of the possessive nouns (following Webber [Web78]). Proper nouns
are represented as skolem constants (i.e., skolem functions without arguments).

The logical form representation for a pronoun must be compatible with our computational
constraints. To be consistent with the modularity constraint, we develop a representation for pronouns
which is generated before their antecedents are known. By providing representations for pronouns without
utilizing the pragmatic and contextual information needed to select antecedents, we are able to
divide the labor of sentence comprehension into the independent modules indicated in figure 1. To obey
the formal consistency and compactness constraints, we develop a representation consistent with the
ways pronouns can act given their position in a sentence. Because of the range of behaviors pronouns
can adopt, we represent them as functions in logical form⁷. A pronoun function is reminiscent of a
skolem function. However, a pronoun function is a composite representation. It limits the possible
antecedents for the pronoun, though it does not commit to one in particular. Each pronoun function
must have a unique name (supplied by adding a unique number to the pronoun), and its argument list
must be specified using only syntax and sentence-level semantics to avoid violating our computational
constraints.

A pronoun should be represented as a function of all the variables corresponding to quantified noun
phrases that can affect its meaning. The argument list should also contain the variables of lambda
operators that have scope over the position the pronoun function will fill in logical form in order to
provide sloppy readings of elided sentences. The quantified and lambda variables are chosen because
they are representations for possible antecedents for the pronoun or can affect the meanings other non-
quantified antecedents (e.g., pronouns and definites). However, by concentrating on variables of possible
antecedents, we automatically include those variables that affect potential non-quantified antecedents.

In English, not every quantified noun phrase in a sentence can be the antecedent for a pronoun in
the sentence. Consider, for example, *He loves every man*, in which the antecedent for *he* cannot be *every
man*. It is possible to determine which noun phrases can bind a pronoun in a particular sentence by
using only syntactic and sentence-level information. To do so, we adapt Reinhart's [Rei83] c-command
(or constituent-command), which is a relation between nodes in a parse tree.

⁶Abstraction of a quantified subject does not imply that it must have scope over quantifiers placed in the lambda
function corresponding to the verb phrase.

⁷Our representation of pronouns is similar in spirit to the pronoun representation in Charniak and McDermott [CM85].
By representing pronouns as unique skolem constants, they provide a representation for the basic logical structure of the
sentence before pronoun resolution is carried out; a division consistent with our compactness and modularity constraints.
However, to obey the formal consistency constraint, a pronoun's initial representation must be compatible with all the
ways that pronoun can act. Because a constant is not compatible with a variable, their representation of pronouns violates
formal consistency.
Node A c( constituent)-commands node B iff the branching node \(a_1\) most immediately dominating A either dominates B or is immediately dominated by a node \(a_2\) which dominates B, and \(a_2\) is of the same category type as \(a_1\). (p. 23)

For example, consider the parse tree for the sentence *Every man who saw every boy kicked his dog*, shown in Figure 2. NP1 c-commands *his*, but NP2 and NP3 do not. Reinhart claims that a pronoun can be bound by a noun phrase if only if the noun phrase c-commands the pronoun. Hence, *every man who saw every boy* can bind *his*, but *every boy* cannot bind *his*. Unfortunately, c-command (as defined above) incorrectly predicts that certain embedded quantified noun phrases cannot bind pronouns in the matrix sentence. For example, the c-command rule does not allow *every man’s* to bind *him* in *Every man’s mother loves him*, even though the universal variable provides a reasonable meaning for the pronoun. Similarly, even though *each candidate* does not c-command *him* in *A friend of each candidate supported him*, the universal noun phrase can bind the pronoun. We assume that when a quantified noun phrase is embedded in a relative clause attached to a noun phrase, then that quantifier cannot bind a pronoun outside of the relative even if the containing noun phrase c-commands the pronoun. For example, in *Every man who saw every boy kicked his dog*, *every boy* cannot be the antecedent of *his*. On the other hand, a quantified noun phrase which is not embedded in a relative clause but is embedded in a noun phrase may bind a pronoun if the containing noun phrase c-commands the pronoun. Hence, we adapt Reinhart’s binding rule\(^8\) to allow quantified noun phrases to bind a pronoun if they c-command the pronoun or are embedded in another c-commanding noun phrase but not contained in a relative clause.

A composite representation for a pronoun is provided once its sentence’s parse tree is available. A pronoun is represented as a uniquely-named function of all the variables corresponding to the lambda operators which have scope over the pronoun function in logical form and any non-subject quantified variables corresponding to noun phrases that can bind the pronoun given our modification of Reinhart’s [Rei83] binding rule. Because a quantified subject’s variable is abstracted by the sentence’s lambda

---

\(^8\)We could have also adapted the binding constraints found in Bach and Partee [BP80].
operator, the lambda variable subsumes the quantified variable making it the only subject variable to include in the argument list for a pronoun function.

Now that the initial representation of a pronoun has been specified, consider a series of examples showing the representation of pronouns before their antecedents are known. Consider example 6.

Example 6
Fred loves himself.
Fred₂ ≔ λ(x)(love x (himself₁ x))

The subject Fred is represented as the skolem constant Fred₂. The verb phrase is represented as the lambda function, λ(x)(love x (himself₁ x)). To create a representation for the entire sentence, we apply the subject to the lambda function; this is indicated by placing the subject to the left of the function. The logical subject fills the first slot after the predicate in the verb phrase and the logical object fills the second. Since the sentence in example 6 contains no universal or indefinite noun phrases, the pronoun function representing himself is simply a function of the lambda variable x. The name of the pronoun function is created by concatenating a unique integer onto the end of the pronoun string. In example 7, the representation of the pronoun is affected by universal noun phrases in the sentence.

Example 7
Every man showed every boy his picture.
∀x : (man x), λ(y)(show y (picture-of (his₂ y z))) [∀z : (boy z) z]

The syntactic subject of the sentence is universally quantified, as is the indirect object, and the logical direct object his picture is represented as a function of the pronoun. The reader should note that there is no order implied by quantifiers in our logical form (following Allen [All87]) and that the pronoun his is represented as a function of the subject’s lambda variable plus the universal variable for every boy. As discussed above, the subject’s lambda variable subsumes its universal variable. Next, consider example 8.

Example 8
Fred believes he must speak to every woman.
Fred₂ ≔ λ(x)(believe x [(the₁ x), λ(z)(speak z [∀y : (woman y) y)])

Though the sentence in example 8 contains a universal noun phrase, he is represented as a function of the lambda variable x alone. The pronoun function’s argument list does not include the variable y because every woman neither c-commands nor is embedded in a noun phrase that c-commands the pronoun. Finally, consider an example of a sentence containing only definite noun phrases.

Example 9
Fred showed his mother her picture.
Fred₂ ≔ λ(x)(show x (picture-of (her₁ x)) (mother-of (his₂ x)))

Both of the pronouns in example 9 are represented as functions of the lambda variable x.

²Though it is more traditional to apply the term to the right of the function, we chose this method to make it easier to see the correspondence between the sentence and its representation (following Sag76).
3.3 Pronouns: Updating Logical Form

Before the final meaning of the sentence can be given, the antecedent for a pronoun must be determined and made explicit in the logical form. Though the process of determining antecedents for pronouns is beyond the scope of this paper, when a pronoun's antecedent is known, the logical form containing it must be updated in a way compatible with its initial representation (because of the formal consistency constraint). We update the logical form by equating a pronoun function with a value depending on the type of its antecedent. In this way, a pronoun function, like a chameleon, adopts the behavior of its antecedent. Depending on the type and location of a pronoun's antecedent, the pronoun function is equated with various values. If it is a universal or indefinite noun phrase in the same sentence, then the pronoun function is set equal to (or replaced by) a quantified variable. If it is a noun phrase represented as a function (i.e., a pronoun or definite) in the same sentence, then the pronoun function is set equal to that function. If it is the syntactic subject of a sentence, then the function is equated with either the subject's lambda variable or something depending on the subject's type. If a pronoun is referentially dependent on a noun phrase in a different sentence or on some non-linguistic entity, then the pronoun function is equated with a discourse entity.

To augment logical form with antecedent information without creating ill-formed logical forms (i.e., logical forms with unbound variables), we assert equality statements in the lambda environment of the pronoun function and limit the types of updates allowed. Pronoun functions can only be equated with values formally consistent with their initial meaning. They can be immediately equated with constants, any of their arguments, any variables that are abstracted by lambda operators whose variables are arguments, and any pronoun or possessive functions whose argument list are compatible with the pronoun function. Some updates cannot be made until more is known about the meaning of some noun phrases, as we will show in later sections. However, no update is allowed unless it is formally consistent with the initial representation of the pronoun function. In this way, we use our initial pronoun representation to constrain the possible antecedents for the pronoun.

Consider some examples of how logical form is augmented following pronoun resolution. Suppose, for example, that we decide that the antecedent for his in example 7 is every boy, then the logical form for the sentence is modified as follows:

Example 10

Every man, showed every boy, his picture.
\[ \forall x: (\text{man } x) \ x, \lambda(y)(\text{show } y \text{ (picture-of (his}_2 y z)) [\forall z: (\text{boy } z) \ z]) \]
\[ (= (\text{his}_2 y z) z) \]
Simplification:
\[ \forall x: (\text{man } x) \ x, \lambda(y)(\text{show } y \text{ (picture-of } z) [\forall z: (\text{boy } z) \ z]) \]

To indicate the fact that the antecedent for his is every boy, the pronoun function (his\(_2\) y z) is equated with the universally quantified variable z. Notice that the equality statement is placed in the environment of the \(\lambda(y)\) operator. If we had placed it outside of this environment, the variable \(y\) would have been unbound. The variable \(z\) is compatible with the pronoun's initial representation because we are limiting the a function of \(y\) and \(z\) to be the identity function on \(z\). Once the equality statement is asserted, we can simplify the logical form as shown in 10.

Next, consider how the representation in example 6 would be augmented after pronoun resolution (shown in 11).
Example 11
Fred, loves himself.
Fred, λ(x)(and (love x (himself; x))
(or (= (himself; x) x) (= (himself; x) Fred)))

Given that the antecedent for himself is the subject Fred, the pronoun can refer to it directly or indirectly. Hence, the pronoun function is equated with either the lambda variable x or Fred. Notice that no simplification of the logical form is possible until one of the alternatives is chosen. By allowing a disjunction of equality statements, we can compactly represent the ambiguous ways that pronouns refer to syntactic subjects. In particular, if there are n pronouns whose antecedents are the subject of the sentence, we can specify this ambiguity with O(n) updates (compared with 2^n different representations for the sentence). Because of the compact representation, we can put off a decision about the pronoun’s intended meaning until we process information supporting a particular decision.

Finally, consider how the logical form in example 9 is updated following pronoun resolution, shown in example 12.

Example 12
Fred, showed (his, mother), her, picture.
Fred, λ(x)(and (show x (picture-of (her; x)) (mother-of (his; x)))
(or (= (her; x) (mother-of (his; x)))
(or (= (his; x) x) (= (his; x) Fred)))

Assume that the antecedent for his is Fred. Since Fred is the subject, the pronoun function (his; x) is equated with Fred or x. Both updates are compatible with the initial pronoun function. Assume that the antecedent of her is his mother. Because (mother-of (his; x)) is compatible with the pronoun function (her; x), it is equated with the pronoun function.

3.4 Verb Phrase Ellipsis

In this section, we discuss how to handle examples of verb phrase ellipsis, using example 4 to illustrate our approach.

Example 4
Trigger Sentence: Fred, loves his, wife.
Elided Sentence: George, does too.
Meanings: 1. George loves Fred’s wife. (strict meaning)
2. George loves George’s wife. (sloppy meaning)

Because we assume the meaning of an elided verb phrase is constrained by the meaning of the trigger verb phrase, the trigger’s meaning must be fixed before we provide the meaning of the elided verb phrase. The trigger sentence in example 4 is initially represented as shown in 13.

Example 13
Fred loves his wife.
Fred, λ(x)(love x (wife-of (his; x)))

Before this representation of the trigger sentence is used to provide the meaning of the elided verb phrase in example 4, we must locate the antecedent for his; otherwise, the final meaning of the pronoun function in the trigger cannot limit the meaning of the pronoun in the elided sentence. Assume that the antecedent for his is Fred. Because Fred is the subject, the pronoun his can refer to it either directly

12
(by equating the pronoun function with the subject’s skolem constant) or indirectly (by equating it with the subject’s lambda variable), so we augment the logical form as shown in 14.

Example 14
Fred, loves his, wife.
Fred$_{22}$, $\lambda(x)(\text{and} \ (\text{love } x \ (\text{wife-of} \ (\text{his;} \ x)))$
(or $\ (= \ (\text{his;} \ x) \ x) \ (= \ (\text{his;} \ x) \ Fred_{22}))$

The disjunction stores the two possible representations of the trigger sentence given the anaphora decision.

Next, consider the initial representation of the elided sentence in 4, for which we introduce a placeholder for the missing verb phrase.

Example 15
George does too.
Georges$_3$, $\lambda(y)(\text{Dummy$_2$} \ y)$

Before replacing the dummy lambda function with its intended meaning, we must locate the trigger sentence (assume 14 is the trigger sentence’s representation) and select a single meaning for it by choosing a single meaning for the pronoun function. How to choose between the two meanings is a topic beyond the scope of this work, but we demonstrate that for each choice, we provide a possible meaning for the elided sentence.

If we assume that the pronoun his refers indirectly to the subject of the trigger sentence, we select the first disjunct in 14. This choice allows us to provide the sloppy reading of the elided sentence, as shown in 16.

Example 16
Trigger Sentence Representation:
Fred$_{22}$, $\lambda(x)(\text{and} \ (\text{love } x \ (\text{wife-of} \ (\text{his;} \ x))) \ (= \ (\text{his;} \ x) \ x))$
Elided Sentence Representation:
Georges$_3$, $\lambda(x)(\text{and} \ (\text{love } x \ (\text{wife-of} \ (\text{his;} \ x))) \ (= \ (\text{his;} \ x) \ x))$

On the other hand, if the pronoun his refers directly to the subject Fred, the second update is chosen, allowing us to derive the strict reading of the elided sentence, as shown in 17.

Example 17
Trigger Sentence Representation:
Fred$_{22}$, $\lambda(x)(\text{and} \ (\text{love } x \ (\text{wife-of} \ (\text{his;} \ x))) \ (= \ (\text{his;} \ x) \ Fred_{22}))$
Elided Sentence Representation:
Georges$_3$, $\lambda(x)(\text{and} \ (\text{love } x \ (\text{wife-of} \ (\text{his;} \ x))) \ (= \ (\text{his;} \ x) \ Fred_{22}))$

There is no way to determine whether a particular sentence will be a trigger sentence until an elided sentence is processed, so we must treat all sentences as possible trigger sentences. Later, when an elided sentence is detected, we must locate and disambiguate its trigger sentence. Pragmatic and contextual information is needed to select trigger sentences, to choose antecedents for pronouns, and to select a single meaning for a pronoun function whose antecedent is a syntactic subject.

We have described a representation of pronouns in logical form. The logical form for a pronoun is provided using only syntactic and sentence-level information, is compatible with the possible meanings for that pronoun given its position in a sentence, and can be updated without violating the formal consistency constraint. Once the antecedent of the pronoun is determined, its function is equated with
a value depending on the type of the antecedent. For example, when the antecedent of a pronoun is a possessive noun phrase, the pronoun function is equated with the function representing that noun phrase. We also provide a compact way to represent the sloppy identity ambiguity. Hence, our logical form for pronouns satisfies our computational constraints. Next, we discuss the representation of singular definite noun phrases, which are similar to pronouns though their structure is more complex.

4 Singular Definite Noun Phrases

In this section, we develop a logical form representation for singular definite noun phrases. As for pronouns, we wish to obey our computational constraints while providing a good model of definite behavior. First, consider the behaviors we cover.

4.1 Definites: Linguistic Evidence

Like pronouns, definite noun phrases can be anaphoric. Anaphoric definites either depend on linguistic antecedents or denote salient individuals in the environment of the speaker/hearer. An anaphoric definite's antecedent can be found in previous sentences as in Fred saw (his; cat). The cat was chasing a mouse, or within the same sentence as in Every boy who loves (his; cat) takes care of the animal. In the first example, the antecedent for the cat is his cat, found in the previous sentence, hence, the cat adopts the discourse entity assigned to his cat. In the second, the animal cannot take a discourse entity as its meaning because its antecedent is his cat, which does not denote a particular cat.

Definites, unlike pronouns, can have a complex syntactic structure. Noun phrases embedded in a definite noun phrase affect the definite's meaning. For example, consider the effect of pronouns embedded in definite noun phrases. While simple non-anaphoric definites seem to act like constants when contained in sentences with universal noun phrases, as in Every boy loves the woman, definite noun phrases with embedded pronouns often cannot be described as constants, as in Every boy loves his mother. The meaning of his mother depends on how the pronoun is resolved. If the antecedent for his is found in another sentence, then his mother could be represented as a constant, but if every boy is the antecedent for his, then the universal quantifier corresponding to every boy distributes over his mother. When a quantifier distributes over a definite, the definite denotes different entities depending on the values assigned to the quantified variable. Any pronoun embedded in a definite noun phrase can affect the definite’s meaning, whether it is a possessive or is contained in a prepositional phrase or in a relative clause attached to the definite.

A possessive quantifier in a definite noun phrase distributes over the noun phrase, preventing it from acting like a constant, as in Every man's mother loves him. The possessive quantifier can also bind pronouns the definite noun phrase c-commands. Quantified noun phrases contained in a prepositional phrase attached to a definite noun phrase can also distribute over the definite, as in The head of every public authority in New York is rich, though the meaning of the definite noun phrase is ambiguous. If the universal distributes over the head of every public authority in New York, then its denotation depends on which public authority is considered. But if the universal does not distribute over the definite, then there is one particular person who heads all of the public authorities. Our initial representation for the
head of every public authority must be compatible with either possibility.

Not all embedded quantified noun phrases can distribute over a definite. Quantified noun phrases embedded in relative clauses attached to a definite noun phrase seem unable to distribute over that noun phrase. This constraint prevents every boy from quantifying over the child who cares for every man, so the definite can only denote one particular child. Universal noun phrases that cannot distribute over a definite noun phrase are also unable to bind a pronoun outside that phrase, as noted by May [May85] and Roberts [Rob87].

We must also consider the behavior of a definite noun phrase in verb phrase ellipsis. The meaning of a definite noun phrase is ambiguous whenever it contains a pronoun whose antecedent is the subject of the sentence, as in example 410.

Example 4
Trigger Sentence: Fred loves his wife.
Elided Sentence: George does too.
Meanings: 1. George loves Fred's wife. (strict meaning)
2. George loves George's wife. (sloppy meaning)

We must also provide a good representation for a definite subject, one that will account for the differences between universal subjects and definite subjects. Consider examples 18 and 19.

Example 18
Every postman saw his dog.
Every policeman did too.
Meanings:
Every policeman saw his own dog. (sloppy reading only)

Example 19
The postman saw his dog.
The policeman did too.
Meanings:
a. The policeman saw the postman's dog. (strict reading)
b. The policeman saw his own dog. (sloppy reading)

Universal quantifiers cannot bind across sentences, so the only possible meaning for the elided sentence in 18 is the sloppy one. However, the elided sentence in 19 has a sloppy and a strict meaning. If we choose a quantifier to represent a definite subject, we would have to alter its binding properties to allow it to bind across sentences.

The meaning of a definite noun phrase is affected by its structure, whether it contains pronouns, and whether or not it is used anaphorically. If used anaphorically, it behaves in a way consistent with its antecedent, just like a pronoun. If it contains pronouns, then its meaning depends on the antecedents chosen for those pronouns. If it contains quantified noun phrases not also contained in a relative clause, then those embedded noun phrases can distribute over the definite. In the remainder of this section, we introduce our logical form representation for definites, describe the ways this logical form is updated once ambiguity is resolved, discuss how the representation is used to handle definites in verb phrase ellipsis, and summarize how well our approach obeys our constraints.

10Or whenever it contains an embedded indefinite. We consider sloppy indefinites in the section 5.
4.2 Definite Noun Phrases: Initial Representation

In this section, we develop a representation for definites in logical form. To be consistent with the modularity constraint, we provide an initial representation for a definite noun phrase that can be generated before determining the antecedents for any embedded pronouns or before locating the definite’s antecedent (if it is anaphoric). To obey the compactness and formal consistency constraints, we initially represent a definite so it is consistent with any possible behaviors. As more information becomes available about the meaning of the definite noun phrase, we update logical form in a way compatible with its initial representation.

We represent a definite as a named function of all the variables associated with operators that can affect its meaning. This representation satisfies our constraints by combining the advantages of definite descriptions (discussed in section 7.2.1) with the functional notation we introduced to represent pronouns. Each definite function is defined by a unique name (i.e., def with a unique integer appended to it), a list of arguments, and a restriction. The restriction of a definite function is derived from the words following the determiner. The argument list of the function consists of the variables associated with lambda operators that have scope over its position, any variables associated with non-subject quantified noun phrases that could bind a pronoun in that position, and any quantified variables associated with embedded quantified noun phrases that are not also embedded in a relative clause\(^{11}\). Because a definite function has a unique name, we can differentiate two occurrences of the same definite noun phrase.

Consider the initial representation of the sentence in example 20 containing a definite noun phrase.

**Example 20**
Every man showed every boy his picture.

\[\forall x: (\text{man } x) \quad x, \lambda(y)(\text{show } y ((\text{def}_1 y z) \land (\text{picture } (\text{def}_1 y z))) (\text{possess } (\text{his}_2 y z) (\text{def}_1 y z))) \]

\[\forall z: (\text{boy } z) z] \]

The representation of this sentence is very similar to example 7 except for the representation of the definite noun phrase. Notice that his picture is represented as a function called \(\text{def}_1\). The restriction of the function is the conjunction of statements following the vertical bar. The vertical bar is syntactic sugar and is expanded like the colon in an existential’s restriction but not until the definite’s final meaning is determined. The argument list of the function consists of the variables \(y\) and \(z\), just like the pronoun his. Anything that affects the meaning of the pronoun his also affects the meaning of the definite.

To provide a good initial representation for a definite noun phrase, we must also account for embedded quantified noun phrases that can distribute over definites. In *Every man’s mother loves him*, the noun phrase *every man’s mother* does not denote a single mother; *every man* distributes over the definite noun phrase. Assuming a possessive quantified noun phrase embedded in a definite noun phrase always distributes over the definite, the variable associated with the quantifier is included in the argument list of the definite function and the quantifier is pulled out of the restriction of the definite function to distribute over it. The initial representation for *every man’s mother* is shown in example 21.

\(^{11}\)We should also add that a sententially attached PP with a quantified object can quantify over a definite as well (as in, *In every car, the driver turned the steering wheel.* This sentence is tricky because we seem to be attaching the PP to both of the NPs while leaving the quantifier to distribute over both definites).

\(^{12}\)As in the representation of pronouns, we omit the variable \(x\) from the argument list because the lambda operator for \(y\) abstracts \(x\), so \(y\) is the more general argument.
Example 21
Every man's mother loves him.

\[ \forall x: (\text{man} \, x) \, ((\text{def} \, x) \, | \, (\text{and} \, (\text{mother} \, (\text{def} \, x)) \, (\text{possess} \, x \, (\text{def} \, x))) \, , \lambda(y)(\text{love} \, y \, (\text{him} \, \, x \, y)) \]

Notice that \( \text{def} \, x \) is a function of the universal variable \( x \) and that the universal operator associated with \( x \) distributes over that function and can therefore bind the pronoun \( \text{him} \).

On the other hand, quantified noun phrases contained in a relative clause attached to a definite noun phrase cannot distribute over the definite. Such quantified noun phrases are prevented from moving out of the relative clause to distribute over the definite, and so the definite acts like a simple definite noun phrase (assuming the restriction contains no embedded pronouns). Consider the initial representation for the child who cares for every man, shown in 22.

Example 22
The child who cares for every man visits him.

\[ ((\text{def}) \, | \, (\text{and} \, (\text{child} \, (\text{def}))) \, ((\text{def}) , \lambda(x)(\text{care} \, x \, (\text{for} \, (\forall y: (\text{man} \, y) \, y))) \, ) \, , \lambda(z)(\text{visit} \, z \, (\text{him} \, \, z)) \]

Notice that the argument list for the definite function does not contain the variable \( y \) and the pronoun \( \text{him} \) is represented as a function of \( z \). Hence, the antecedent for the pronoun \( \text{him} \) cannot be \textit{every man} without violating formal consistency.

If a quantified noun phrase is contained in a prepositional phrase attached to a definite noun phrase, the representation of the definite differs from both the possessive and relative clause cases. Consider The head of every public authority in New York is rich. The meaning of the definite subject is ambiguous and so its initial representation must be compatible with its two possible meanings. When a quantified noun phrase is the object of a preposition attached to a definite noun phrase, we include its variable in the argument list of the initial definite function and place the quantifier inside the restriction of the definite function. Hence, the initial representation for the head of every public authority is shown in 23.

Example 23
The head of every public authority in New York is rich.

\[ ((\text{def}) \, | \, (\text{head-of} \, [\forall x: (\text{and} \, (\text{public-authority} \, x) \, (\text{in} \, x \, \text{New York})) \, x]) \, (\text{def}) \, ) \, , \lambda(y)(\text{rich} \, y) \]

Notice that the quantifier is placed inside the restriction of \( \text{def} \), and the variable \( x \) is placed in the argument list (for the semantics of such a function, see the Appendix). Later, when we decide whether or not the quantifier distributes over the definite, the initial representation is limited, as discussed in the next section.

Because a definite function is a composite representation for all possible meanings of a definite noun phrase, we must restrict the function in certain ways before the final meaning is available. The initial representation of a definite places an upper and lower bound on the definite's behavior. The lower bound is a constant, while the upper bound is the initial representation. These bounds must be tightened to settle on the intended meaning of the definite. We provide two methods to pinpoint a definite function.

---

13 Notice that we do not provide an explicit representation for \( \text{who} \). Instead, we simply represent it by using the relative head's representation. In 22, \( \text{who} \) is represented as a definite function. If the relative head was quantified, we would have represented the relative pronoun using the variable associated with the quantifier.
4.3 Definite Noun Phrases: Two Ways To Update Initial Representation

If a definite is used anaphorically, it is equated with some value depending on its antecedent, just like a pronoun function. For example, if a possible antecedent for a definite noun phrase occurs in another sentence, we would equate the definite function with the antecedent’s discourse entity once the pronoun resolution module verifies that the discourse entity is compatible (in number and other possible features) with the definite noun phrase. If they are incompatible, then the update is not allowed.

Antecedents for definite noun phrases can also be found in the same sentence. The anaphora resolution module checks the possible intrasentential antecedents for formal consistency with the definite function and applies additional constraints to limit possible antecedents (e.g., a definite’s antecedent does not typically c-command it). Consider The owner of every dog is afraid of the animal. The initial representation of this sentence is shown in 24.

Example 24
The owner of every dog is afraid of the animal.

\[
\begin{align*}
 & \text{((def; x) | (and (owner (def; x)) (of (def; x) \[ \forall x: (\text{dog x}) x]))),} \\
 & \lambda(y)(\text{afraid-of y ((def}_2 \ x \ y) | (\text{animal (def}_2 \ x \ y)))))
\end{align*}
\]

Because every dog does not c-command the animal and the universal variable \( x \) is formally consistent with (def 2 x y), every dog is a reasonable antecedent for the animal. Hence, (def 2 x y) can be equated with the universal variable \( x \) (as shown in 25).

Example 25
(The owner of every dog) is afraid of the animal,

\[
\begin{align*}
 & \text{((def; x) | (and (owner (def; x)) (of (def; x) \[ \forall x: (\text{dog x}) x]))),} \\
 & \lambda(y)(\text{afraid-of y ((def}_2 \ x \ y) | (\text{animal (def}_2 \ x \ y))))) (= (\text{def}_2 \ x \ y) x))
\end{align*}
\]

This example would be very difficult for an approach that uses either definite descriptions or definite quantifiers.

The other way to pinpoint a definite function applies once we determine the antecedents for embedded pronouns and decide whether quantifiers corresponding to embedded quantified noun phrases (not also contained in relative clauses) should distribute over the definite. Consider the initial representation of the sentence in example 20. The definite function def 1 is a function of all of the variables that can potentially cause it to change. However, once we know the antecedent for its embedded pronoun, the argument list of the function should be limited. To limit the argument list, we make use of the insights gained from definite descriptions. Because of the uniqueness assumption, any definite description which does not contain variables bound by outside quantifiers acts like a constant. On the other hand, if a pronoun embedded in a definite description adopts the behavior of a universally quantified variable, then the definite description will change what it denotes depending on the instantiation of that variable. Hence, we conclude that a definite function should only change as a function of those variables bound by operators outside of its restriction (not including the variables in its original argument list). Once we determine the meanings of noun phrases in the definite function’s restriction and determine whether any of the embedded quantifiers distribute over the definite function, we limit the argument list to precisely those arguments that are bound by operators outside of the restriction by replacing the original function with a new function over those necessary arguments. By using this argument reduction constraint, we limit the initial composite representation of a definite noun phrase to its final meaning (given pronoun
and quantifier information).

Consider how we would limit the function \((\text{def}_1 \, y \, z)\) from example 20. If we decide that the antecedent of his is every boy, we update the logical form as shown in 26.

Example 26
Every man showed every boy, his, picture.
\[
\forall x: (\text{man } x) \, x, \, \lambda(y) (\text{show } y \, ((\text{def}_1 \, y \, z) \mid (\text{picture } (\text{def}_1 \, y) \\
\quad (\text{possess } (\text{his}_2 \, y \, z) \, (\text{def}_1 \, y \, z)) \\
\quad (= (\text{his}_2 \, y \, z) \, z))) \\
\left[ \forall z: (\text{boy } z) \, z \right]
\]
Then by applying the argument reduction constraint, we replace the function \((\text{def}_1 \, y \, z)\) by a function of \(z\) (since \((\text{his}_2 \, y \, z)\) is replaced with the variable \(z\)), as shown in 27.

Example 27
Every man showed every boy, his, picture.
\[
\forall x: (\text{man } x) \, x, \, \lambda(y) (\text{show } y \, ((\text{def}_1 \, y) \mid (\text{picture } (\text{def}_1 \, y) \\
\quad (\text{possess } (\text{his}_2 \, y \, z) \, (\text{def}_1 \, y \, z)) \\
\quad (= (\text{his}_2 \, y \, z) \, z))) \\
\left[ \forall z: (\text{boy } z) \, z \right]
\]
\[
\text{Simplification:}
\forall x: (\text{man } x) \, x, \, \lambda(y) (\text{show } y \, (\text{def}_3 \, z) \left[ \forall z: (\text{boy } z) \, z \right] \, (\text{picture } (\text{def}_3 \, z)) \, (\text{possess } z \, (\text{def}_3 \, z)))
\]
Because of the meanings of equality and the vertical bar in the restriction of the function, this representation is simplified as shown above.

Now consider how we update the initial representation of the sentence in example 23. If we decide every public authority in New York distributes over the definite function, then the universal quantifier \(\forall x\) is moved out of the restriction prior to applying the argument reduction constraint, as shown in 28.

Example 28
The head of every public authority in New York is rich.
\[
\forall x: (\text{and } (\text{public-authority } x) \, (\text{in } x \, \text{New York})) \, ((\text{def}_1 \, x) \mid (\text{head-of } x \, (\text{def}_1 \, x))), \, \lambda(y) (\text{rich } y)
\]
Because the variable \(x\) is free in the restriction of \((\text{def}_1 \, x)\), the function retains the variable in its argument list. On the other hand, if every public authority in New York does not distribute over the function, then the quantifier remains in the restriction, as shown in 29.

Example 29
The head of every public authority in New York is rich.
\[
((\text{def}_1 \, x) \mid (\text{head-of } [\forall x: (\text{and } (\text{public-authority } x) \, (\text{in } x \, \text{New York})) \, x] \, (\text{def}_1 \, x))), \, \lambda(y) (\text{rich } y)
\]
Because the restriction of \((\text{def}_1 \, x)\) contains no free variables, the argument reduction constraint replaces it with a function with no arguments, as shown in 30.

Example 30
The head of every public authority in New York is rich.
\[
(\text{and } ((\text{def}_1 \, x) \mid (\text{head-of } [\forall x: (\text{and } (\text{public-authority } x) \, (\text{in } x \, \text{New York})) \, x] \, (\text{def}_1 \, x))), \, \lambda(y) (\text{rich } y)
\]
\[
(= (\text{def}_1 \, x) \, (\text{def}_3)))
\]
Hence, our approach provides both of the possible meanings for the definite noun phrase the head of every public authority without using multiple initial representations (which would violate compactness).

Our solution is not without problems, however. The decision about whether a quantified noun phrase contained in a definite noun phrase distributes over the definite impacts the quantifier's ability to bind
a pronoun in the matrix sentence. Quantified objects of prepositions attached to a definite noun phrase can bind matrix pronouns only when they distribute over the definite (observed by Roberts [Rob87] and May [May85]). Consider the sentence The secretary of every spy keeps an eye on him. In this case, the noun phrase every spy can bind the pronoun him only when it has scope over the definite noun phrase, giving it a distributive reading. Does this make our representation of the pronoun him contingent on a quantifier scoping decision? We cannot determine whether a quantified object of a preposition attached to a noun phrase will distribute over that noun phrase using syntactic information alone. Hence, we must include the variable in the argument list of the pronoun, and then make certain that the pronoun function is not bound by the quantifier, unless it distributes over the noun phrase containing it. This requirement could be easily incorporated into pronoun resolution.

Our representation has several strengths. First, the representation provides useful constraints for limiting possible intrasentential antecedents for a definite, in addition to traditional constraints like number and gender agreement. Another strength is illustrated with the following example.

Example 31
Fred told the teacher who discussed every student with his mother to examine her educational history.
((def.) | (name (def.) Fred)),
  λ(x)(tell x ((def2 x) | (and (inst (def2 x) teacher)
  ((def2 x), λ(y)(discuss y [∀(z): (inst z student) z]
  (with ((def2 x y z) | (and (inst (def3 x y z) mother)
  (possess (his x y z)
  (def x y z))))))))
[((def2 x), λ(u)(examine u ((def3 x u) | (and (inst (def3 x u) ed-history)
  (possess (her x u) (def x u))))))]

What are the legal antecedents for her in this sentence? Certainly, the teacher is a fine candidate, but what about his mother. We cannot immediately determine whether his mother can be the antecedent for her because (her x w) is not immediately compatible with the representation for his mother (i.e., (def3 x y z)). Before we can assert that his mother is the antecedent for her we must pinpoint the meaning of that noun phrase by selecting the antecedent for his. Depending on the choice, the final meaning of his mother may or may not be accessible to the pronoun. If the antecedent for his is Fred or the teacher, then his mother can be the antecedent for her after the argument reduction constraint is applied. However, if the antecedent is every student, then his mother cannot be the antecedent for her because every student cannot have scope over her. The c-command relation does not accurately predict when de/finite antecedents are accessible for anaphoric expressions. This is not surprising, given the fact that the accessibility of a definite antecedent depends on its final meaning which cannot be determined using only syntactic information.

One shortcoming of our approach is that we must have a single parse tree for a sentence to provide a single logical form for that sentence. For example, Fred saw the bird with his binoculars would yield two parse trees and hence two logical forms. One possible solution to this problem is to store partial

---

14Roberts [Rob87] modifies the definition of c-command to allow a prepositional-phrase-attached quantified noun phrase to optionally c-command the same noun phrases as the containing noun phrase.

15Strictly speaking, universal noun phrases cannot bind across sentences. However, speakers sometimes allow a universal to be the antecedent for a singular pronoun outside of its scope. Such pronouns are not usually understood as giving a bound variable reading. See Webber [Web78] for a discussion of this issue.
logical forms in a parse forest, and then as more information is processed, to restrict the parse forest and logical forms to a single tree containing one logical form.

### 4.4 Verb Phrase Ellipsis

To handle verb phrase ellipsis, we must limit the meaning of a definite function contained in the verb phrase of a trigger sentence before providing the interpretation of an elided sentence. Unless we constrain the initial representation of a definite in the trigger verb phrase before deriving the meaning of the elided sentence, we cannot provide strict meanings for the elided sentence. To see this, consider the meaning of an elided sentence whose trigger sentence contains a definite noun phrase without embedded pronouns.

**Example 32**

Fred saw the dog.
George did too.

**Meanings:**
George saw the same dog that Fred saw.

Notice that *the dog* must denote the same dog in the trigger and elided sentences. Consider the initial representation of the trigger sentence, shown in 33.

**Example 33**

Fred saw the dog.

\[
((\text{def}_1) \mid (\text{name} (\text{def}_1) \text{Fred})), \lambda(x)(\text{see} x ((\text{def}_2) x) (\text{dog} (\text{def}_2 x)))
\]

Though both *Fred* and *the dog* are represented as functions, \(\text{def}_1\) is a function with no arguments and must act like a constant, but \(\text{def}_2\) is a function of the lambda variable \(x\).

Suppose that we derive the meaning of the elided sentence using the representation of the verb phrase shown in 33, without limiting the function representing *the dog*.

**Example 34**

George did too. (George saw a different dog than Fred saw.)

\[
((\text{def}_3) \mid (\text{name} (\text{def}_3) \text{George})), \lambda(x)(\text{see} x ((\text{def}_2) x) (\text{dog} (\text{def}_2 x)))
\]

The meaning expressed in 34 is not possible for the elided sentence. Before deriving the meaning of the elided sentence from the representation of its trigger sentence, we must apply the argument reduction constraint to all definite functions in the verb phrase. Assuming that *the dog* is non-anaphoric, the argument list of \(\text{def}_2\) must be reduced before the meaning of the elided sentence is derived. Because the restriction of \((\text{def}_2 x)\) contains no free variables, it is replaced by a function with an empty argument list, as shown in 35.

**Example 35**

Fred saw the dog.

\[
((\text{def}_1) \mid (\text{name} (\text{def}_1) \text{Fred})), \lambda(x)(\text{and} x ((\text{def}_2) x) (\text{dog} (\text{def}_2 x)) ((\text{def}_3)))
\]

Simplification:

\[
((\text{def}_1) \mid (\text{name} (\text{def}_1) \text{Fred})), \lambda(x)(\text{and} x (\text{def}_1)) (\text{dog} (\text{def}_2))
\]

Now that the meaning of the definite function is pinpointed, we can derive the meaning of the elided sentence. Using the representation of the verb phrase in 35, the elided sentence receives the meaning shown in 36.
Example 36
George did too. (George saw the same dog as Fred did.)

\((\text{def}) | (\text{name (def}_{4} \text{ George)}), \lambda(x)(\text{and (see x ((def}_{2} x) | (\text{dog (def}_{2} x)))) (= (\text{def}_{2} x) (\text{def}_{3})))\)

Simplification:
\((\text{def}_{4} | (\text{name (def}_{4} \text{ George)}), \lambda(x)(\text{and (see x (def}_{3}) (\text{dog (def}_{3}))\))\)

Because the final meaning for the dog is (def\_3) in 35 and 36, it must denote the same dog in both.

Consider also how we handle example 4.

Example 4
Trigger Sentence: Fred loves his wife.
Elided Sentence: George does too.
Meanings: 1. George loves Fred's wife. (strict meaning)
2. George loves George's wife. (sloppy meaning)

The initial representation of the trigger sentence is shown in 37.

Example 37
Fred loves his wife.

\((\text{def}_{1} | (\text{name (def}_{1} \text{ Fred)}), \lambda(x)(\text{love x ((def}_{2} x) | (\text{and (wife (def}_{2} x))

\begin{align*}
& (\text{possess (his}_{3} x) (\text{def}_{2} x)))
& (\text{or ( (his}_{3} x) (\text{def}_{1}))
& (= (\text{his}_{3} x) (\text{x}) ))
\end{align*}\)

Suppose the pronoun resolution module determines that the antecedent for his is Fred, then the logical form for the trigger sentence is modified, as indicated in 38.

Example 38
Fred loves his wife.

\((\text{def}_{1} | (\text{name (def}_{1} \text{ Fred)}), \lambda(x)(\text{love x ((def}_{2} x) | (\text{and (wife (def}_{2} x))

\begin{align*}
& (\text{possess (his}_{3} x) (\text{def}_{2} x))
& (\text{or ( (his}_{3} x) (\text{def}_{1}))
& (= (\text{his}_{3} x) (\text{x}) ))
\end{align*}\)

Depending on the selected meaning for the pronoun his, we can derive two different readings of the elided sentence.

If his refers indirectly to Fred, the intended meaning for the trigger sentence is shown in 39.

Example 39
Fred loves his wife.

\((\text{def}_{1} | (\text{name (def}_{1} \text{ Fred)}), \lambda(x)(\text{love x ((def}_{2} x) | (\text{and (wife (def}_{2} x))

\begin{align*}
& (\text{possess (his}_{3} x) (\text{def}_{2} x))
& (= (\text{his}_{3} x) (\text{x}) ))
\end{align*}\)

Simplification:
\((\text{def}_{1} | (\text{name (def}_{1} \text{ Fred)}), \lambda(x)(\text{love x ((def}_{2} x) | (\text{and (wife (def}_{2} x))

\begin{align*}
& (\text{possess x (def}_{2} x))))
\end{align*}\)

Notice that def\_2's restriction contains a free variable x and so its argument list is unchanged by the argument reduction constraint. Hence, the representation of the verb phrase in 39 is used to derive the sloppy reading of the elided sentence shown in 40.

Example 40
George does too. (George loves George's wife.)

\((\text{def}_{1} | (\text{name (def}_{1} \text{ George)}), \lambda(x)(\text{love x ((def}_{2} x) | (\text{and (wife (def}_{2} x))

\begin{align*}
& (\text{possess (his}_{3} x) (\text{def}_{2} x))
& (= (\text{his}_{3} x) (\text{x}) ))
\end{align*}\)

Simplification:
\((\text{def}_{1} | (\text{name (def}_{1} \text{ George)}), \lambda(x)(\text{love x ((def}_{2} x) | (\text{and (wife (def}_{2} x))

\begin{align*}
& (\text{possess x (def}_{2} x))))
\end{align*}\)

Notice that the function def\_2 denotes different individuals in the trigger and elided sentences, depending
on the value substituted for \( x \).

On the other hand, suppose that \( his \) refers directly to \( Fred \). Then the intended meaning of the trigger sentence is shown in 41.

**Example 41**

Fred, loves his, wife.

\[
\begin{align*}
\text{(def)} & | \text{(name (def) Fred)), } \lambda(x)(\text{love } ((\text{def}_2 x) | \text{(and (wife (def}_2 x))}) \\
& \quad | \text{(possess (his}_3 x) \text{(def}_2 x)) \\
& \quad | \text{(=} \text{(his}_3 x) \text{(def}_1 )))
\end{align*}
\]

Simplification:

\[
\begin{align*}
\text{(def)} & | \text{(name (def) Fred)), } \lambda(x)(\text{love } ((\text{def}_2 x) | \text{(and (wife (def}_2 x))}) \\
& \quad | \text{(possess (def}_1 ) \text{(def}_2 x)))
\end{align*}
\]

Notice that once the pronoun function is replaced by \( (\text{def}_1) \), the restriction of \( (\text{def}_2 x) \) contains no free variables except those in the argument list of the function itself. Because of this, we replace \( (\text{def}_2 x) \) with a function whose argument list is empty, as shown in 42.

**Example 42**

Fred, loves his, wife.

\[
\begin{align*}
\text{(def)} & | \text{(name (def) Fred)), } \lambda(x)(\text{and (love } ((\text{def}_2 x) | \text{(and (wife (def}_2 x))}) \\
& \quad | \text{(possess (his}_3 x) \text{(def}_2 x)) \\
& \quad | \text{(=} \text{(his}_3 x) \text{(def}_1 )))
\end{align*}
\]

Simplification:

\[
\begin{align*}
\text{(def)} & | \text{(name (def) Fred)), } \lambda(x)(\text{and (love } (\text{def}_4) \text{(def}_1 )) \text{(possess (def}_1 ) \text{(def}_4))
\]

Using the representation of the verb phrase in 42, we are able to derive the strict reading of the elided sentence shown in 43.

**Example 43**

George does too. (George loves Fred's wife.)

\[
\begin{align*}
\text{(def)} & | \text{(name (def) George)), } \lambda(x)(\text{and (love } ((\text{def}_2 x) | \text{(and (wife (def}_2 x))}) \\
& \quad | \text{(possess (his}_3 x) \text{(def}_2 x)) \\
& \quad | \text{(=} \text{(his}_3 x) \text{(def}_1 )))
\end{align*}
\]

Simplification:

\[
\begin{align*}
\text{(def)} & | \text{(name (def) George)), } \lambda(x)(\text{and (love } (\text{def}_4) \text{(def}_1 )) \text{(possess (def}_1 ) \text{(def}_1))
\]

Notice that \( (\text{def}_4) \) is a constant, and so denotes the same individual in the meanings of the trigger and elided sentences. Hence, our general representation of definite noun phrases in logical form allows us to derive both the sloppy and strict readings of the elided sentence.

We have introduced a composite representation for definite noun phrases with two ways to update their meanings as more information becomes available. This approach is consistent with the three computational constraints discussed in section 1, and also provides a good model of definite behavior. For a discussion of a wider variety of examples see Harper [Har89, Har90].

### 5 Indefinites in Logical Form

In this section, we develop an initial representation for singular indefinites in logical form and way to update it once additional information is processed.
### 5.1 Indefinites: Linguistic Evidence and Initial Representation

Singular indefinite noun phrases share many behaviors with singular definites. The meanings of noun phrases embedded in each strongly affect their final meanings. However, there are differences which prevent us from initially representing singular indefinites as functions in logical form; the meaning of an indefinite is affected by negation as shown in the next example.

Example 44
Fred did not see a woman.

Meanings:
- a. $\exists x$: (woman $x$) Not(see Fred $x$)
- b. Not $\exists x$: (woman $x$) (see Fred $x$) $\equiv \forall x$: (woman $x$) Not(see Fred $x$)

Whenever there is negation in a sentence with an indefinite, two meanings of the sentence are possible. If the negation does not have scope over the indefinite, then the indefinite is represented as an existential outside the scope of the negation as shown in 44a (and could be represented as a function). In contrast, if the negation has scope over the indefinite, then it has scope over the existential operator making it equivalent to a universal (as shown in 44b). If we represent the indefinite in 44 as a function before deciding whether the negation has scope over it, then the second reading could not be expressed.

Because of the way indefinites behave with respect to negation, we represent indefinites initially as existentially quantified and restricted variables, as in example 45.

Example 45
Fred saw a dog.

($(\text{name } (\text{def } 1) \text{ Fred}), \lambda(x)(\text{saw } x [\exists y: (\text{dog } y)])$

Following Schubert and Pelletier [SP84] and Allen [All87], we leave the quantifier in the predicate-argument structure for the verb phrase. This initial representation of the indefinite is provided using only syntactic information and knowledge about how to map arguments into the predicate-argument structure, obeying the modularity constraint. Once quantifier scoping information is available, we add this information to the logical form for the sentence, using a mechanism which also allows us to account for several interesting indefinite behaviors.

### 5.2 Indefinites: Linguistic Evidence and Updating the Initial Representation

An approach which models indefinites solely as existentially quantified variables cannot account for the variety of behaviors indefinite noun phrases show. For example, a quantifier cannot have scope across sentences in logic, as in *The boy kissed every girl. She slapped him*, where the pronoun *she* cannot have *every girl* as its antecedent. However, in *The boy kissed a girl. She slapped him*, the antecedent for *she* can be *a girl*. The problem is that an existential quantifier cannot have scope across those two sentences, and, yet, *a girl* can be the antecedent for *she*.

A similar problem arises in verb phrase ellipsis. If the trigger verb phrase contains a pronoun whose antecedent is an indefinite subject, two possible meanings for the elided sentence are possible, as in example 46.
Example 46
A postman; saw his dog.
A policeman; did too.

Meanings:

a. A policeman saw the postman's dog. (strict reading)
b. A policeman saw his own dog. (sloppy reading)

When the antecedent for his is a man, the elided sentence has two possible meanings\(^{16}\), but a quantifier representation for the indefinite can only account for the reading in 46b. Given that the antecedent for his is a dog, there are two possible representations for the trigger verb phrase. The pronoun function is either replaced by the lambda variable or by the existential variable corresponding to the subject. If we combine the subject of the elided sentence with the first verb phrase representation, we are able to derive the sloppy meaning in 46b. But, if we use the second, then the meaning for the elided sentence would contain an unbound variable because existential quantifiers do not have scope across sentences.

Donkey sentences (originally noticed by Geach [Gea62]) suffer a similar difficulty. A typical donkey sentence is *Every miner who owns a donkey beats it*. Though the existential operator corresponding to a donkey cannot have scope over the pronoun it, a donkey can be its antecedent. In contrast, in *Every miner who brushed every donkey beat it*, every donkey cannot be the antecedent for it.

These examples indicate that our initial representation of a singular indefinite is not sufficient to handle a variety of linguistic phenomena. However, the existential operator is only necessary until we decide what has scope over the indefinite. Once this information is available, there is little reason to continue representing an indefinite as an existentially quantified variable. If there is some way to transform the initial representation into a form more compatible with these other indefinite behaviors and the transformation only limits the meaning of the initial indefinite (in keeping with the formal consistency constraint), then such a transformation is desirable. The transformation we have in mind is replacing an existential's variable with a skolem function.

There is precedence for converting existentially quantified variables into functions. Once scoping is specified, each existentially quantified variable is replaced by a function whose argument list consists of all of the universally quantified variables that have scope over its operator. To demonstrate how existential variables are replaced by functions during skolemization, consider example 47.

Example 47
Some man saw every woman.

Meanings:

1a. \(\exists x: (\text{man } x) \ \forall y: (\text{woman } y) (\text{see } x \ y)\)
1b. \(\forall y (\text{if } (\text{woman } y) (\text{see } y))\)

2a. \(\forall y: (\text{woman } y) \ \exists x: (\text{man } x) (\text{see } x \ y)\)
2b. \(\forall y (\text{if } (\text{woman } y) (\text{and } (\text{man } y) (\text{see } y)))\)

There are two meanings for the sentence in 47, indicated in 1a and 2a. These meanings are preserved when the existential variables are replaced by skolem functions, as shown in 1b and 2b.

When an indefinite is represented as an existentially quantified variable and is not in the scope of negation, it can be replaced by a function. However, additional information about the indefinite noun phrase must be gathered before such a transformation is performed. In particular, we must determine the antecedents of pronouns and anaphoric definites, pinpoint the meanings of embedded definite and

\(^{16}\)This example is in sharp contrast with *Every postman, saw his dog. Every policeman did too*. The elided sentence can only mean *Every policeman saw his own dog.*
indefinite noun phrases, and determine whether any embedded universally quantified noun phrases, not contained in a relative clause, distribute over the indefinite. The operators which bind the variables contained in the restriction of the existential operator, but which are not themselves contained in the restriction, necessarily have scope over the existential operator (as in the case of definite functions). We must also determine whether external quantified noun phrases and lambda operators have scope over the existential. This last requirement differs from the requirements for determining the final meaning of a definite noun phrase. We discuss why lambda operator scope over an indefinite is an issue for determining its final meaning in the next section.

The final meaning of an indefinite is affected by quantifiers that could never affect the meaning of a definite. Consider the sentence, *His mother saw every boy*. Despite the fact that *his* needs an antecedent, syntactic constraints eliminate *every boy* from the list of possible candidates. Hence, *his mother* acts as a constant in the sentence. Compare the previous sentence with *A friend of his saw every boy*. Despite the fact that the antecedent for *his* cannot be *every boy, a friend of his* could still be in the scope of *every boy*. Because the same constraints that apply to definites do not apply to indefinites, we must determine whether quantifiers within a sentence have scope over the indefinite, even if those quantifiers do not bind a variable contained in the indefinite's restriction.

To demonstrate how much information is necessary to determine the final meaning of an indefinite noun phrase, consider the initial representation for the sentence *Every man showed every boy a picture of his mother*.

**Example 48**

*Every man showed every boy a picture of his mother.*

\[
\forall x: (\text{man } x), \lambda(y)(\text{show } y \ [\exists w: (\text{and } (\text{picture } w) \\
\text{(of } w \ ((\text{def; } w \ y \ z)) \ | \ (\text{and } (\text{mother } (\text{def; } w \ y \ z) \\\n\text{(possess } (\text{his; } w \ y \ z)) \\\n\text{(def; } w \ y \ z)))))) \ w] \ \\
[\forall z: (\text{boy } z) \ z])
\]

Before determining the final meaning of the indefinite, we must determine the meaning of the definite noun phrase *his mother* and decide whether \(\forall x, \forall z, \lambda(y)\) have scope over it. If we decide that the antecedent for *his* is *every boy*, the updated logical form is shown in example 49.

**Example 49**

*Every man showed every boy a picture of his mother.*

\[
\forall x: (\text{man } x), \lambda(y)(\text{show } y \ [\exists w: (\text{and } (\text{picture } w) \\
\text{(of } w \ ((\text{def; } w \ y \ z)) \ | \ (\text{and } (\text{mother } (\text{def; } w \ y \ z) \\\n\text{(possess } (\text{his; } w \ y \ z) \\\n\text{(def; } w \ y \ z)))))) \ w] \ \\
[\forall z: (\text{boy } z) \ z])
\]

Since the restriction on \((\text{def; } w \ y \ z)\) contains only the unbound variable \(z\), we replace it with a function of \(z\).
Example 50
Every man showed every boy a picture of his mother.
\[ \forall x: (\text{man } x), \lambda(y)(\text{and } (\text{show } y \ [\exists w: (\text{and } (\text{picture } w)
\ (\text{of } w (\text{def } w y z) \ | \ (\text{and } (\text{mother } (\text{def } w y z))
\ (\text{possess } (\text{his}_2 w y z) \ | \ (\text{def } w y z)))
\ (= (\text{his}_2 w y z) z))) w]
\[ \forall z: (\text{boy } z) z] 
\[ (= (\text{def } w y z) (\text{def}_3 z)) \]
\]
\[ \forall z \text{ must have scope over } \exists w \text{ because } z \text{ is unbound in the restriction of the existential, but we must still determine whether } \forall x \text{ or } \lambda(y) \text{ also have scope over the existential. Assuming they don't, we can update the logical form in 50 by replacing the existential variable } w \text{ with a function of } z. \]

Example 51
Every man showed every boy a picture of his mother.
\[ \forall x: (\text{man } x), \lambda(y)(\text{and } (\text{show } y ((\text{indef }_4 z) \ | \ (\text{and } (\text{picture } (\text{indef }_4 z)))
\ (\text{of } (\text{indef }_4 z)
\ ((\text{def }_1 (\text{indef }_4 z) y z) \ | \ (\text{and } (\text{mother } (\text{def }_1 (\text{indef }_4 z) y z))
\ (\text{possess } (\text{his}_2 (\text{indef }_4 z) y z) \ | \ (\text{def }_1 (\text{indef }_4 z) y z)))
\ (= (\text{his}_2 (\text{indef }_4 z) y z) z))))))
\[ \forall z: (\text{boy } z) z] 
\[ (= (\text{def }_1 (\text{indef }_4 z) y z) (\text{def}_3 z)) \]
By replacing the existential variables with a function of \( z \), we indicate that only \( \forall z \) has scope over the existential. This final meaning is compatible with the initial representation of the indefinite in 48; we have simply constrained the initial meaning with additional information.

5.3 Indefinites: Verb Phrase Ellipsis

If we are to handle verb phrase ellipsis in our model, we must also consider whether the lambda operators in a verb phrase representation have scope over indefinites represented as existentially quantified variables. Consider example 52.

Example 52
Fred saw a dog.
George did too.
Meanings:
1. George saw the same dog that Fred saw.
2. George saw a different dog than Fred saw.

When an indefinite noun phrase occurs in the verb phrase of a trigger sentence, an ambiguity arises in the meaning of the indefinite in the elided sentence. George can possibly see a different dog than Fred saw\(^{17}\). If we do not consider lambda operators when converting existential variables to functions, then we cannot provide the second meaning for the elided sentence given in example 52. To see this, consider the initial representation of the trigger sentence in 53.

Example 53
Fred saw a dog.
\(((\text{def}_1) \ | \ (\text{name } (\text{def}_1) \text{ Fred})) , \lambda(x)(\text{saw } x \ [\exists y: (\text{dog } y)])
\]
\[^{17}\text{This is in contrast to } \text{Fred saw the dog. George did too, in which the elided sentence can only mean } \text{George saw the same dog that Fred saw. This example is discussed in section 4.4.}\]
If we skolemize the existential corresponding to a dog without considering the lambda operator, we replace the existential variable with a function of no variables.

**Example 54**
Fred saw a dog.
((def) | (name (def) Fred)), \(\lambda(x)\)\((\text{saw } x (((\text{indef}) | (\text{dog } (\text{indef})))))\)

The problem with the representation in 54 is that the verb phrase can only be used to generate the first meaning of the elided sentence, shown in 55.

**Example 55**
George did too. (George saw the same dog as Fred saw.)
((def) | (name (def) George)), \(\lambda(x)\)\((\text{saw } x (((\text{indef}) | (\text{dog } (\text{indef})))))\)

To handle example 52, we must not only take into account whether universal operators have scope over an existential, but we must also decide whether lambda operators do. Because the representation in 53 contains a single lambda operator, \(\lambda(x)\), we must determine whether it has scope over the existential operator. If it does then the existential variables are replaced with a function of \(x\), as shown in 56.

**Example 56**
Fred saw a dog.
((def) | (name (def) Fred)), \(\lambda(x)\)\((\text{saw } x (((\text{indef}) | (\text{dog } (\text{indef}) x)))\))

Using the verb phrase representation from 56, we are able to provide the second meaning for the elided sentence of 52, shown in 57.

**Example 57**
George did too. (George saw a potentially different dog than Fred did.)
((def) | (name (def) George)), \(\lambda(x)\)\((\text{saw } x (((\text{indef}) | (\text{dog } (\text{indef}) x)))\))

In contrast, if the lambda operator does not have scope over the existential operator, then the trigger sentence is represented as was shown in 54 and the representation of the elided sentence is derived as was shown in 55. Hence, by determining whether a lambda operator has scope over an indefinite, we provide the two readings for the elided sentence in 52.

Once we pinpoint the meanings of all noun phrases embedded in an existential’s restriction and determine which quantifiers and lambda operators have scope over it, we replace the existential’s variables with a function whose argument list consists of those variables corresponding to operators that have scope over it. In order to replace an existential variable by a function, anaphora resolution, definite argument reduction, and quantifier scoping information must be available. Hence, when an existential variable is replaced by a function, the indefinite’s meaning is pinpointed.

### 5.4 Indefinites: Benefits of the Final Representation

The representation of an indefinite as a function is very useful, especially for capturing the behavior of indefinite subjects in verb phrase ellipsis. Consider example 46 again. By converting existentially quantified variables into functions, we are able to provide the strict meaning for the elided sentence without creating an ill-formed representation; we simply replace the subject’s existential variables with a skolem constant (assuming it is not in the scope of another quantifier).
The functional representation is also quite useful for determining whether a singular indefinite can be the antecedent for a singular pronoun in a subsequent sentence. Consider example *Every woman saw a dog. It bit the tallest woman.* The antecedent for the pronoun *it* can be a *dog* only if the universal operator corresponding to *every woman* does not have scope over the existential. Consider the initial representation of the first sentence.

**Example 58**
*Every woman saw a dog.*
\[ \forall x : (\text{woman } x) \ x, \ \lambda(y)(\text{see } y [\exists z : (\text{dog } z) z]) \]

Now, suppose that we decide that the universal has scope over the indefinite, then the logical form would be expanded, as shown in 59.

**Example 59**
\[ \forall x : (\text{woman } x) \ x, \ \lambda(y)(\text{see } y ([\text{indef}_{54} x] \ | (\text{dog } [\text{indef}_{54} x]))) \]

In consequence of this scoping decision, the antecedent for *it* cannot possibly be a *dog*. We cannot replace the function for *it* with *z* without violating the formal consistency constraint. Even if we construct a discourse entity for *a dog*, following Webber [Web78], the discourse entity for (indef_{54} x) would denote a set of dogs, and the pronoun resolution module would not allow a plural entity to be the antecedent for a singular pronoun. On the other hand, if we decide that the universal does not have scope over a *dog*, then the logical form for the first sentence (shown in 58) is updated as shown in 60.

**Example 60**
\[ \forall x : (\text{woman } x) \ x, \ \lambda(y)(\text{see } y ([\text{indef}_{57} ] \ | (\text{dog } [\text{indef}_{57}]))) \]

Because a *dog* is represented as a function with no arguments, the indefinite function is compatible with the function representing *it* or it could be used to create a singular discourse entity compatible with the singular pronoun. This example demonstrates that indefinites in one sentence may be legal antecedents for singular pronouns in subsequent sentences, if the scoping decisions we make do not make them incompatible with the pronoun functions in the other sentences.

Another benefit of our approach is our ability to handle the donkey sentence, *Every miner who owns a donkey; beats it*. The antecedent for *it* is a *donkey*, yet in English, a quantified noun phrase contained in a relative clause attached to a noun phrase cannot bind a pronoun in the matrix sentence. Hence, *it* cannot be bound by the existential quantifier corresponding to a *donkey*. How then is the meaning of this sentence possible? Consider the initial representation of this sentence shown in 61.

**Example 61**
*Every miner who owns a donkey beats it.*
\[ \forall x : (\text{and } (\text{miner } x) \ x, \ \lambda(y)(\text{own } y [\exists z : (\text{donkey } z) z]) \ x, \ \lambda(w)(\text{beat } w (it_{58} w)) \]

In our approach, we represent *it* as a function of *w* only and cannot replace *(it_{58} w)* by *z* in an attempt to capture the meaning expressed in the sentence. However, if we replace the existential variable in the relative clause with a function, we may then be able to assert that the antecedent for *it* is a *donkey*.

To replace the variables corresponding to the existential operator with a function, we must determine whether the existential quantifier is in the scope of negation. Assuming that the negation introduced by
the restriction on the universal operator does not have scope over a donkey, we replace the existential variables with a function of those variables corresponding to operators that have scope over it. If we decide that only $\forall x$ has scope over the existential (i.e., $\lambda(y)$ does not have scope), we can assert the anaphoric relationship between it and a donkey. Consider how the logical form of 61 is modified to indicate the update.

Example 62
Every miner who owns a donkey beats it.
$\forall x \ (\text{if} \ (\text{and} \ (\text{miner} \ x) \ x, \ (\lambda(y) \ (\text{own} \ y \ ((\text{indef}_2 \ x) \ | \ (\text{donkey} \ (\text{indef}_2 \ x)))))) \ x, \ (\lambda(w) \ (\text{beat} \ w \ (\text{it}_{58} \ w)))$

Since (it$_{58}$ w) is consistent with a function of $x$ (because $\lambda(w)$ abstracts the variable $x$), we can assert the anaphoric relationship, as shown in 63$^{18}$.

Example 63
Every miner who owns a donkey, beats it.
$\forall x \ (\text{if} \ (\text{and} \ (\text{miner} \ x) \ x, \ (\lambda(y) \ (\text{own} \ y \ ((\text{indef}_2 \ x) \ | \ (\text{donkey} \ (\text{indef}_2 \ x))))))$
$x, \ (\lambda(w) \ (\text{and} \ (\text{beat} \ w \ (\text{it}_{58} \ w))) \ (= \ (\text{it}_{58} \ w) \ (\text{indef}_2 \ x))))$

It is important to note that the type of the quantified noun phrase strongly affects whether skolemization can make it accessible to a pronoun function whose argument list does not contain that noun phrase's variable. If a noun phrase is initially represented as a universal, then unless the universal variable is included in the argument list of the pronoun, it cannot be the antecedent for that pronoun even if it is in the scope of negation, as in Every miner who did not see every donkey beat it, in which the antecedent for it cannot be not every donkey. In contrast, so long as an indefinite remains an existential, even if it cannot bind the pronoun, it may be accessible to the pronoun once we determine its precise behavior and convert it into a function.

There are several advantages gained by replacing existential variables by functions. First, it provides a way to indicate quantifier scoping in a representation containing only universal and existential quantifiers. Second, the method of indicating scope is similar to Allen's [All87] method; we are not limited to expressing scope as a linear string of operators. Third, universal variables cannot be replaced with functions. Hence, skolemization may be useful for modeling the differences between universals and indefinites in English. Furthermore, once quantifier scoping information is available, replacing existential variables by functions is a meaning preserving operation as required by the formal consistency constraint. Finally, a functional representation for an indefinite allows us to account for several behaviors that are poorly modeled using an existential variable representation of indefinites alone.

6 Implementation

In this section, we describe the implementation of a system to generate the logical form for a sentence. The parser takes a sentence as input and outputs its logical form, demonstrating that the logical form for a sentence can be provided using only lexical and syntactic information. Additionally, it provides

$^{18}$Our solution has much in common with Webber's [Web78] parameterized individuals. Webber introduces a parameterized individual (which looks much like an indefinite function) as the antecedent for it. However, she does not modify the initial representation of the indefinite.
the meaning for a sentence with verb phrase ellipsis, demonstrating that an intermediate representation for a trigger sentence can be updated to a reasonable final meaning for that sentence and consequently provide a reasonable meaning for an elided sentence.

We illustrate how our implementation works on example 64.

Example 64
Fred gave the psychiatrist who cares for (his, mother), her diary.
George did too.
Meanings:
1. George gave the psychiatrist who cares for Fred’s mother Fred’s mother’s diary.
2. George gave the psychiatrist who cares for George’s mother George’s mother’s diary.
3. George gave the psychiatrist who cares for George’s mother Fred’s mother’s diary.
4. George gave the psychiatrist who cares for Fred’s mother George’s mother’s diary.

The program processes the sentences in this example, one at a time, parsing the trigger sentence first. The trigger sentence’s logical form is provided as the parser processes its components. Logical form is created in a nearly compositional way. However, because the program must keep track of variables associated with operators in order to provide initial representations for pronouns and definites, complete compositionality is impossible. Once the end of a sentence is reached, the logical form for the sentence is available. However, because the parser provides all parses for a sentence and because logical form depends on the parse tree, the parser can sometimes generate multiple logical forms for a sentence. In such a case, the user is asked to pick the intended parse (and hence, the intended logical form). Because the trigger sentence in example 64 has a single parse tree, the parser outputs one logical form, shown in 65.

Example 65
Fred gave the psychiatrist who cares for his mother her diary.

The program keeps track of logical subjects and objects to prevent a passive voice sentence from being the trigger for an active voice sentence (or vice versa). In the previous examples of this paper, logical roles were indicated by their position to avoid cumbersome labeling of roles. However, many times a passive sentence does not contain a logical subject, so it would be hard to detect the voice of the sentence without labeling. Hence, we indicate a sentence has active voice by labeling the subject’s lambda variable as logical subject and passive voice by labeling it as logical object. Notice that we do not specify case roles like agent because their determination can require contextual information. Also, the case role of a subject need not be the same in the trigger and elided sentence, e.g., Fred hit the window. The hammer did too. The subject in the first sentence is probably filling the agent role, while the subject of the second fills the role of instrument.

Because our program provides meanings for sentences with verb phrase ellipsis, it examines each
logical form output by the parser to see if it contains verb phrase ellipsis. The logical form in 65 does not contain an elided verb phrase, hence, additional processing is put off until the final meaning of the sentence is needed (i.e., if it is the trigger sentence for a verb phrase ellipsis). Ideally, processing should be done as information becomes available. However, in this implementation, it is simpler to add additional information to the logical form of a trigger sentence only when its meaning must be determined to provide the meaning of an elided verb phrase. Hence, the logical form in 65 is saved on a stack of recently processed sentences.

Once the program saves the logical form for the potential trigger sentence, the parser processes the elided sentence and outputs a logical form for it. The fact that the representation contains an elided verb phrase is indicated by the dummy predicate in 66.

Example 66
George did too.

((def) | (name (def) George), λ(u) (dummy3 (subject u)))

To determine the meaning of the elided sentence, the program must locate its trigger sentence. It presents the program user with potential logical forms, the most recently processed first. The user decides which is the intended trigger sentence. Assume the user indicates 65 is the representation for the trigger sentence. The ambiguity in the elided sentence’s meaning arises because there are many ways to indicate the final meaning of its trigger sentence. Hence, before the program can determine the meaning of the elided sentence, it must determine the trigger sentence’s meaning by locating antecedents for all pronouns and anaphoric definite noun phrases, determining quantifier scoping, and pinpointing meanings of all non-anaphoric definites.

To determine the final meaning of the logical form in 65, the program must obtain additional information. First, it attempts to find antecedents for all of the pronoun functions (in this case, (her5 y) and (his4 y z)). Once the pronouns have antecedents, it handles all definites, and then all quantified noun phrases. However, this control sequence is only realized when there is no interdependency between noun phrases in the sentence (e.g., the antecedents for all pronouns and anaphoric definites are outside of the current sentence). Once the program attempts to determine whether a definite or indefinite noun phrase in a sentence can be the antecedent for a pronoun, the flow of control can change because additional information about the possible antecedent may be required to determine whether it is compatible with the pronoun function.

Suppose the program begins disambiguating the trigger sentence by searching for the antecedent of the pronoun function (her5 y). First, the program asks the user whether the antecedent occurs in a previous sentence. If so, the user also provides the discourse entity for that antecedent. Assume that the user decides that the antecedent is in the same sentence as the pronoun. So the program asks the user to choose the antecedent from all possible antecedent candidates after eliminating impossible ones using the pronoun compatibility routine.

The pronoun compatibility routine determines whether a noun phrase in a sentence can be the intrasentential antecedent for a pronoun. If the pronoun is reflexive, its antecedent must occur in the same clause and cannot be embedded in a noun phrase. On the other hand, if the pronoun is not reflexive, its antecedent cannot be in the same clause unless the pronoun or its antecedent is embedded in another noun phrase. This routine also checks whether potential antecedents are formally consistent with the
pronoun function. If the potential antecedent is an indefinite or definite not immediately compatible with the pronoun, it may still become compatible with the pronoun function after its meaning is pinpointed. When more information is needed to determine compatibility, the program obtains that information as its next task.

So the program checks each noun phrase in the logical form to determine whether any of them (i.e., (def_1), (def_5 y), (def_2 y), (his_4 y z), or (def_3 y z)) is compatible with (her_6 y). The program rules out (def_5 y) as an antecedent because (her_6 y) is embedded in that function’s restriction. If gender information was available in the lexicon, it could have been used to rule out (def_1) and (his_4 y z) as antecedents. Assume instead that the user rules both of them out. The remaining antecedent candidates are (def_2 y) and (def_3 y z). The function (def_2 y) is compatible with the pronoun function, but (def_3 y z) is not immediately compatible with it. Hence, the program shifts its attention to (def_3 y z).

To further process (def_3 y z), the program determines its necessary arguments by pinpointing the meaning of each embedded noun phrase. In this case, there is only an embedded pronoun function, (his_4 y z), with no assigned antecedent. Hence, the program must find that pronoun’s antecedent. As before, the program asks the user if the antecedent for the pronoun occurs in a previous sentence. Assume a negative reply. The program then determines which of the noun phrases in the sentence are compatible with the pronoun function. In this case, only (def_1), (def_5 y), (her_6 y), and (def_2 y) are possible antecedents. Assume that the user picks (def_1) as the antecedent. However, because (def_1) is a subject, the program asks the user to decide whether the pronoun reference is direct or indirect. Assuming the user takes the indirect option, the logical form is updated as shown in example 67.

Example 67
Fred gave the psychiatrist who cares for his mother her diary.
((def_1) | (name (def_1) Fred),
\lambda(y)\textit{give (subject y) (object ((def_5 y) | (and (diary (def_5 y)) (possess (her_6 y) (def_5 y)))))
(indirect-object
((def_2 y) | (and (psychiatrist (def_2 y)))
(def_2 y), \lambda(z)\textit{care (subject z)
(for ((def_3 y z) | (and (mother (def_3 y z)))
(possess y (def_3 y z)))))))))

Notice that the pronoun is replaced by its value in the logical form.19

Once the program determines the necessary arguments for a definite function, it must decide whether it is anaphoric or not. If the restriction of the definite function does not contain an unbound variable, the definite could be anaphoric so the program asks the user to decide. If the user decides that it is anaphoric, then just like for pronouns, the program queries the user about antecedents. If a potential antecedent occurs in the same sentence as the definite, the program must test it to determine whether it is compatible. A definite cannot be c-commanded by its antecedent, cannot be embedded in its antecedent, and must be formally consistent with its antecedent. If a definite function’s restriction contains any free variable or if the user decides that the definite is non-anaphoric, then the program performs argument reduction on the definite function.

Because the program has determined the meanings of all of the noun phrases embedded in (def_3 y z),

---

19 The program does keep track of pronouns and their values to ensure that the multiple pronoun constraint described in [Har90] is obeyed. The logical forms shown in this section are the simplified forms output by a special print routine.
it finishes processing that function. The restriction of the function (see 67) contains the variable \( y \), bound by an operator outside of the restriction, so the definite cannot be anaphoric. However, since the only necessary argument is \( y \), the function is replaced by a function of \( y \), as shown in 68.

**Example 68**

Fred gave the psychiatrist who cares for his mother her diary.

\[
\lambda(y)(\text{give (subject } y \text{) (object ((def} \ y \mid (\text{and (diary (def}_5 \ y) \text{) (possess (her } y \text{) (def}_5 \ y))))})
\]

\[
\text{(indirect-object)}
\]

\[
((\text{def}_2 \ y) \mid (\text{and (psychiatrist (def}_2 \ y)))
\]

\[
(\text{def}_2 \ y), \lambda(z)(\text{care (subject } z) \mid (\text{for ( (def}_7 \ y) \mid (\text{and (mother (def}_7 \ y) \text{) (possess } y (\text{def}_7 \ y))))))))
\]

Because \((\text{def}_7 \ y)\) replaces \((\text{def}_3 \ y \ z)\) as the meaning for \textit{his mother}, the program determines that \textit{his mother} can be the antecedent for \textit{her}. Assume the user confirms it as the antecedent and the program updates the logical form as shown in 69.

**Example 69**

Fred gave the psychiatrist who cares for (his mother); her diary.

\[
\lambda(y)(\text{give (subject } y \text{) (object ((def} \ y \mid (\text{and (diary (def}_5 \ y) \text{) (possess ( (def}_7 \ y) (def}_5 \ y))})})
\]

\[
\text{(indirect-object)}
\]

\[
((\text{def}_2 \ y) \mid (\text{and (psychiatrist (def}_2 \ y)))
\]

\[
(\text{def}_2 \ y), \lambda(z)(\text{care (subject } z) \mid (\text{for ((def}_7 \ y) \mid (\text{and (mother (def}_7 \ y) \text{) (possess } y (\text{def}_7 \ y))))))))
\]

To finish processing the sentence, the program must specify the meanings of the functions \((\text{def}_2 \ y)\), \((\text{def}_5 \ y)\), and \((\text{def}_1)\). First, it must determine the necessary arguments (i.e., all of the unbound variables in the restriction) for each function. Notice that \( y \) is a necessary argument for both \((\text{def}_2 \ y)\) and \((\text{def}_5 \ y)\), and so the definites are not anaphoric and do not require argument reduction. In contrast, the restriction of \((\text{def}_1)\) does not contain any free variables and so could be anaphoric. Assume the user informs the program that it is not anaphoric, then since the function contains no arguments, there is no need to perform argument reduction.

Next, the program asks the user to determine which of the embedded verb phrases in the trigger sentence is the trigger for the elided verb phrase. Assume the user picks the lambda function for the matrix verb phrase shown in 69. The program checks it to ensure that it contains no free variables\(^{20}\), is compatible with the voice of the elided verb phrase, and that a trigger sentence with more than one pronoun whose antecedent is the same syntactic subject obeys the multiple pronoun constraint discussed in [Har90]. Notice the verb phrase in 69 contains no free variables and is compatible in voice to the elided verb phrase in 66. Hence, the program replaces the elided verb phrase with the trigger

\(^{20}\)This program only deals with intersentential verb phrase ellipsis. Hence, all variables in the trigger verb phrase must be bound in the verb phrase, otherwise the elided sentence cannot receive a meaning. If we augment our approach to handle antecedent-contained ellipsis, we would have to allow variables bound by an operator outside of the verb phrase but inside the meaning of the sentence.
verb phrase, as shown in 70.

Example 70
George did too.
((def; ) | (name (def; ) George)),
\[ \lambda(y)(\text{give (subject } y) (\text{object } ((\text{def; } y) | (\text{and (diary (def; } y)) (\text{possess (def; } y) (\text{def; } y))))))
\]
(indirect-object
((def; y) | (\text{and (psychiatrist (def; y))})
(\text{def; } y), \lambda(z)(\text{care (subject } z)
(\text{for } ((\text{def; } y) | (\text{and (mother (def; } y))
(\text{possess } y (\text{def; } y))))))))

One other important routine required by the program but not needed for the above example converts indefinites into functions (as discussed in section 5). Since we only use universal and existential quantifiers, once existential variables are replaced with functions, quantifier scope is specified. Also, by replacing existential variables with functions, the program provides another way to establish compatibility between an anaphoric noun phrase and an indefinite. Even if an indefinite is initially incompatible with a pronoun because its variable is not contained in the pronoun function’s argument list, the indefinite’s function may be compatible with the pronoun.

We have described how our program provides the logical for for a sentence and determines the meaning of a sentence with verb phrase ellipsis. Once an elided sentence is detected, the program locates the trigger sentence with user assistance and then attempts to determine a single meaning for that sentence. The program updates logical form on a need-to-know basis. If it needs to decide whether a certain noun phrase can be the antecedent for a pronoun, additional information may be required to determine compatibility. Gathering information on a need-to-know basis seems to be a necessary strategy because quantifier scoping and anaphora decisions affect a pronoun’s access to a possible antecedent.

7 Related Work

There are two current approaches philosophically similar to ours, that of Pollack and Pereira [PP88] and Alshawi and van Eijck [AvE89]. Pollack and Pereira provide the interpretation of a sentence using an approach which relies on information collected after the semantic representation is constructed. Their semantic representation is constructed using compositional rules which rely on syntactic information. Their pragmatic rules operate on the semantic representation and rely on contextual information to pinpoint the meaning of the sentence. Alshawi and van Eijck introduce the concept of quasi logical form, which is precisely what we have called logical form in this paper. They deal with a different variety of phenomena, but philosophically our approaches are similar in many ways. We all agree that in order to determine the meaning of a sentence we must first use all the clues given by the lexicon and the syntactic analysis of a sentence to build a partial meaning, then we have to have some way to exploit contextual information to specify the precise meaning of the sentence.

In the rest of this section, we review research on past representations of pronouns, definite noun phrases, and indefinite noun phrases. We emphasize verb phrase ellipsis research because it considers
not only the representation of sentences in general, but also the representation of trigger sentences. Each approach is examined in the light of our computational constraints and their modeling capability. We review the research on pronouns, describe why definite noun phrases are not modeled well using either a definite description or a definite quantifier, and discuss how Heim’s file change semantics and Discourse Representation Theory (or DRT) handle differences between indefinite noun phrases.

7.1 Verb Phrase Ellipsis and Models of Pronouns

Pronouns have often been classified as either bound variable or referential pronouns. The adequacy of this dichotomy is questionable for modeling pronouns in sentences with or without verb phrase ellipsis (e.g., Sag [Sag76], Webber [Web78], Reinhart [Rei83], Partee and Bach [PB81]). The models of verb phrase ellipsis consider not only pronouns in normal sentences but also pronouns in trigger sentences; they must account for the ambiguity that arises when a pronoun’s antecedent is the syntactic subject of a trigger sentence. Sag [Sag76] and Webber [Web78] handle this ambiguity by introducing a rule to replace a pronoun whose antecedent is known to be the syntactic subject of a trigger sentence with the lambda variable corresponding to that subject. They also assume that a pronoun whose antecedent is a non-subject definite is necessarily referential. A problem arises because a pronoun’s antecedent can be a non-subject non-referential definite, as in:

Example 71
Trigger Sentence: Fred showed (his, mother), her, dog.
Elided Sentence: George did too.
Meanings:
1. George showed Fred’s mother Fred’s mother’s dog.
2. George showed George’s mother George’s mother’s dog.
3. *George showed George’s mother Fred’s mother’s dog.
4. *George showed Fred’s mother George’s mother’s dog.

Given the indices on the noun phrases, the elided sentence has two meanings, that is the first and second meaning shown in 71. However, Sag’s and Webber’s models sanction the first and third meaning because Fred’s mother can be the only meaning for her given their approach.

Reinhart [Rei83] also indicates that pronouns are either bound variables or referential, providing a syntactic rule for determining when a pronoun can be bound by its antecedent; a pronoun can be bound by a noun phrase if and only if it c-commands the pronoun. Reinhart does not represent pronouns or definites as quantified terms, yet she claims that when a pronoun’s antecedent is a definite noun phrase or a pronoun that c-commands the pronoun, then the pronoun is bound by a lambda operator abstracting the antecedent. At first glance, the idea of binding a pronoun with the lambda operator of its antecedent (given that the noun phrase c-commands the pronoun) seems promising; it can be used to handle example 71. However, in English, a non-referential definite can be a pronoun’s antecedent even if it does not c-command the pronoun, as in Every man gave the psychiatrist who cares for (his; mother), her, diary. Reinhart can only provide the pronoun her with a referential meaning, which is inappropriate in this case. Also, consider Reinhart’s assumption that the lambda variable is the only non-referential representation for a pronoun whose antecedent is a definite noun phrase in the same sentence. If it is correct, then there should only be one meaning for a pronoun whose antecedent is a
non-referential definite subject. However, consider example 72\textsuperscript{21}.

**Example 72**

Every man believes that (his, wife), can defend herself, better than he, can.

**Meanings:**

1. Every man believes that his wife can defend herself better than he can defend himself.
2. Every man believes that his wife can defend herself better than he can defend her.

Reinhart's approach can only provide the first meaning of the elided sentence in 72. This example suggests that pronouns can refer to definite subjects in two non-referential ways; lambda abstraction accounts for only one of them.

Partee and Bach [PB81] attempt to dispense with logical form in translating from syntax to final interpretation, building on Montague's [Mon70] general theory (with a few modifications to get around the strict compositionality of that approach). All of the possible representations for ambiguous sentences are simultaneously generated, avoiding the need for an intermediate level of representation. They directly provide model-theoretic interpretations for sentences containing pronouns and elided verb phrases. In their approach, null or elided verb phrases and pronouns are initially represented as variables. Pronouns are represented as variables which are either bound by some operator or remain free within the representation of the sentence. If a pronoun variable is unbound, it is assigned some value by a context assignment function, that is, a function which maps the variable to the individual that the pronoun denotes. In other words, pronouns are either bound variables in this model or they are referential. An elided verb phrase is represented as a free property variable, typed to receive a value corresponding to a verb phrase already in discourse. It receives its interpretation in much the same way as an unbound pronoun variable, with the exception that its antecedent must be available in linguistic context. Once the value of the null verb phrase is specified, the meaning of the elided sentence is determined.

Difficulties arise in Partee and Bach's approach to verb phrase ellipsis because there is an interesting interaction between the assignment of values for pronouns and the assignment of values for the null verb phrase. Bach and Partee point out a variety of examples for which their approach fails. Because there is no mechanism for ensuring that a pronoun bound in the trigger sentence is bound by the same operator in the elided sentence, their approach provides a host of impossible interpretations for elided sentences. For example, their approach provides an impossible interpretation for the elided sentence in *No man believes that Mary loves him. But she does*, given that *no man* is the antecedent for *him* and the meaning of the null verb phrase is *loves him*. Since the variable for *him* is unbound in the elided interpretation, it is must be assigned a value (e.g., Fred) by the context assignment function. Partee and Bach discuss additional examples in which an elided sentence receives an impossible interpretation when a free pronoun variable in the trigger verb phrase becomes accidently bound by a quantifier in the elided sentence. In addition to the problems pointed out by Partee and Bach, others arise if we assume that definite noun phrases are quantified.

In addition to the previously mentioned difficulties modeling pronouns, the past approaches also violate our computational constraints. For example, Sag's [Sag76] and Reinhart's [Rei83] approaches violate the modularity constraint since each provides a pronoun's representation only after its antecedent is known. Webber's [Web78], Sag's, and Reinhart's approaches violate the formal consistency constraint

\textsuperscript{21} The inspiration for this example was a similar example discussed in Sells, Zaenen, and Zec [SZZar].
by replacing pronoun strings with a variable to account for bound variable meanings of a pronoun. Partee and Bach’s approach [PB81] does not violate the formal consistency constraint but violates our compactness constraint.

7.2 Past Representations of Definites and Indefinites

In this section, we examine previous representations of definite noun phrases. In particular, we review definite descriptions and definite quantifiers. We also examine some recent work which departs from traditional representations of definite noun phrases (e.g., Heim [Hei82], Roberts [Rob87], Kamp [Kam81], Klein [Kle87]).

7.2.1 Definite Descriptions and Definite Quantifiers

Many researchers have attempted to represent definite noun phrases using definite descriptions or definite quantifiers. Russell [Rus71] introduced definite descriptions to capture the meaning of definite noun phrases like the dog in example 73.

Example 73
The dog barked. (barked (ax) (dog x))
which means:
\( \exists x \) (and (Dog x)) ; The dog exists.
\( \forall y ((dog y) \leftrightarrow x=y) \) ; It is the one-and-only dog.
(barked x)) ; It barked.

The definite description, (ax)(dog x), which stands for the object x such that the property (dog x) is true names a unique object, and hence, is translated into the formula, \( \exists x \) (and (dog x) \( \forall y ((dog y)\leftrightarrow (x=y)) \)). Notice three important features of the meaning of the sentence in example 73; the dog described by the definite noun phrase is assumed to exist, the dog described by the definite noun phrase is assumed to be unique, and the definite fills some role in the sentence.

Definite descriptions suffer from several problems. First, there is no role specified for the effect of context on the uniqueness statement (a problem noted by many people, including Allen [All87] and Hintikka and Kulas [Hin85]). For example, the dog in 73 is described as the-one-and-only the dog, regardless of context. Second, definite descriptions do not adequately model anaphoric definites (as noted by [Hin85]), which need not be unique and seem to adapt to the behavior prescribed by their antecedents, as in Every boyi saw (hisi dog)j before the beastj saw himi. To cover this example, the definite description for the beast could be replaced by some value consistent with the representation of its antecedent, but not without violating the formal consistency constraint, or we could devise another representation for anaphoric definites, but this would violate our compactness constraint. Another difficulty involves the representation of Bach-Peters sentences, like (The boy who wrote herj kissed (the girl who loved himi)j, which cannot be represented without infinite recursion (as noted by Hintikka and Kulas [Hin85]).

Other researchers have represented definites using the quantificational meaning of a definite description directly (e.g., Webber [Web83] and Montague [Mon70]). While an in-place definite description simply fills an argument slot in a predicate-argument structure representing the sentence, a quantifier scopes an open sentence. For example, the sentence, The dog barked, could be represented as shown in
Example 74
The dog barked.
\( \exists x \: (\text{dog } x) \: (\forall y \: ((\text{dog } y) \iff x = y)) \: (\text{barked } x) \)
Short hand notation: \( \exists x: (\text{dog } x) \: (\text{barked } x) \)

The quantifier \( \exists ! \) means there exists a unique. Does the definite quantifier model the behavior of a definite noun phrases better than the in-place definite description?

The quantificational representation for definites suffers from several problems. First, as one might guess, it suffers from the same uniqueness problem that in-place definite descriptions have\(^{22}\). Another problem stems from the fact that other noun phrases represented using quantifiers exhibit behaviors that definite noun phrases do not share. For example, quantified noun phrases often participate in quantifier scope ambiguities, many of which do not affect definites. For example, compare *Every man loves a woman* with *Every man loves the woman*.

Example 75
a. *Every man loves a woman.*
1. \( \forall x: (\text{man } x) \: \exists y: (\text{woman } y) \: (\text{loves } x \: y) \)
2. \( \exists y: (\text{woman } y) \: \forall x: (\text{man } x) \: (\text{loves } x \: y) \)

While the two representations in 75a express different meanings, the representations in 75b express the same meaning because uniqueness ensures their equivalence\(^{23}\). Definite descriptions, on the other hand, provide only a single representation for *Every man loves the woman* while providing multiple representations for sentences with definite scope ambiguity, as in *The mechanic adjusted the steering wheel in each car*. As we already discussed in section 4.1, verb phrase ellipsis poses another difficulty for quantified definite subjects (e.g., example 19). Finally, pronoun references to definites are not constrained in the way that pronoun references to other quantified noun phrases are. For example, when a quantifier is embedded in a relative clause attached to a noun phrase, it cannot bind a pronoun in the matrix clause, as in *Fred gave the psychiatrist who cares for every woman her diary*. In contrast, the pronoun *her* in *Every man gave the psychiatrist who cares for his mother her diary* can have his mother as its antecedent. If definites and universals are both quantified, then why is it that definites behave so differently from universals? One might, like Hornstein [Hor84], decide that definites are quantified but have different properties than universal quantifiers. However, this assumption does not correct some of the problems of definite quantifiers, like uniqueness or their inability to model anaphoric definites.

Rather than attempting to patch up definite quantifiers, it seems far better to represent definites as functions. First, the representation of a definite noun phrase as a uniquely named function provides us with a way to handle anaphoric definite noun phrases, in contrast to definite descriptions and quantifiers. Additionally, we represent definites as a function with a restriction and the restriction provides us

\(^{22}\)Dowty, Wall, and Peters [DWP81] suggest a nice way to fix the uniqueness problem. They eliminate the one-and-only aspect of a definite quantifier by relativizing uniqueness to a context of utterance (much as the domain of a universal noun phrase must be relativized to a context of utterance). Though their solution improves definite quantifiers, it does not eliminate the problem with definite anaphora.

\(^{23}\)In order for 75b to provide a different reading from 75b, the variable \( x \) would have to be found in the restriction of the quantifier \( \exists y \).
with a nice mechanism for determining the final meaning of the definite. Hence, we are able to capture
the nice properties of a definite description without neglecting anaphoric definites. Definite functions,
because of their unique name, also do not have the problem with uniqueness which both definite
descriptions and quantifiers have. Definite functions are distinct unless an equality is asserted between
them. Finally, while definite quantifiers can violate the compactness and formal consistency constraints
(formal consistency is violated when a wide scope definite is replaced with something that cannot be
described as a constant and compactness is violated when a different representation is introduced to
handle anaphoric definites), our approach obeys our constraints more closely.

7.2.2 Heim (1982) and Discourse Representation Theory

Another approach to handling definites is to provide a representation only after more information is
available about the meaning of a sentence [Hei82, Kam81, Rob87, Kle87]. These approaches build a
model for the meaning of a series of sentences in discourse, not just individual sentences. The hallmark
of these approaches is their ability to handle anaphoric definites in a reasonable way.

Heim [Hei82] claims that definites and indefinites should be treated very similarly since both can
be referred to across sentence boundaries, unlike universal noun phrases. To provide an interpretation
for a sentence, Heim first determines the logical form for a sentence. The logical form is essentially
a parse tree with quantifier scoping information indicated, though it is not a logical representation
for the meaning of the sentence. Once the logical form for a sentence is constructed, she provides a
file change semantics for the sentence using felicity conditions to distinguish definites from indefinites.
For example, the novelty-familiarity felicity condition states that indefinites should always cause a new
discourse referent (or file card) to be created in the discourse model but definite noun phrases should
not introduce a new discourse referent (or file card). Clearly, Heim’s approach emphasizes the anaphoric
aspect of the definite noun phrase, which both definite descriptions and definite quantifiers fail to handle
well.

Heim’s model handles anaphoric definites, but must introduce accommodation to cover non-anaphoric
definites. Consider the sentence Every man loves his mother. In Heim’s model, accommodation, a
process originally introduced by Lewis [Lew79], allows the introduction of a new file card for a definite
noun phrase if and only if that noun phrase is related to a previous file card. In the example, his mother
is non-anaphoric, however, given that the antecedent for his is every man, accommodation allows the
introduction of a new discourse referent. However, to provide a file card for his mother, the model
must know that his refers to every man; accommodation requires more than syntax and sentence-level
information to provide the representation for some definite noun phrases. Additionally, Heims’s model
would have trouble handling any definite noun phrases without an accommodation link to a previous
noun phrase in the discourse model.

Heim’s approach requires a considerable amount of information to be known before a definite or
indefinite noun phrase is represented in file change semantics. Her logical form for a sentence is not a
logical representation for the meaning of the sentence, though it does indicate quantifier scoping informa-
tion. Also, before a file card can be created for a non-anaphoric definite, pronoun antecedents have
to be known, providing another violation of the modularity constraint. Hence, the process of represent-
ing sentences in file change semantics does not seem to be consistent with our goal to incrementally
Kamp [Kam81] introduces a discourse theory similar to Heim’s, called Discourse Representation Theory (DRT), providing a model-theoretic interpretation for discourse models. Both theories are motivated by the fact that pronouns in one sentence can have definite and indefinite antecedents in another sentence, while universals cannot bind pronouns in other sentences. Kamp’s approach has been extended by several researchers (e.g., Klein [Kle87], Roberts [Rob87]). We introduce DRT, not as it was given in Kamp [Kam81], but as it is discussed in Roberts [Rob87] because Kamp does not discuss quantifier scope ambiguities or definites.

In DRT, a set of construction rules converts natural language into discourse structures. To do so, however, quantifier scoping information must be specified, since the discourse model provided depends on the quantifier scoping chosen for the sentence. Consider how Someone loves everyone is handled in DRT. This sentence has two possible meanings (corresponding to the two quantifier scoping orders). In the first case, shown in Figure 3a, someone has scope over everyone and acts like a constant. The discourse referent for someone is $x2$ and the discourse referent for everyone is $x1$. Universal noun phrases always introduce an antecedent-consequent box like the one shown in Figure 3a. Because $x2$ is defined outside of the influence of $x1$ in this representation of the sentence, it acts like a constant. In contrast, consider how the sentence is represented if everyone has scope over someone, shown in Figure 3b. Because the discourse referent for someone is created in the consequent box of the universal, its denotation depends on $x1$, the discourse referent for everyone. Hence, each quantifier scoping requires a different discourse representation for the meaning of the sentence.

Mapping into a discourse representation is a top-down process which reduces the original sentence to a structure with a discourse referent for each noun phrase, with predicates indicating restrictions on the discourse referents as well as relations between discourse referents. As we already pointed out, a universal is represented by placing its discourse referent and restriction into an antecedent box, with additional sentence information placed in the consequent box (giving a meaning like a universal in predicate calculus). Indefinites and definites are represented by placing their discourse referents and restriction information in the box corresponding to the current level in the model. An accessibility relation determines when a pronoun can have a particular discourse referent as its antecedent. A pronoun’s antecedent can be any discourse referent defined in the box where the pronoun is instantiated or in any box containing that box. Additionally, a pronoun in a consequent box can also refer to
Figure 4: Discourse Representation for *Every miner who owns a donkey beats it*.

anything in the antecedent box (unless the antecedent is embedded in another box contained in the antecedent box).

Roberts [Rob87] combines DRT with c-command to distinguish two types of binding, c-command binding and discourse binding. C-command binding occurs when the best way to represent the anaphoric noun phrase is by replacing it with the variable associated with the operator of the noun phrase that c-commands it. On the other hand, discourse binding is needed to handle anaphoric dependencies on things that don’t c-command a pronoun or anaphoric definite. For example, consider the sentence *Every miner who owns a donkey beats it*. In Robert’s approach, if a donkey had c-commanded it, then no discourse referent would be created for it, instead, the pronoun would be represented using the discourse referent of its antecedent. However, because a donkey does not c-command it, the sentence is handled as shown in Figure 4. Notice that the pronoun is represented as a discourse referent $x_3$, which is equated with the discourse referent for a donkey (i.e., $x_2$). Pronouns that haven’t already been replaced by a discourse referent, must be equated with some accessible discourse referent.

DRT has also been augmented to handle verb phrase ellipsis. Klein [Kle87] develops a model of verb phrase ellipsis by introducing a device which is very similar to lambda abstraction. With this device, he is able to represent verb phrases and *abstract* the syntactic subject of the sentence. Consider how example 4 is handled.

**Example 4**

**Trigger Sentence:** Fred loves his wife.

**Elided Sentence:** George does too.

**Meanings:**
1. George loves Fred’s wife.  (strict meaning)
2. George loves George’s wife.  (sloppy meaning)

The discourse representation for the sloppy reading of the elided sentence is shown in Figure 5a. Klein represents the verb phrase for the trigger sentence as a boxed structure named $P$. Within this box is a distinguished variable $x_2$ (distinguished variables are marked with brackets), which corresponds to the abstracted subject. The trigger sentence is represented as $P(x_0)$, which is very similar to applying the discourse referent for the subject to a lambda function named $P$. Now the discourse referent for *his* is $x_4$, which can either be equated with the distinguished discourse referent (i.e., $x_2$) or with something outside of the verb phrase box. To get the sloppy reading, it is equated with the distinguished discourse referent, as shown in Figure 5a. The elided sentence is represented initially as $Q(x_1)$, where $x_1$ is the discourse referent for the subject of the elided sentence. The sloppy reading for the sentence
is provided when $Q$ is equated with $P$. On the other hand, to derive the strict reading of the elided sentence, the discourse referent for the pronoun in the verb phrase is equated with the discourse referent for the subject, namely $x_0$ (as shown in Figure 5b). Again the meaning of the elided sentence is derived by equating $Q$ with $P$, but in this case the pronoun’s discourse referent is equated with the discourse referent for the subject. Hence, Klein derives the two expected readings for the elided sentence in 4.

This approach to verb phrase ellipsis is similar to ours, except we introduce explicit differences between definite and indefinite noun phrases. To see why this is an issue compare examples 76 and 77.

Example 76
Fred saw a friend of his.
George did too.
Meanings:
1. George saw the same friend of Fred’s.
2. George saw a different friend of Fred’s.
3. George saw a friend of George’s.

Example 77
Fred saw his friend.
George did too.
Meanings:
George saw Fred’s friend.
George saw George’s friend.

Because definites and indefinites are represented in the same way in DRT, the strict readings for both of these examples are represented in precisely the same way (shown in figure 6a) as are the sloppy readings (shown in figure 6b). However, the meaning of *his friend* is quite different from the meaning of *a friend of his* in verb phrase ellipsis. The elided sentence in 77 cannot mean *George saw a different friend of Fred’s*, in contrast to the elided sentence in 76. One more interpretation is available for the elided sentence in 76 than for the elided sentence in 77, but only two representations are provided.

In our approach, definites and indefinites are treated quite differently, not simply in how they are
Figure 6: A Discourse Representation for the Readings of Examples 76 and 77

initially represented, but also in how they are treated to determine final meanings. Notice, that we have no counterpart of the argument reduction constraint for indefinites. Indefinites can be expressed as functions of variables even when there are no unbound variables in their restrictions. Because we treat definites differently from indefinites, it is easy for us to explain the differences between examples 76 and 77. We demonstrate how we provide the three readings of the elided sentence in 76. The initial representation for the trigger sentence is shown in 78.

Example 78
Fred saw a friend of his.
((def,Fred) | (name (def,Fred) Fred)), λ(x)(see x [∃y: (and (friend y) (possess (his(x) y)) y)]

Given that the antecedent for his is Fred, this logical form is updated as shown in 79.

Example 79
Fred saw a friend of his.
((def,Fred) | (name (def,Fred) Fred)), λ(x)(see x [∃y: (and (friend y)
(possess (his(x) y)
(or (= (his(x) x) (= (his(x) (def,Fred))) y)

Now, before converting the existential into its functional form, we must decide which meaning of the pronoun is intended.

Suppose that the pronoun his refers indirectly to the subject, then, the logical form of the trigger is restricted as shown in 80.

Example 80
Fred saw a friend of his.
((def,Fred) | (name (def,Fred) Fred)), λ(x)(see x [∃y: (and (friend y)
(possess (his(x) y)
(= (his(x) x)) y))

Since λ(x) must have scope over the existential to bind the variable x in its restriction, the existential must be a function of that variable and can be used to provide the third meaning of the elided sentence in 76, as shown in 81.
In contrast, assume that the pronoun his refers directly to Fred. This choice is reflected in the logical form shown in 82.

Example 82
Fred saw a friend of his.

\begin{verbatim}
((def_{def}) | (name (def_{def}) Fred)), \lambda(x)(see x (indef_{x} x) | (and (friend (indef_{x} x))
(possess (his_{x} x) (indef_{x} x))
(= (his_{x} x) (def_{x}))))
\end{verbatim}

Ellipsis: George did too. (George saw a different friend of Fred's.)

\begin{verbatim}
((def_{def}) | (name (def_{def}) George)), \lambda(x)(see x (indef_{x} x) | (and (friend (indef_{x} x))
(possess (his_{x} x) (indef_{x} x))
(= (his_{x} x) (def_{x}))))
\end{verbatim}

We must still determine whether \(\lambda(x)\) has scope over the existential. If it does, we replace the existential variables with a function of \(x\) as shown in 83, allowing us to provide the second reading of the elided sentence in 76.

Example 83
Fred saw a friend of his.

\begin{verbatim}
((def_{def}) | (name (def_{def}) Fred)), \lambda(x)(see x [\exists y: (and (friend y)
(possess (his_{y} x) y)
(= (his_{x} x) (def_{def})))))
\end{verbatim}

Ellipsis: George did too. (George saw the same friend of Fred's.)

\begin{verbatim}
((def_{def}) | (name (def_{def}) George)), \lambda(x)(see x (indef_{x} x) | (and (friend (indef_{x} x))
(possess (his_{x} x) (indef_{x} x))
(= (his_{x} x) (def_{def}))))
\end{verbatim}

On the other hand, if \(\lambda(x)\) does not have scope over the existential, then we replace the existential variable with a skolem constant, allowing us to provide the first reading of the elided sentence in 76.

Example 84
Fred saw a friend of his.

\begin{verbatim}
((def_{def}) | (name (def_{def}) Fred)), \lambda(x)(see x (indef_{x} x) | (and (friend (indef_{x} x))
(possess (his_{x} x) (indef_{x} x))
(= (his_{x} x) (def_{def}))))
\end{verbatim}

Ellipsis: George did too. (George saw the same friend of Fred's.)

\begin{verbatim}
((def_{def}) | (name (def_{def}) George)), \lambda(x)(see x (indef_{x} x) | (and (friend (indef_{x} x))
(possess (his_{x} x) (indef_{x} x))
(= (his_{x} x) (def_{def}))))
\end{verbatim}

Hence, we are able to provide three readings for the elided sentence in 76. In contrast, our approach provides only two readings for the elided sentence in 77 as shown with a similar example in section 4.

In Klein’s [Kle87] approach to verb phrase ellipsis, definite and indefinite representations are provided only after quantifier scoping information is available. Hence, like us, Klein provides the same kind of final representation for both definite and indefinite noun phrases (ours is a function, his is a discourse referent). One can easily see the correspondence between discourse referents in boxes and functions with argument lists. Whenever a discourse referent is created in a box introduced by a universal, it is
like a function of the variable corresponding to that universal. However, Klein does not notice a very important scoping issue (i.e., does the lambda have scope over noun phrases in the verb phrase or not). In particular, whenever a discourse referent is created in a verb phrase box, it is like a function of the variable corresponding to the lambda operator used to abstract the subject. Hence, before representing a definite or indefinite in discourse representation theory, a decision about whether a discourse referent should be defined inside or outside of a verb phrase box must be made. To handle examples 76 and 77 in the framework of Discourse Representation Theory, Klein could stipulate that a discourse referent is created inside a box iff the operator responsible for introducing that box has scope over the noun phrase. This information must be known, however, before building the discourse model.

A Appendix: The Syntax and Semantics of Logical Form

In this appendix, we extend the syntax and semantics of first order logic (as given by Morgenstern [Mor88]). We lambda-abstract syntactic subjects for the purpose of handling the sloppy identity ambiguity in verb phrase ellipsis. In other words, we represent a sentence $S$ using the formula $\tau, \lambda(w)\psi$, where $\tau$ is a term representing the subject and $\lambda(w)\psi$ is the lambda function representing the verb phrase. However, the meaning of the formula $\tau, \lambda(w)\psi$ is simply $\psi(\tau/w)$, that is, $\psi$ with each free occurrence of $w$ replaced with $\tau$. Before providing the truth value for a sentence, we must specify antecedents for anaphoric noun phrases, specify quantifier scoping, and eliminate the abstraction operator by applying the subject to the verb phrase. However, before eliminating the lambda operator, we must determine the meanings of sentences with verb phrase ellipsis. The syntax and semantics of logical form are straightforward except we often use syntactic sugar and place quantifiers inside the predicate argument structure.

A.1 Syntax

1. The logical constants: $\neg, \forall, \land, \rightarrow, \leftarrow$, $\left(, =, \forall, \exists, ;, |, \lambda$. We often use English equivalents of the logical constants (e.g., *and* for $\land$).

2. Non-logical constants: These include numerical constants (e.g., 1, 2, etc.), character constants (e.g., A, b, etc.), non-numerical, non-character constants (e.g., Fred34).

3. Variables: For example, $x, y, z$.

4. Predicate symbols: For example, run, boy, etc.

5. Function symbols: For example, def$_i$, indef$_{66}$, his$_{66}$.

We will also characterize terms, a-terms, atomic formulas, well-formed formulas, and sentences.

1. Terms: A term is any expression that refers to an object. Formally, they are defined by the following rules:

   (a) If $\tau$ is a constant, $\tau$ is a term.

   (b) If $\tau$ is a variable, $\tau$ is a term.
(c) If \( \tau_1, \tau_2, \ldots, \tau_n \) are terms, and \( \phi \) is an n-ary function symbol, then \( (\phi \, \tau_1 \, \tau_2 \ldots \, \tau_n) \) is a term. We allow a function to have a restriction (which is a well-formed formula). This is indicated by creating a list consisting of the function, \( | \), and the restriction. For example, \( ((\phi \, \tau_1 \, \tau_2 \ldots \, \tau_n) \mid \psi) \)

2. Q-term: We introduce the idea of a q-term to the syntax. A q-term is a quantifier, restriction (which is a well-formed formula), and variable used as a pseudo-term in a formula. The meaning of a q-term will be introduced in the section on semantics. For example, \([\forall \alpha: \psi \, \alpha]\) is a q-term, where \( \alpha \) is a variable and \( \psi \) is a well-formed formula.

3. Atomic formula: If \( \pi \) is an n-ary predicate symbol, and \( \tau_1, \tau_2, \ldots, \tau_n \) are terms or q-terms, then \( (\pi \, \tau_1 \, \tau_2 \ldots \, \tau_n) \) is an atomic formula.

4. Well-formed formula: Well-formed formula are defined by the following formation rules:

   (a) If \( \phi \) is an atomic formula, then it is a well-formed formula.

   (b) If \( \phi \) is a well-formed formula \( \tau_1=\tau_2 \) and \( \tau_1 \) and \( \tau_2 \) are terms, then \( \phi \) is a well-formed formula.

   (c) If \( \phi \) is a well-formed formula, then \( \neg \phi \) is a well-formed formula.

   (d) If \( \phi \) and \( \psi \) are well-formed formulas, then (or \( \phi \, \psi \)), (and \( \phi \, \psi \)), (if \( \phi \, \psi \)), and (iff \( \phi \, \psi \)) are well-formed formulas.

   (e) If \( \phi \) is a well-formed formula and \( \alpha \) is a variable, then \( \forall \alpha \, \phi \) and \( \exists \alpha \, \phi \) are well-formed formula.

   (f) If \( \phi \) is a formula and \([\forall \alpha: \psi \, \alpha]\) or \([\exists \alpha: \psi \, \alpha]\) is a q-term in the formula, then that formula is well-formed.

   (g) If \( \phi \) is a well-formed formula and \( \tau \) is a term (corresponding to the syntactic subject of the represented sentence) in the formula, then \( \tau, \lambda(x)\phi' \) (where \( \phi' \) is \( \phi(x/\tau) \)) is a well-formed formula.

5. Sentences: Sentences are well-formed formulas that do not contain free variables. We must add that whenever we include a quantifier and its variable as a q-term in the predicate, then that quantifier is able to bind those variables inside the formula containing it.

A.2 Semantics

The model \( M \) for language \( L \).

1. Domain \( D \) of objects in the world.

2. A mapping assigning each non-logical constant of the language a member of the domain.

3. A mapping assigning each n-ary predicate of \( L \) a set consisting of n-tuples that can be formed out of elements of \( D \).

4. A mapping assigning each n-ary function of \( L \) a set of \( n+1 \) tuples formed from the elements of \( D \).

We define the value of a constant term \( \tau \) under interpretation \( M \) as follows:
1. If \( \tau \) is a constant, then the value of \( \tau \) under \( M \) is the element of \( D \) which \( M \) maps to \( \tau \).

2. If \( \tau \) is of the form \( \theta(\tau_1, \tau_2, \ldots, \tau_n) \) where \( \theta \) is an n-ary function symbol, \( \sigma_1 \) is the value for \( \tau_1 \), \( \sigma_2 \) is the value for \( \tau_2 \), \ldots, \( \sigma_n \) is the value for \( \tau_n \), and \( [\sigma_1, \sigma_2, \ldots, \sigma_n, \sigma] \) is an element in the set of \( n+1 \) tuples that \( M \) maps to \( \theta \), then \( \sigma \) is the value of \( \tau \).

To this we add:

1. If \( (\text{pro}; \tau_1 \tau_2 \ldots \tau_n) \) occurs in a formula and \( \tau_1, \ldots, \tau_n \) are terms and for any \( \tau_i \) which is a variable bound by \( \lambda(\tau_i) \), the lambda operator must have scope over the function. Also for any variable \( \tau_j \) not bound by a lambda operator, then if there is an operator over \( \tau_j \) contained as a q-term in the formula containing the pronoun function (or in a higher formula), then that operator \( \text{op}_j \) has scope over the pronoun function.

2. If \( (\text{def}; \tau_1 \tau_2 \ldots \tau_n) \) occurs in a formula \( \psi \) and \( \tau_1, \ldots, \tau_n \) are terms and for any \( \tau_i \) which is a variable bound by \( \lambda(\tau_i) \), then the lambda operator has scope over the function. Also for any variable \( \tau_j \) not bound by a lambda operator, then if there is an operator over \( \tau_j \) contained as a q-term in the same formula (or in a higher formula) as the function but outside of the function's restriction, then that operator \( \text{op}_j \) has scope over the function. On the other hand, if for any \( \tau_j \) which is a variable whose operator is a universal or existential contained in the function's restriction, the function receives the following meaning:

\[
(\text{or}[\text{op}_j\tau_j \ (\text{def}; \tau_1 \tau_2 \ldots \tau_j \ldots \tau_n)] \ (\text{def}; \tau_1 \tau_2 \ldots \tau_{j-1} \tau_{j+1} \ldots \tau_n))
\]

What it means for a sentence \( \phi \) to be true under an interpretation \( M \).

1. If \( \phi \) is an atomic sentence (i.e., \( \phi \) is of the form \( (\pi \tau_1, \ldots, \tau_n) \) where \( \sigma_1 \) is the value of \( \tau_1 \), \ldots, and \( \sigma_n \) is the value of \( \tau_n \), then \( M \models \phi \) if and only if \( [\sigma_1, \ldots, \sigma_n] \) is a member of the set which \( M \) assigns to \( \pi \).

2. If \( \phi \) has the form \( \neg \omega \), where \( \omega \) is an atomic sentence of the form \( \pi(\tau_1, \ldots, \tau_n) \) and \( \sigma_1 \) is the value of \( \tau_1 \), \ldots, and \( \sigma_n \) is the value of \( \tau_n \), then \( M \models \phi \) if and only if \( [\sigma_1, \ldots, \sigma_n] \) is in the antiextension of the set which \( M \) assigns to \( \pi \).

3. If \( \phi \) has the form \( \psi \lor \chi \), \( M \models \phi \) if and only if \( M \models \psi \) or \( M \models \chi \) or both.

4. If \( \phi \) has the form \( \psi \land \chi \), \( M \models \phi \) if and only if \( M \models \psi \) and \( M \models \chi \).

5. If \( \phi \) has the form \( \psi \rightarrow \chi \), \( M \models \phi \) if and only if \( M \models \neg \psi \) or \( M \models \chi \) or both.

6. If \( \phi \) has the form \( \psi \leftrightarrow \chi \), \( M \models \phi \) if and only if \( M \models \psi \) and \( M \models \chi \) or \( M \models \neg \psi \) and \( M \models \neg \chi \).

7. For sentences of the form \( \forall \alpha \psi \) or \( \exists \alpha \psi \), we use \( \beta \)-variants. If \( M \) and \( M' \) are interpretations with identical domains, and \( \beta \) is a constant, \( M \) is a \( \beta \)-variant of \( M' \) if \( M \) and \( M' \) differ only in what they assign to \( \beta \).

1) If \( \phi \) has the form \( \forall \alpha \psi \), \( M \models \phi \) if and only if, for all \( M' \), if \( M' \) is a \( \beta \)-variant of \( M \), \( M' \models \psi(\beta/\alpha) \), where \( \psi(\beta/\alpha) \) is the expression obtained by substituting \( \beta \) for all free occurrences of \( \alpha \) in \( \psi \).

2) If \( \phi \) has the form \( \exists \alpha \psi \), \( M \models \phi \) if and only if, for some \( M' \), if \( M' \) is a \( \beta \)-variant of \( M \), \( M' \models \psi(\beta/\alpha) \).
\[ \psi(\beta/\alpha), \] where \( \psi(\beta/\alpha) \) is the expression obtained by substituting \( \beta \) for all free occurrences \( \alpha \) in \( \psi \).

8. If \( \phi \) is a formula \( \tau, \lambda(x)\psi \), then the \( M \models \phi \) if and only if \( M \models \psi(\tau/x) \).

Additionally:

1. If \((\phi \tau_1 \tau_2 \ldots \tau_n | \psi_1)\) is contained in a formula \( \psi_2 \), the formula is equivalent to (and \( \psi_3 \psi_1 \)), where \( \psi_1 \) is removed from \( \psi_2 \) to give \( \psi_3 \).

2. If \((\pi [\text{op}_1 \alpha_1: \psi] \ldots)\) is a formula, then this is equivalent to:
   \( \text{op}_1 \) (and \( \pi (\alpha_1 \ldots) \)) if \( \text{op}_1 \) is \( \exists \) or \( \text{op}_1 \) (if \( \pi (\alpha_1 \ldots) \)) if \( \text{op}_1 \) is \( \forall \).

3. If \((\pi \ldots [\text{op}_j \alpha_j: \psi_j \alpha_j] \ldots [\text{op}_i \alpha_i: \psi_i \alpha_i] \ldots) \) and \( \alpha_j \) is free in \( \psi_i \) then \( \text{op}_j \alpha_j \) must have scope over \( \text{op}_i \).

4. If \( \tau_1, \lambda(\alpha_1)(\pi \ldots [\text{op}_i \alpha_i: \psi_i \alpha_i] \ldots) \) and \( \alpha_1 \) is free in \( \psi_i \) then \( \lambda(\alpha_1) \) must have scope over \( \text{op}_i \).

5. \( \pi [\text{op}_1 \alpha_1: \psi_1 \alpha_1] [\text{op}_2 \alpha_2: \psi_2 \alpha_2] \ldots [\text{op}_n \alpha_n: \psi_n \alpha_n] \) is equivalent to the disjunction of formulae whose restrictions are expanded as indicated in 2 and whose quantifiers form a legal partial order of the quantifiers \( \text{op}_1 \alpha_1 \text{op}_2 \alpha_2 \ldots \text{op}_n \alpha_n \) (given constraints 3 and 4 above).

6. If \( \tau_1, \lambda(\alpha_1)(\pi \ldots [\text{op}_i \alpha_i: \psi_i \alpha_i] \ldots) \) and \( \text{op}_i \) is an existential operator, then either replace every \( \alpha_i \) with a function of \( \alpha_1 \) and every variable corresponding with an operator that has scope over it or replace every \( \alpha_i \) with a function of those variables corresponding with operators that have scope over it excluding \( \alpha_1 \).

References

[All87] James Allen. *Natural Language Understanding*. The Benjamin/Cummings Publishing Company, Menlo Park, CA, 1987.

[AvE89] H. Alshawi and Jan van Eijck. Logical forms in the core language engine. In *The Proceedings of the 27th Annual Meeting of ACL*, pages 25-32, June 1989.

[BP80] Emmon Bach and Barbara Partee. Anaphora and semantic structure. In Jody Kreiman and A. E. Ojeda, editors, *Papers from the Parasession on Pronouns and Anaphora*. Chicago Linguistic Society, Chicago IL, 1980.

[CM85] E. Charniak and D. McDermott. *Introduction to Artificial Intelligence*. Addison-Wesley, Reading, MA, 1985.

[DWP81] D. R. Dowty, R. E. Wall, and S. Peters. *Introduction to Montague Semantics*. D. Reidel Publishing Company, Boston, MA, 1981.

[Eva80] Gareth Evans. Pronouns. *Linguistic Inquiry*, 11:337-362, 1980.

[Gea62] Peter T. Geach. *Reference and Generality*. Cornell University Press, Ithaca, 1962.

[Har88] Mary P. Harper. Representing pronouns in logical form: Computational constraints and linguistic evidence. In *The Proceedings of the 7th National Meeting of AAAI*, 1988.

[Har89] Mary P. Harper. The representation of noun phrases in logical form. Technical Report CS-89-16, Brown University, 1989.

[Har90] Mary P. Harper. *The representation of noun phrases in logical form*. PhD thesis, Brown University, 1990.
[Hei82] Irene Heim. The Semantics of Definite and Indefinite Noun Phrases. PhD thesis, University of Massachusetts, 1982.

[Hin79a] J. Hintikka. Quantifiers in logic and quantifiers in natural language. In E. Saarinen, editor, Game Theoretical Semantics. D. Reidel, Dordrecht, 1979.

[Hin79b] J. Hintikka. Quantifiers vs quantification theory. In E. Saarinen, editor, Game Theoretical Semantics. D. Reidel, Dordrecht, 1979.

[Hin85] J. Hintikka. Anaphora and Definite Descriptions: Two applications of Game-Theoretical semantics. D. Reidel Publishing Company, Boston, 1985.

[Hor84] Norbert Hornstein. Logic as Grammar: An Approach to Meaning in Natural Language. MIT Press, Cambridge, MA, 1984.

[Kam81] Hans Kamp. A theory of truth and semantic representation. In Jeroen Groenendijk, Theo Janssen, and Martin Stokhof, editors, Formal Methods in the Study of Language, volume 1. Mathematische Centrum, Amsterdam, 1981.

[Kar69] L. Kartunnen. Pronouns and variables. CLS, 5, 1969.

[Kle87] Ewan Klein. VP ellipsis in DR theory. In J. Groenendijk, D. de Jongh, and M. Stokhof, editors, Studies in Discourse Representation and the Theory of Generalized Quantifiers. Foris, Dordrecht, 1987.

[Lew79] D. Lewis. Score-keeping in a language game. In R. Bäuerle, U. Egli, and A. von Stechow, editors, Semantics from different points of views. Springer, Berlin, 1979.

[May85] Robert May. Logical Form: Its Structure and Derivation. MIT Press, Cambridge, MA, 1985.

[Mon70] R. Montague. Universal grammar. Theoria, 36:373--398, 1970.

[Mor88] L. Morgenstern. Foundations of a Logic of Knowledge, Action, and Communication. PhD thesis, New York University, 1988.

[PB81] Barbara Partee and Emmon Bach. Quantification, pronouns, and VP; anaphora. In Jeroen Groenendijk, Theo Janssen, and Martin Stokhof, editors, Formal Methods in the Study of Language, volume 1. Mathematische Centrum, Amsterdam, 1981.

[PP88] M. E. Pollack and F. C. N. Pereira. An integrated framework for semantic and pragmatic interpretation. In The Proceedings of the 26th Annual Meeting of ACL, pages 75--86, June 1988.

[Rei83] T. Reinhart. Anaphora and Semantic Interpretation. Croom Helm, London, 1983.

[Rob87] Craig Roberts. Modal Subordination, Anaphora, and Distributivity. PhD thesis, University of Massachusetts, 1987.

[Ros67] J. R. Ross. Constraints on Variables in Syntax. PhD thesis, MIT, 1967.

[Rus71] B. Russell. Reference. In J. F. Rosenberg and C. Travis, editors, Readings in the Philosophy of Language. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1971.

[Sag76] Ivan A. Sag. Deletion and Logical Form. PhD thesis, MIT, 1976.

[SP84] L. K. Schubert and F. J. Pelletier. From English to Logic: Context-free computation of 'conventional' logical translations. American Journal of Computational Linguistics, 10:165--176, 1984.

[SZZar] P. Sells, A. Zaenen, and D. Zec. Reflexivization variation: Relations between syntax, semantics, and lexical structure. In M. Iida, S. Wechsler, and D. Zec, editors, Studies in Grammatical Theory and Discourse Structure, volume 1. CSLI, Stanford, to appear.

[Web78] B. L. Webber. A Formal Approach to Discourse Anaphora. PhD thesis, Harvard, 1978.

[Web83] B. L. Webber. So what can we talk about now? In M. Brady and R. Berwick, editors, Computational Models of Discourse. MIT Press, Cambridge MA, 1983.

[Wil77] Edwin S. Williams. Discourse and logical form. Linguistic Inquiry, 8:161--139, 1977.