SRI International

Technical Note 490 • May 1990

INCREMENTAL INTERPRETATION

Prepared by:

Fernando C. N. Pereira
AT&T Bell Laboratories

Martha E. Pollack
Artificial Intelligence Center
Computing and Engineering Sciences Division
and
Center for the Study of Language and Information
Stanford University

This paper will appear in *Artificial Intelligence*.

The research was funded by the Defense Advanced Research Projects Agency under Contract N000-39-84-C-0524. The preparation of this paper was partly supported by a gift from the System Development Foundation as part of a coordinated research effort with the Center for the Study of Language and Information, Stanford University.

333 Ravenswood Avenue • Menlo Park, CA 94025-3493 • (415) 326-6200 • FAX: (415) 326-5512 • Telex: 334486

Incremental Interpretation

Fernando C. N. Pereira

AT&T Bell Laboratories
600 Mountain Ave.
Murray Hill, NJ 07974

Martha E. Pollack

Artificial Intelligence Center
and
Center for the Study of Language and Information
SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025

This paper will appear in *Artificial Intelligence*.

Abstract

We present a system for the incremental interpretation of natural-language utterances in context. The main goal of the work is to account for the influences of context on interpretation, while preserving compositionality to the extent possible. To achieve this goal, we introduce a representational device, conditional interpretations, and a rule system for constructing them. Conditional interpretations represent the potential contributions of phrases to the interpretation of an utterance. The rules specify how phrase interpretations are combined and how they are elaborated with respect to context. The control structure defined by the rules determines the points in the interpretation process at which sufficient information becomes available to carry out specific inferential interpretation steps, such as determining the plausibility of particular referential connections or modifier attachments. We have implemented these ideas in Candide, a system for interactive acquisition of procedural knowledge.

1 Introduction

The interpretation of natural-language utterances in context is riddled with representational and computational difficulties. One way to organize the interpretation process is to adopt a "divide and conquer" approach, in which the interpretation of an utterance is systematically determined from the interpretation of its parts, whose interpretations are, in turn, determined from the interpretations of their parts. This paper presents just such an approach, which we call *incremental interpretation.*

Clearly, the ultimate contribution of a phrase to the interpretation of an utterance containing it is not determined by the phrase alone. Instead, its contribution will depend, in general, upon its linguistic context, that is, the syntactic relation between the phrase and the rest of the utterance; its discourse context, the relation between the utterance and the discourse embedding it; and its world context, the relation between the entire discourse, its parts, and the facts of the world. Therefore, incremental interpretation raises three interconnected problems:

Analysis: What is the informational content of a phrase in isolation?

Representation: How can one represent that content so that it can interact in the necessary ways with the phrase's linguistic, discourse, and world contexts?

Control: At what points in the interpretation process is the content of a phrase combined with different kinds of contextual information to develop the ultimate contribution of the phrase to the interpretation?

A minimalist answer to the above questions is the *compositional* approach to interpretation, arising from the philosophy of language [29, 25] and widely explored in linguistic [11, 7] and in computational settings [38, 37]. Compositional interpretation assigns to each phrase a *denotation*, a set theoretic object characterizing the contribution of the phrase to the truth conditions of sentences containing it, and makes the denotation of a complex phrase a function of the denotations of its immediate constituents. With this analysis and repre-

sentation of phrase interpretations, control becomes simply the application of appropriate functions following directly the structural decomposition of a sentence into phrases.

Compositional interpretation is extremely attractive for its simplicity, mathematical grounding, and power. However, to accommodate the contextual influences on phrase interpretation, compositional denotations must become very complex. This is true even when only the effects of linguistic context are considered. For example, Cooper provides a compositional treatment of quantified noun phrases [7]. The scopes of quantifiers arising in noun-phrase interpretation cannot be determined until the noun phrases are incorporated into larger phrases. Therefore, the denotations used in Cooper's system must directly encode the combination possibilities of the quantifiers that arise during the interpretation of quantified noun phrases. The resulting denotations are not only awkward, but they lack the logical status originally intended for compositional denotations. Furthermore, it is difficult to see how this approach can be extended to deal with contextual interactions of a more open-ended nature, for example, the interactions between quantification and reference.

Cooper's work, as well as the earlier, but quite similar, computational treatment of quantifier scoping due to Woods [45], relies on the important observation that a quantified phrase provides two distinct contributions to the interpretation of a sentence: a matrix, which combines compositionally with the rest of the sentence, and a quantifier store, which encodes the scoping potential of the quantifiers in the phrase. Our work can be seen as an extension and application of this insight to a wider range of interpretation questions, including those that involve referring expressions, anaphora, ambiguous attachments, and ambiguous modifying expressions.

In our interpretation system, the contribution of a phrase is represented by a *conditional interpretation*, which has two parts: a *sense*, which participates compositionally in the interpretation of larger phrases, and a set of *assumptions*, which represent constraints on how the sense may be further connected to its context. Incremental interpretation involves two interleaved processes: building conditional interpretations for phrases, and elaborating conditional interpretations with respect to context by *discharging* interpretation assumptions.

It is important to distinguish two different aspects of incremental interpretation. The *combinatorial* aspect, which is our main concern in this work, makes explicit the interpretation alternatives that arise from the interaction between phrase interpretations and contextual information. The *inferential* aspect concerns the means by which a language interpreter chooses amongst combinatorial alternatives. Consider, for example, the interpretation of the phrase "his" in an utterance, "He asked each guest to tell him his story". The problem can be decomposed into two parts: identification of the referential possibilities for the phrase, and the determination that a particular possibility is the most likely one. Any interpretation system must, of course, deal with both combinatorial and inferential issues. A number of computational linguistics systems have concentrated on the inferential, rather than the combinatorial, side of the interpretation problem [28, 9, 22]. Others have focused on the combinatorial aspects of single-sentence interpretation [45, 38, 32, 37]. We do not claim any particular novelty in the inferential machinery that our system relies on to produce interpretations. Rather, our goal has been to develop a detailed model of the combinatorial aspects of the incremental interpretation of utterances in context, one that shows how a range of inferential processes can be coordinated with one another, and how they can be provided with the information they need to solve their problems at the appropriate times.

2 A Framework for Incremental Interpretation

2.1 Candide

We have exercised and illustrated our theory of incremental interpretation by using it in the Candide system for knowledge acquisition. Incorporating both a graphical interface and a processor for English discourse, Candide is designed to be used in the construction and maintenance of knowledge bases for the Procedural Reasoning System (PRS) [12]. In Candide, a procedure is described by a network whose nodes represent states in the procedure's execution and whose arcs represent state transitions described by suitable arc annotations. Each procedural network also contains *invocation conditions*, which encode [pre]conditions that must be satisfied for the procedure to be used, as well as [post]conditions that are made true by the successful performance of the procedure. To create a procedural network, a Candide user draws arcs and nodes with the graphical editor, and specifies arc invocation conditions and arc annotations with English sentences. Each sentence is interpreted using the methods presented in this paper, yielding a logical representation of the information it expresses; this is then translated into the specialized language of procedural networks supported by PRS. The current version of Candide has been used to construct networks that represent malfunction procedures for NASA's space shuttle.

Candide is a tool for building procedural networks; there is no dialogue involved when the networks are used by PRS to perform particular procedures. Figure 1 depicts a fragment of a very simple procedural network. On each arc, we have included both the English query and the logical representation that results from Candide's interpretation process. The network contains examples of the three types of arc supported by PRS: assertional arcs, whose annotations are prefixed by "—", achievement arcs, prefixed by "!", and query arcs, prefixed by "?".

The type of an arc determines the conditions under which it will be traversed when PRS executes the procedure. An assertional arc specifies some assertion that should be added

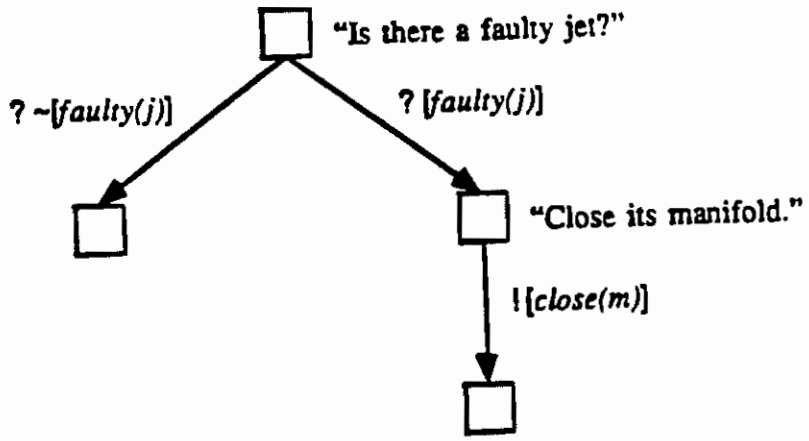

Figure 1: A Procedural Network

to the PRS knowledge base if the arc is traversed. A Candide user will typically express an assertion arc using a declarative sentence, such as "The jet failed." This sentence expresses the assertion to be added to the database.

An achievement arc calls for some action to be performed if the arc is traversed. Achievement arcs are expressed using imperative sentences. Finally, a query arc specifies a condition whose truth is prerequisite to the arc's traversal; these arcs are expressed using an interrogative sentence. In practice, query arcs are created in pairs in Candide, one corresponding to the truth of a question, and the other to its falsity. Wh-questions have no direct counterpart in Candide.

Invocation conditions may either be assertions, which denote propositions that, when true, indicate that the procedure is relevant, or they may be imperatives, which denote goals that are likely to be satisfied as a result of performing the procedure.

We defer until later a detailed explanation of the complete interpretations shown in Figure 1, but it is important to note one general point about procedural discourse. (For

more familiar but similar examples, consider recipes, operating instructions, and the like.) For the most part, reference resolution in these types of dialogue does not involve actual entities in the universe of discourse, but rather parameters of the procedure whose value will not be known until the procedure is executed. Thus, in Figure 1, the word "its" in the phrase "its manifold" should be resolved to a parameter introduced by the definite noun phrase "the jet". When the procedure is executed, the parameter will be instantiated to some particular jet that failed. In the network, we see that "the jet" was resolved to some entity j, and "its manifold" was resolved to some entity m; as we will see later, the discourse model will contain the information that m is the unique manifold attached to j. Most of the reasoning involved in the interpretation of procedural discourse is about sorts rather than about specific entities and their individual properties.[1]

Figure 1 exemplifies only a small subset of the range of syntactic, semantic, and pragmatic phenomena handled by the Candide system. We present it here to illustrate the Candide application. In this rest of this section, we will use simple phrases similar to those in Figure 1 to describe in general our incremental interpretation framework. In Section 3, we will turn to a number of other linguistic phenomena not represented in the figure.

2.2 Conditional Interpretations

The process of incremental interpretation assigns *conditional interpretations* to phrases. Conditional interpretations separate the context-independent aspects of an interpretation from those that are context-dependent—that is, they separate the part of the information that is invariant with respect to further incremental interpretation from the part that may vary.

A conditional interpretation has two parts: a sense and a set of assumptions, which represent constraints on how the sense may be further connected to the context. We denote by $A : s$ a conditional interpretation with sense s and set of assumptions A. If A and B

[1]The major exceptions to this are a small class of proper names and definite references that resolve to specific entities in the domain of discourse.

are assumption sets and a and b are assumptions, we use the simplified notation A, B for $A \cup B$; we use A, a for $A \cup \{a\}$; and we use a, b for $\{a, b\}$. A conditional interpretation can be elaborated by assumption discharge, which adds new constraints on senses, parameters they may contain, and the discourse context. We say that a conditional interpretation is *complete* when its set of assumptions is empty.

The present version of our interpretation model provides two types of assumptions: *bind* assumptions and *restrict* assumptions. A bind assumption introduces a new parameter in an interpretation and constrains the way in which the parameter may receive its value from the context. The discharge of a bind assumption eliminates the parameter in one of three ways: replacement by a specific value drawn from the context, replacement by another parameter, or binding the parameter with a binding operator such as a quantifier.

Bind assumptions are most typically introduced in the interpretation of noun phrases. The general form of a bind assumption is $\mathsf{bind}(x, C, T)$ where x is a parameter, C specifies linguistic constraints on the value of this parameter (for example, it may indicate that the parameter is associated to a phrase with a definite determiner), and T gives the sort of the parameter x (its domain of possible values). For example, a conditional interpretation for the noun phrase "the jet" might be

$$\mathsf{bind}(x, \mathsf{def}, \mathit{jet}) : x.$$

This interpretation states that the sense of "the jet" is x where the value of x will be an entity of sort *jet* determined from the context according to the constraints of definite reference.

Restrict assumptions are introduced during the interpretation of phrases that complement or modify other phrases. Prepositional phrases (PPs) provide one example. Syntax does not fully specify what phrase a given PP modifies, as evidenced by PP-attachment ambiguities. During the conditional interpretation of a phrase containing a PP, we cannot completely specify the informational contribution of the PP until we have considered the contribution of larger constituents of the phrase.

Consider the phrase "car in the park", which we shall call Q; let the constituent "in the park" be identified as P. The conditional interpretation for Q will contain a restrict assumption, which signifies that the syntactic role of one of the phrase's constituents, P, is underspecified. It further signifies that P will contribute certain information either to the overall interpretation of Q or to that of some larger phrase containing Q. For instance, Q may be part of the larger phrase "man near the car in the park", and P may contribute to this entire phrase by modifying "man". When the role of P is determined, the restrict assumption is discharged, and the sense associated with the larger phrase is elaborated to reflect P's contribution. In this example, if we determine that "in the park" does in fact modify "car", then the restrict assumption introduced by the PP is discharged, and the sense of Q is updated accordingly.

The general form of a restrict assumption, introduced during the interpretation of a phrase Q that includes a modifying phrase P, will be $\mathsf{restrict}(R, C)$. The *role* component R represents P's underspecified syntactic role: for example, a PP. The *constraint* component C specifies how P affects the interpretation of some other phrase Q', where P and Q' are related by R. Note that Q' may or may not be the same phrase as Q. Thus, the phrase "car in the park" may be given the conditional interpretation

$$\mathsf{restrict}(\mathsf{pp}(in, i), x), \mathsf{bind}(x, \mathsf{def}, park) : car \quad .$$

Here, i is a unique *index* for the phrase " in the park". We will use indices to enforce certain syntactic constraints on modification: the details of this will be explained later. The interpretation given here states that the sense of the phrase "man in the park" is the sort *man*, subject to the constraint that the phrase with index i modifies some as yet unspecified phrase as a PP with prepositional object x. The bind assumption for x is similar to the one in the previous example.

2.3 The Interpretation Process

We will now outline the overall process of incremental interpretation. To ground the process, we make use of a *least-commitment grammar*, which parses utterances and produces *analysis trees* that are neutral with respect to context-dependent attachment decisions. The grammar we use in Candide, for instance, leaves quantifiers in place (following [45]), attaches all prepositional phrases low and right (following [32]), and brackets to the right all compound nominals.[2] We also employ a *discourse model*, which is an encoding of the contextual influences on a phrase that are a result of its surrounding discourse. As shown in Figure 2, the incremental interpretation process operates on a least-commitment analysis tree for an utterance, along with an initial discourse model, to produce a complete interpretation for the utterance, along with a resulting discourse model.

Our interpretation system is based on two kinds of rules: *structural rules*, which build the conditional interpretation of a phrase from the conditional interpretations of its constituents, and *discharge rules*, which elaborate conditional interpretations by discharging assumptions.

To interpret a phrase, Candide applies structural and discharge rules to construct a *derivation* of a conditional interpretation for the phrase from the conditional interpretations of the constituents of the phrase. A derivation for a phrase is a tree in which each node represents the conditional interpretation of a constituent of the phrase; moreover, the conditional interpretation represented by every node results from the application of a structural or discharge rule to the node's daughters. The root of the derivation is an interpretation of the whole phrase. If the rule deriving a particular node is structural, the daughters of that node will correspond to the interpretations of the immediate constituents of the phrase whose interpretation is represented by the mother node. On the other hand, if the rule deriving a node is a discharge rule, then the mother node and its single daugh-

[2] We consider low attachment and right bracketing to be convenient encodings of least-commitment analyses, since all the other attachment or bracketing possibilities can be reconstructed from them. However, this should be seen as a temporary design decision, because there are reasons to suspect that ultimately syntactic analysis should be incorporated into the same stage of processing as semantic and pragmatic analysis. In particular, it is difficult to develop syntactically neutral representations for certain constructions such as coordination [27].

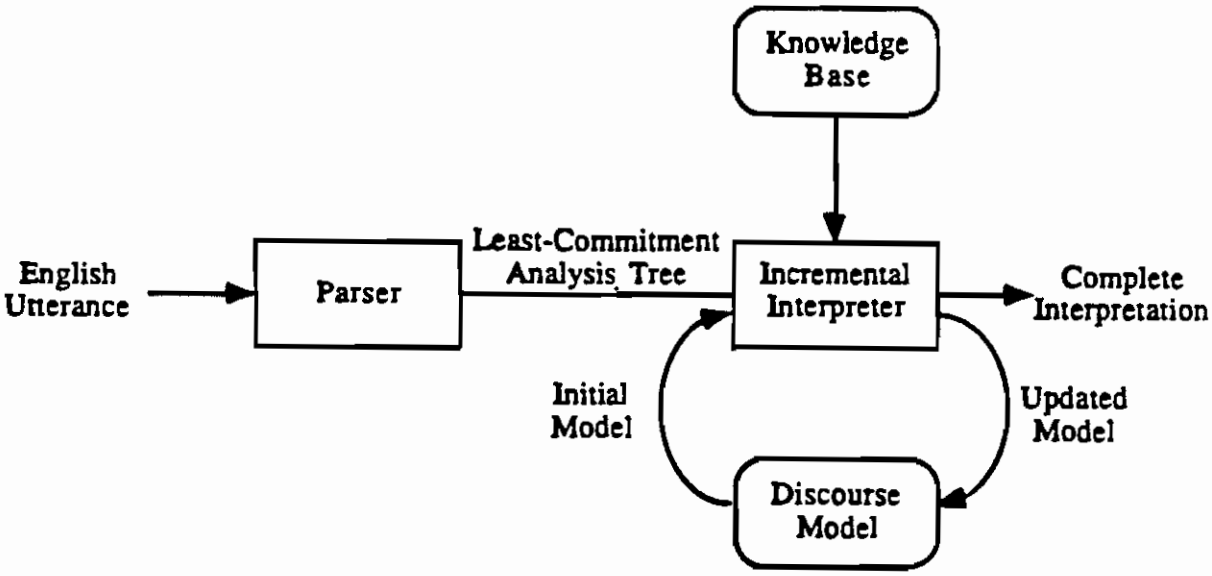

Figure 2: The Architecture of an Incremental Interpretation System

ter will be interpretations of the same phrase, but the mother node interpretation will be elaborated with respect to the daughter node interpretation.

Let D be a derivation of an interpretation for a phrase P. Then the above definition implies that each node n of D is the interpretation of some constituent Q of P. Furthermore, each descendant n' of n is associated with Q or one of Q's constituents. It is thus convenient to think of derivations as being related to syntactic analyses in the way depicted in Figure 3. The tree on the left of the figure is a simplified syntactic analysis of the phrase "the jet", given in the feature-value notation common to various unification-based grammar formalisms. (Our grammar was written for the PATR-II unification-grammar system [39]). The tree on the right is an associated derivation. The horizontal dashed lines associate derivation nodes to phrases whose interpretations they represent. Node 1 of the analysis tree is associated to two derivation nodes, $1'$ and $1''$. Node $1''$ is the result of applying to node $1'$ a discharge rule that identifies a possible referent for the noun phrase. The

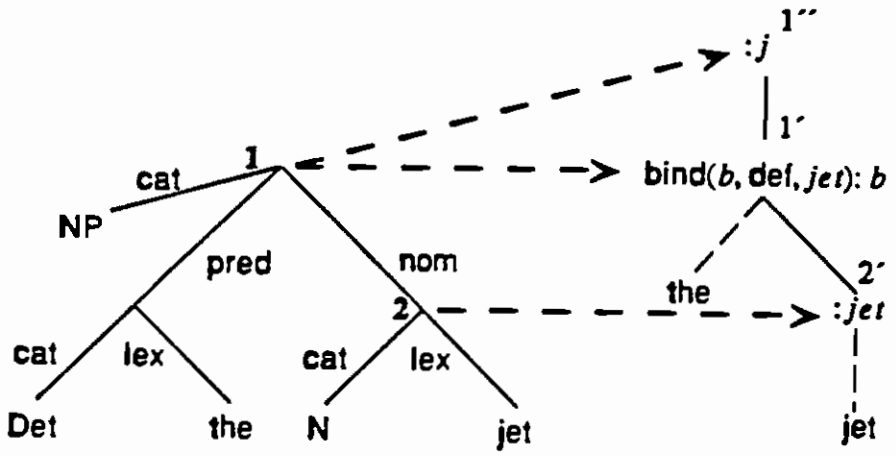

Figure 3: Syntactic Analysis and Semantic Derivation

interpretation at node 1′ is the result of applying a structural rule for syntactically definite noun phrases, which takes as input the interpretation for the noun in node 2′. We indicate access to lexical information in a derivation by dashed lines connecting the appropriate node to a word. Thus, the semantic lexicon is accessed to obtain the interpretation in 2′ of the word "jet". The word "the" is not given a separate interpretation, but instead participates in the licensing of the structural rule applied at node 1′.

It is clear from the foregoing discussion that the relation between phrases and their interpretations is central to the derivation process. We will use the symbol $\rightsquigarrow$ for that relation, and write $p \rightsquigarrow A : i$ to mean that phrase p has $A : i$ as its conditional interpretation. This is actually an oversimplification in that the relationship between phrases and their interpretations both affects and is affected by the discourse model. Thus, strictly speaking, we should have a 4-place interpretation relation between phrases, their interpretations, input discourse models, and resulting discourse models. However, only the discharge rules

use and update the discourse model; the parts of a derivation involving structural rules serve as conduits that pass discourse model fragments between constituents so that the appropriate discourse information is made available to discharge nodes. Therefore, we will not burden our notation with explicit mention of input and output discourse models. Instead, we will explicitly describe how specific discharge rules interact with the discourse model. For structural rules, we use a simple default threading mechanism for discourse model propagation.

It is important not to confuse analysis trees with derivation trees. The former result from parsing an utterance, and serve as input to the incremental interpretation process. The latter can be thought of as traces of the incremental interpretation process.

2.3.1 Structural Rules

Structural rules define the interpretation relation $\rightsquigarrow$ for a phrase in terms of the same relation for the phrase's immediate constituents. The general form of a structural rule is

$$\mathcal{P} \rightsquigarrow A : s \text{ if } P_1 \rightsquigarrow A_1 : s_1 \text{ and } \cdots \text{ and } P_k \rightsquigarrow A_k : s_k \tag{1}$$

where $\mathcal{P}$ is a pattern, containing variables P_1 through P_k, that describes a type of node in an analysis tree. As usual for such rules, the term to the left of the "if" is the *consequent* of the rule, while the terms to the right are the *antecedents*.

For example, the structural rule for definite-noun phrase interpretation used at node $1'$ in Figure 3 is

$$\textsf{def-np}(\textsf{nom} = N) \rightsquigarrow A, \textsf{bind}(x, \textsf{def}, s) : x \text{ if } N \rightsquigarrow A : s \tag{2}$$

The pattern def-np(nom = N) specifies the kind of phrase to which this rule applies. A matching phrase must belong to the def-np class of phrases, as defined in the grammar; it will have a feature nom, whose value N is the phrase whose interpretation is picked up in the antecedent of the rule. The rule states that if a phrase is a definite noun phrase

whose nominal part has the interpretation s under assumptions A, then the interpretation of the whole noun phrase is the new parameter x under the union of the assumptions A with a new bind assumption that requires x to be bound according to the constraints of definite reference to an entity of sort s. In Figure 3, the nominal N is the word "jet"; its interpretation is : *jet*; therefore, the interpretation of the whole definite noun phrase "the jet" is $\mathsf{bind}(x, \mathsf{def}, \mathit{jet}) : x$.

It is important to note that rules such as (2) are *schematic* in the sense that each use of the rule should be thought of as involving an appropriate new substitution instance of the rule. In particular, the parameter x in a new bind assumption $\mathsf{bind}(x, C, T)$ should be thought of as a metavariable standing at each use of the rule for a new parameter that does not occur elsewhere in the derivation except for ancestors of the rule's application. The same applies to discharge rules and lexical entries. In derivation examples that follow, we will thus rename variables as needed to avoid naming clashes.

As we remarked earlier, the only role of structural rules with respect to the discourse model is to propagate discourse model information to the discharge rules that use it. This is done by a simple *threading* arrangement: given a rule of the general form in (1), the input discourse model for the phrase P is just the input discourse model for phrase P_1, the output discourse model for P is the output discourse model for phrase P_k, and the output discourse model for each phrase P_i, $1 \leq i < k$ is identified with the input discourse model for phrase P_{i+1}. In this way, any discourse model updates made in derivation subtrees that are descendants of a node associated to P_i will be propagated to the right-hand sisters of P_i. Clearly, the effect of this threading depends on the ordering of the antecedent conditions for the rule, and determines how syntax affects the availability of contextual information at different stages in the interpretation. This is important in treating such matters as constraints on coreference, but in this work we adopted a simplistic left-to-right threading, which is sufficient for the issues we address.

2.3.2 Discharge Rules

Discharge rules increment the conditional interpretation of a phrase by specifying how assumptions in the interpretation may be eliminated with respect to the context of utterance, yielding a new interpretation and an updated context.

For example, in Figure 3, a discharge rule for definite reference applies to node $1'$ to produce node $1''$. The rule discharges an assumption with general form $\mathsf{bind}(x, \mathsf{def}, T)$ provided that there is some unique contextually available entity e of sort T.[3] The rule will thus map a conditional interpretation $A, \mathsf{bind}(x, \mathsf{def}, T) : s$ into $A[x/e] : s[x/e]$, that is, the contextually unique entity e is replaced for x in A and s.[4] Furthermore, the discourse model will be updated to record this particular mention of e. In the example, applying the rule to the interpretation

$$\mathsf{bind}(b, \mathsf{def}, jet) : b$$

and a discourse model containing a single entity j of sort jet will result in the complete interpretation

$$: j$$

together with an updated discourse model in which this particular mention of the jet j has been recorded.

In general, discharge rules have the form

$$P \rightsquigarrow A' : s' \text{ if } P \rightsquigarrow A : s$$

that is, the phrase under consideration is the same for both antecedent and consequent. As for the rule's use of the discourse model, the input model for the consequent is the same

[3] Contextual availability will in general take into account syntactic factors, such as forbidden coreference constraints, as well as discourse factors.

[4] We will use the notation $e[x/t]$ to represent the result of replacing all occurrences of x in e by t.

as for the antecedent, while the output model for the consequent is the result of updating appropriately the output model provided by the antecedent of the rule.

2.3.3 Rule Application

Structural rules are obligatory in that some structural rule associated with a given syntactic phrase type must be applied to any phrase of that type. In contrast, the application of discharge rules is optional, although discharging a particular assumption too early or too late may lead to a dead end in the interpretation process. Applying the same discharge rule at different points in the interpretation process for some utterance may lead to alternative interpretations, as we shall illustrate with the examples in Section 3.4.

Given a sentence and its syntactic analysis, the interpretation process applies structural and discharge rules, according to their applicability conditions, to construct the derivation of a complete interpretation of the sentence. In Candide, this process resembles a syntax-directed translation system [1]. Interpretation starts at the root node of the analysis tree. For each node of the tree, the interpretation process selects a structural rule and calls itself recursively for each of the node's daughters. Interpretations are constructed on return from the recursion, and discharge rules are optionally applied in a discharge cycle that follows each application of a structural rule to a node.

The nondeterministic nature of the discharge-rule application stands in contrast to the application of analysis rules in, say, a context-free grammar, but it recalls the use of inference rules in a proof system. And, as is the case in proof systems, alternative orderings of discharge-rule application can lead to different results. Within Candide, we control the nondeterminism by means of some relatively simple heuristics, such as early discharge of assumptions and bounds on assumption percolation wherever it can be shown that an assumption would not be dischargeable outside a certain syntactic domain.

2.4 Semantic Representation

Our approach to semantic interpretation is not tied to a specific choice of representation formalism for phrase senses or for sorts. However, to give further examples we need to explain some particular choices that were made in Candide. These choices should not be taken as substantive claims of a semantic theory, but just as convenient placeholders that fit the needs of the application. In particular, we do not deal with the problems created by intensionality and opaque contexts.

The representation language used in Candide is basically that of the predicate calculus supplemented by generalized quantifiers and by lambda abstraction to form relations and properties out of formulas. Furthermore, we have a set of *basic sort* constants that represent the basic sorts of the Candide domain. For any sort s and formula $P(x)$ with x as a free variable we have the *restricted sort* $s|_x P(x)$ of the entities x of sort s that satisfy $P(x)$. Sorts are used to qualify assumption parameters and discourse entities and are also used as restrictions for quantified variables in logical forms resulting from assumption discharge.

Senses will be represented by first-order terms, first-order formulas, or, in the case of nominal senses, sorts. We also identify basic sorts with predicate constants when necessary. Strictly speaking, our representation language is *typed*, with basic types $\mathtt{e}$ (entities) and $\mathtt{t}$ (truth values). First-order terms and variables bound by quantifiers are of type $\mathtt{e}$, while first-order formulas are of type $\mathtt{t}$. In addition, we will use derived functional types such as $\mathtt{e} \rightarrow \mathtt{t}$ (properties understood as functions from entities to truth values, which will be taken to include sorts where appropriate). In most cases the types of expressions can be inferred from the context. However, in certain rules it will be convenient to subscript rule schematic variables with types, to indicate that the rule will only apply if the variables are filled by objects of their subscript types.

As an illustration, the following table gives the conditional interpretations before assumption discharge for various phrases:

phrase	assumptions	sense
jet		jet
faulty jet		$jet\|_x faulty(x)$
the faulty jet	$\mathsf{bind}(j, \mathsf{def}, jet\|_x faulty(x))$	j
the faulty jet failed	$\mathsf{bind}(j, \mathsf{def}, jet\|_x faulty(x))$	$failed(j)$

2.5 The Discourse Model

A language interpretation system needs some means of representing those aspects of an utterance's context that are relevant to its meaning. Contextual information derives both from the particular communication process to which the utterance belongs, which we encode in Candide in a *discourse model*, and from real-world constraints on entities, types, and relations described by the utterance, which we represent in a *knowledge base*. In what follows we will discuss the theoretical basis, organization, and construction of the discourse model in some detail. We have less to say about the knowledge base, which in Candide is static and fairly simplistic; we will describe it as needed for the explanation of specific interpretation rules in Section 3.

Our approach to interpretation could in principle accommodate a variety of discourse models. However, any discourse model, and in particular the one used in Candide, must reflect certain general principles. First, a discourse model must have some organizing structure. We chose to structure our model to reflect a particular theory of discourse context, that developed by Grosz and Sidner [17]. Second, the specific contents of the model must match what is required by the rules that access those contents, in our case, the discharge rules. Third, the mapping from states in the communication process to the discourse model must reflect the modes of communication with the language-using system, in our case, Candide.

2.5.1 Structure and Content

The theory of discourse structure developed by Grosz and Sidner [17] is one of the most comprehensive models of discourse in the computational-linguistics literature.[5] The key idea in the theory is that discourses are structured by three interacting components. The first, the *linguistic structure*, is found in the actual utterances of a discourse, which can be naturally factored into hierarchically organized segments. The linguistic structure in general is isomorphic to the second structure, the *intentional structure*, which, as its name suggests, represents the intentions of the discourse participants. The third component of discourse structure is the *attentional state*, which "is an abstraction of the participants' focus of attention as their discourse unfolds ...It is inherently dynamic, recording the objects, properties, and relations that are salient at each point in the discourse" [17, p. 179]. The attentional state is the discourse structure that most directly affects reference resolution questions; it is also the structure that corresponds most directly to the discourse model in Candide. As discussed below, the constraints of our language-using system allow us, in the short-term, to avoid reasoning about agents' intentions, and thus to avoid building a model of the intentional structure.

Candide's discourse model carries information about the objects referred to in a discourse, thereby enabling the interpretation of anaphoric and referring phrases, as well as various kinds of modification. A more sophisticated model would also contain information about the events referred to, typically with verb phrases and entire clauses, to support the interpretation of tense, verb phrase ellipsis, and verb phrase anaphora.

The attentional structure reflects the accessibility of information. In line with this, Candide's discourse model has three components. The most readily accessible information concerns entities referred to in the "current" utterance, the one that is undergoing interpretation at any point in time. Detailed information about those entities is encoded in the first component of our discourse model, which we call the *immediate context*, and which we

[5] But it is not the only one; examples of alternative models include [22, 26].

use primarily for resolving intrasentential anaphora and for making modification decisions that depend on sortal information. The second component of our discourse model, the *local context*, contains detailed information about slightly less-accessible entities, generally those referred to in the immediately preceding utterance. We use the local context primarily for pronoun resolution, following the theory of centering introduced by Grosz *et al.* [16]. The third component of our discourse model is the *global context*, which contains somewhat less detailed information about entities referred to throughout longer stretches of the discourse. The global context is employed primarily for the resolution of definite anaphora, and is structured as a stack to make use of the theory of focusing [14, 40, 5].

Because the immediate and local contexts both contain information about entities that are highly salient, entries in both these components of the discourse model are detailed. The entry for an entity in the immediate or the local context will include (1) a unique parameter that specifies it; (2) the surface syntactic position of the expression that was used to refer to it; and (3) the sort of the entity (for example, *jet*). Entries in the global context include only an entity's identifying parameter and its sort: information about surface syntactic position is presumed to be lost as the discourse proceeds.

2.5.2 Discourses in Candide

As we noted earlier, discourses in Candide consist of procedure descriptions. The Candide user "draws" a procedural network, using a graphical editor, while using English to specify the invocation conditions and arc annotations for the network. The English sentences constitute the utterances of the discourse. After each utterance is interpreted, Candide must update the discourse model. Within the body of the procedural network, Candide attaches the current discourse model to each node.

Consider Figure 4, which depicts the same procedural network as that in Figure 1, but with the discourse model that Candide constructs attached to the nodes. We can specify in general terms the discourse-model update relation as follows: let IC stand for immediate context, LC stand for local context, and GC stand for global context. There are two cases

to consider. In the case of nonbranching arcs (achievement or assertional arcs), let *in* be the start node of the arc, and let *out* be the end node of the arc. Then $LC(out) = IC(in)$ and $GC(out) = push(LC'(in), GC(in))$, where *push* describes the operation of pushing an entry onto a stack, and LC' is a variant of LC, in which certain information, for example syntactic position, has been deleted. $IC(out)$ is determined only from the content of the utterance labeling the arc leaving *out*. In other words, during the interpretation of an arc annotation, the context that applies is that which is associated with the node above the arc.

The second case of discourse model update involves branching arcs. Let *in* be the node starting both arcs, let out_y be the final node of the positive answer arc, and let out_n be the final node of the negative answer arc. Then

$$\begin{aligned} LC(out_y) &= IC(in) \\ LC(out_n) &= IC'(in) \\ GC(out_y) &= push(LC'(in), GC(in)) \\ GC(out_n) &= push(LC'(in), GC(in)) \end{aligned}$$

As with nonbranching arcs, $IC(out_y)$ and $IC(out_n)$ are determined from the utterances that label the arcs outgoing from them. The operator LC' is also as before. The operator IC', however, corresponds to a variant of IC in which queried indefinites have been removed, thus making them inaccessible for subsequent pronominal or definite reference. This is used to disallow fragments like the following, in which the response "No." corresponds to the negative branch of a query arc:

(3) Is there a faulty jet?

No.

Close *it*.

The procedural network directly induces a discourse structure. In particular, as one moves down one branch of a branching arc, one is engaging in a subdialogue. The entities

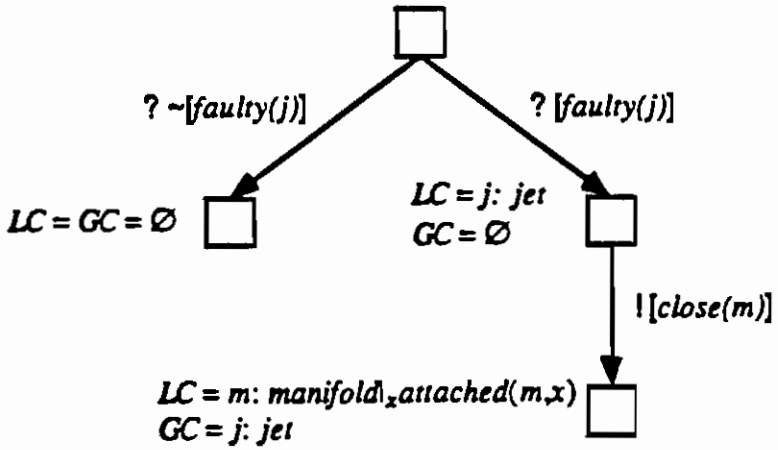

Figure 4: Attachment of the Discourse Model to a Procedural Network

introduced in the subdialogue will be inherited only by nodes in the subdialogue, and thus will be inaccessible to utterances made in the sister subdialogue (the one corresponding to the alternative branch). Discourse structure is also induced by moves to subordinate networks, which are allowed, and indeed heavily used, in PRS.[6]

The correspondence between procedure (or task) structure and discourse structure recalls the early work of Grosz [14], in which just such a correspondence was pointed out. As Grosz and Sidner point out [17], later researchers have sometimes misread this earlier work, taking it to claim that task structure is necessarily the determiner of discourse structure, and that, moreover, the task structure must be known before the discourse structure can be inferred. In the Candide system, the task structure need not, and indeed, typically will not, be known to the system *a priori*. The user makes the task structure explicit by drawing the procedural network. While other forms of language use may not have methods for

[6]In fact, procedural networks are not restricted to tree structures, but can be arbitrary directed acyclic graphs. Nodes with multiple parents present a problem for our current system, since they represent a situation in which the procedural structure of the discourse does not follow directly the temporal order of presentation. Currently, we somewhat arbitrarily simplify our treatment of the discourse model at such nodes, by applying the update rules to the union of the parent discourse contexts.

indicating discourse structure that are as direct as this, there are nonetheless devices that serve just this purpose, for example, cue words and intonation. Those devices are not yet well-enough understood to make the recognition of discourse structure in general a readily solvable problem. One of the major advantages of the Candide application is that we have not had to rely upon these devices, since the network itself provides the needed structure.

3 Interpretation Rules

We now turn to the specific interpretation rules in Candide. We begin, in Section 3.1, by discussing the interpretation of referring noun phrases. In Section 3.2, we show how the conditional interpretations of noun phrases are integrated into the clauses that contain them. In Section 3.3, we deal with modifying expressions, such as prepositional phrases and adjectival and nominal modifiers. Finally, in Section 3.4, we address the interpretation of quantified noun phrases.

3.1 Interpreting Referring Noun Phrases

We first focus on the structural rules used to interpret referring singular noun phrases (NPs): definites, indefinites, pronouns, and proper nouns. Nonquantified plural noun phrases are not treated in our current system. We shall call the rules for interpreting referring noun phrases *the reference class* of interpretation rules.

A referring singular noun phrase is interpreted as a parameter, with a bind assumption that restricts the parameter to be bound in the appropriate way to an entity of the appropriate sort. Thus, as we saw earlier, the phrase "the jet" can be interpreted as $\mathsf{bind}(b, \mathsf{def}, \mathit{jet}) : b$. The bind assumption constrains b to be bound according to the constraints of definite reference to an entity of the sort *jet*. The def parameter indicates that the phase is syntactically definite, that is, its determiner is "the". Discharge rules may later determine that the expression is in fact functioning as an indefinite expression. Henceforth, when we speak of a definite (indefinite) noun phrase, we shall mean one that includes a definite (indefinite) determiner.

Recall the structural rule for interpreting a definite noun phrase:

[def-np] $\mathsf{def\text{-}np}(\mathsf{nom} = N) \rightsquigarrow A, \mathsf{bind}(x, \mathsf{def}, s) : x$ **if** $N \rightsquigarrow A : s$

Figure 5 shows both an analysis tree and a derivation tree for the phrase "the jet". In

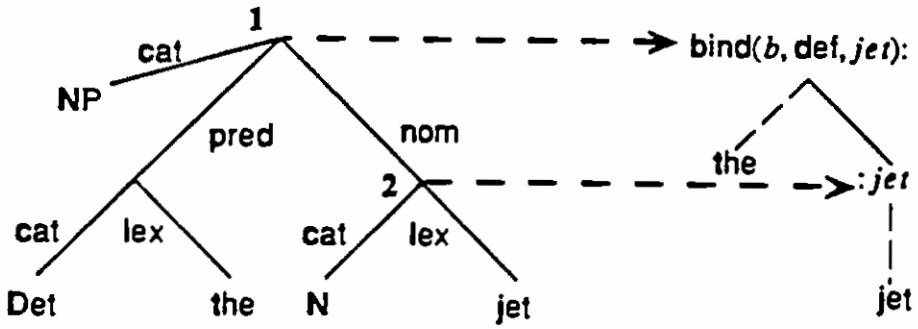

Figure 5: Interpretation of a Definite Noun Phrase

the analysis tree, the feature cat encodes syntactic category information used by pattern predicates.

The derivation is grounded by the lexical access rule lex. This rule selects an interpretation for a word w of syntactic category C from the set $I(w, C)$ of interpretations given in the system's semantic lexicon. The rule is simply

$$[\text{lex}] \quad \text{lex-item}(\text{lex} = w, \text{cat} = C) \rightsquigarrow i \ \text{ if } \ i \in I(w, C)$$

For example, rule lex applies to the lexical subtree rooted at node 2 to associate the interpretation : *jet* with the word "jet". [7] In the conditional interpretation of a common noun, the sense is always a sort term.

Because the tree rooted at node 1 is a definite NP (that is, an NP with a definite determiner), the structural rule def-np applies to it, resulting in the conditional interpretation:

$$\text{bind}(b, \text{def}, jet) : b \tag{4}$$

[7] Node 2 is also lexical, but definite determiners contribute only to the interpretation of their mother NP, by rule def-np, rather than being given a separate interpretation.

The other structural rules of the reference class are similar. Their interpretations differ only in the constraints and sort restrictions of the bind assumption that the rules introduce. The constraint in each depends directly on the form of the noun phrase, while the sort restriction is derived from it. For instance, a proper noun P is identified with the individual whose sort is any entity (e), restricted to having the name P.

[indef-np] $\mathsf{indef\text{-}np}(\mathsf{nom} = N) \rightsquigarrow A, \mathsf{bind}(x, \mathsf{indef}, s) : x$ **if** $N \rightsquigarrow A : s$

[pronoun] $\mathsf{pronoun}(\mathsf{pn} = P) \rightsquigarrow A.\, \mathsf{bind}(x, \mathsf{pronoun}, s) : x$ **if** $P \rightsquigarrow A : s$

[propn] $\mathsf{propn}(\mathsf{name} = P) \rightsquigarrow A.\, \mathsf{bind}(x, \mathsf{pn}, \mathsf{e}|_y name(y, n)) : x$ **if** $P \rightsquigarrow A : n$

3.1.1 Discharge Rules for Referring Noun Phrases

Each of the structural rules presented above introduces a bind assumption, whose kind depends upon the syntactic construction being analyzed. Our procedures for discharging these bind assumptions do not represent a new theory of reference resolution. Instead, our goal has been to show how our interpretation framework can incorporate the contributions of several pragmatic theories not specifically geared toward use in an integrated interpretation system.

In Candide, each major pragmatic phenomenon, such as definite reference, is handled by a separate module consisting of a small set of discharge rules and procedures for their application. It is thus feasible to investigate alternative approaches to a particular pragmatic phenomenon without modifying the modules responsible for other pragmatic processes. Of course, discharge rules interact with one another, since interpretation and discourse model consequences of the application of one rule affect the applicability and results of other rules. In particular, different rule-application orders may lead to different outcomes.

All the rules that discharge bind assumptions introduced by referential phrases have the common effect of binding a parameter x occurring in the sense of a conditional interpretation. The binding may occur in two ways. The parameter may be bound to some discourse entity e; in this case, an entry for e will be made in the discourse model, specifically, in the

immediate context. Alternatively, the parameter may be bound to the parameter of some other, as yet undischarged, bind assumption that appears in the conditional interpretation of the current phrase. This corresponds to determining that the referring expression (the one whose sense is the parameter x) is coreferential with another expression in the phrase, even through the actual mutual referent has yet to be determined. In either type of binding, all instances of the parameter x in the conditional interpretation are replaced with instances of the entity or parameter to which x has been bound. Thus, the rules for discharging a bind assumption introduced by a reference class structural rule all have the following schematic form

[ref-discharge] $P \rightsquigarrow A[x/r] : s[x/r]$ if $P \rightsquigarrow A, \mathsf{bind}(x, C, t) : s$

where C is def, indef, pronoun, or pn, and r is either an allowable referent or an allowable coreferring parameter. In addition, there will be conditions that relate the various rule parameters to the discourse model, and, in particular, supply possible values r to which x can be bound.

We illustrate these ideas with the discharge rules for definite noun phrases. There are two primary sources of candidate referents for definite noun phrases in Candide: the knowledge base and the discourse model. An entity e that is the only instance of a particular sort s in the knowledge base can be always referred to using a definite noun phrase, even discourse initially: for example, the noun phrase "the current president of Mexico" would unambiguously refer to the single entity who is the current president of Mexico. We presume that both the system and the user have mutual knowledge of the entities in the knowledge base.

Alternatively, candidate referents may have been introduced in the discourse itself, and hence may be stored in the discourse model. We adopt a set of control rules that are similar to those proposed in earlier work on definite reference resolution [14, 40, 44, 5]: we look for both intrasentential and intersentential antecedents, restricting the latter to entities introduced in the current discourse segment or in a parent segment. We prefer entities that

are in the current discourse segment and that have the same sort or a supersort of the target definite NP. Our next preference is for same-sort or supersort entities in a parent segment. If a candidate antecedent still cannot be found, we seek one that can be construed as being functionally dependent upon the entity referred to by the target NP; again, we look first in the current segment and then in parent segments. Functional dependency is illustrated in the following discourse fragment, similar to that in Figure 1:

(5) The jet failed.

Close the manifold.

Because the knowledge base contains the information that each jet is attached to one and only one manifold, we can determine that the definite NP used in the second sentence refers to the unique manifold that is attached to the jet mentioned in the first sentence. If "the jet" has been interpreted as j, "the manifold" will be interpreted as the unique entity m with the sort $manifold|_x attached\text{-}to(x, j)$.

As we noted before, discourse entities introduced during the interpretation of a discourse, such as m in the current example, may not be associated to a particular domain entity until the interpretation result is executed by PRS: the execution of a procedural network is not part of Candide's operation. In the case of the discourse fragment (5), the PRS assertional arc labeled with (the interpretation of) the first sentence will be traversed when there is a particular jet, j, under discussion, and PRS has just determined that it failed. As the achievement arc corresponding to the second sentence is traversed, m will be bound to whatever manifold is attached to the jet that, in fact, failed.

When interpreting a definite noun phrase as functionally dependent on some other object, the determination of the intended function may require arbitrarily complex domain reasoning [22]. In practice, though, it is often the case that the intended function follows immediately from the sorts of the objects being related and a small set of essential attributes, such as the part/whole relations in which they participate. We have thus chosen in Candide to restrict the class of functions considered to those that can be obtained from

knowledge-base information about sorts and their roles, which are binary relations between elements of a sort and their attributes. More specifically, we consider as candidate functions those roles that are functional, the inverse relations of roles if they are functional, and compositions of these. The restriction is motivated by the uniqueness requirement of most definite reference.

For example, a knowledge base containing information about a sort *work-station-display* may also describe two of its roles: *cursor*, which is functional, because a display typically has a unique cursor, and *window*, which is not, because displays typically show multiple windows. Both of the inverse roles are functional: every cursor and window is part of a unique work-station display. It is easy to see in this example how the functionality requirement operates. Having made salient in the discourse some specific display, one can then go on the speak of "the cursor", but cannot felicitously speak of "the window". On the other hand, if one has made salient a particular window, one *can* then refer to "the display". If a role and its inverse are functions, one may in fact chain in either direction: one can speak first of a particular display, and then mention "the cursor", or one can speak first of a particular cursor, and then mention "the display".

The most appropriate control strategy for searching for an appropriate functional dependency remains an open research question. We adopt the commonly used approach of performing a breadth-first graph traversal, trying to link the sort of the definite noun phrase to the sort of some target by chaining through functional roles and functional inverses of roles. As before, we begin the search with entities in the current discourse segment, and then proceed through parent segments.

We have provided details of our treatment of definite reference as illustration of the sort of pragmatic theory we incorporate into Candide. We shall not provide a similar level of detail about our treatment of indefinites, pronouns, or proper nouns, since, again, we have generally adopted the approaches of other researchers.

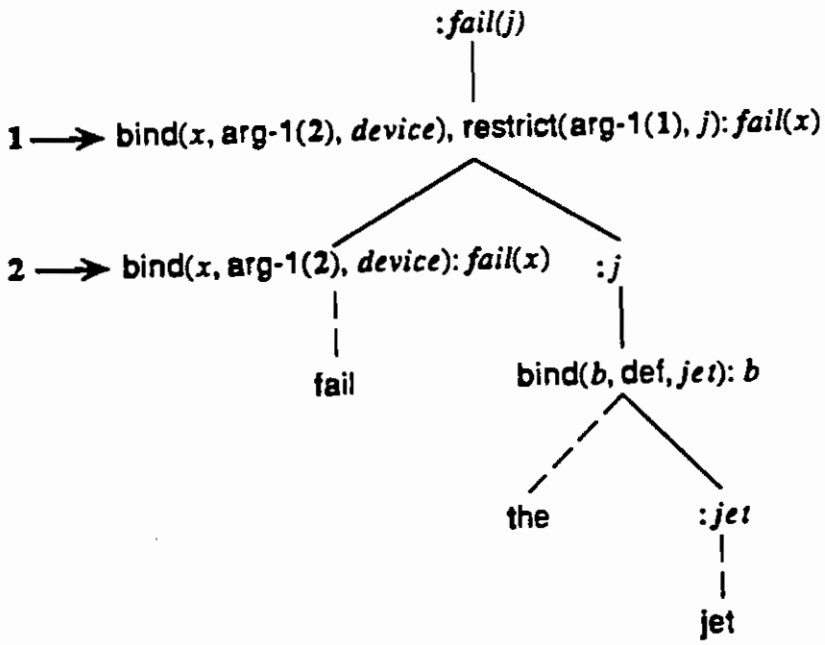

Figure 6: Interpreting a Main Clause

3.2 Interpreting Clauses

We now consider how the conditional interpretations of NPs are integrated into interpretations of the clauses that contain them. We distinguish between copular clauses and noncopular ones, and begin our discussion with the former.

3.2.1 Noncopular Clauses

We introduce the interpretation of clauses with an example: Figure 6 shows a derivation tree for an interpretation of "the jet failed". Notice that it includes the derivation tree for "the jet" that we previously discussed.

The interpretation of the main verb "fail" provided by rule lex is

bind(x, arg-1(2), *device*) : *fail*(x).[8] Informally, this interpretation expresses the fact that the sense of "fail" is *fail*(x), where x is to be bound according to the constraints specified by the bind assumption. Let us consider that assumption in some more detail.

Entries for verbs in the semantic lexicon refer, through assumptions, to the grammatical roles that will provide arguments to the predicate representing the sense of the verb. Thus the interpretation for "fail" refers to arg-1, a particular grammatical role that happens to correspond to what is often called "surface subject". Since we are not defending any particular theory of grammar in this paper, we have chosen to skirt a theoretical and terminological minefield by naming the grammatical roles relevant to our purposes arg-i for $i = 1, \ldots, n$.

Note also the index 2 included in the assumption constraint. This is an index into the analysis tree for the phrase: it specifies that the assumption is referring to the arg-1 role of the phrase whose main verb is rooted at node 2 of the analysis tree. In other words, the parameter x in the sense *fail*(x) is to be bound to whatever is found to be the interpretation of the arg-1 role of the verb phrase rooted at node 2, that is, the interpretation of the phrase "the jet failed". In our derivation trees, we represent analysis-node indices by bold-face numbers, and show the correspondence between analysis nodes and structural derivation nodes by arrows from the appropriate numbers to the derivation nodes. In Figure 6, two indices are used: 2, which corresponds to the analysis node for the verb "fail", and 1, which corresponds to the analysis node for the entire clause "the jet failed".

Conditional interpretations for verbs also encode selectional restrictions. In the current example, the bind assumption's sort parameter is *device*, representing the the constraint that the first argument of the verb "fail" must be filled by an entity of sort *device*—things that fail are devices.

In general, then, rule lex will return for an n-argument verb a conditional interpretation of the form

[8] We are ignoring tense.

$$\mathsf{bind}(x_1, \mathsf{arg\text{-}1}(i), s_1), \ldots, \mathsf{bind}(x_n, \mathsf{arg\text{-}}n(i), s_n) : r(x_1, \ldots, x_n)$$

where i is the analysis node for the phrase being interpreted by lex.

We can now consider the interpretation of entire clauses. Noncopular clauses, both main and subordinate, are interpreted by the following schematic rule:

$$\begin{aligned}[\text{clause}] \quad & \mathsf{clause}(\mathsf{pred} = V, \mathsf{arg\text{-}1} = P_1, \ldots, \mathsf{arg\text{-}}n = P_n) \sim \\ & A_0, A_1, \ldots, A_n, \mathsf{restrict}(\mathsf{arg\text{-}1}(\uparrow), s_1), \ldots, \mathsf{restrict}(\mathsf{arg\text{-}}n(\uparrow), s_n) : r \ \textbf{ if} \\ & V \sim A_0 : r \ \textbf{and} \ P_1 \sim A_1 : s_1 \ \textbf{and} \ \cdots \ \textbf{and} \ P_n \sim A_n : s_n\end{aligned}$$

The symbolic index $\uparrow$ used in the restrict assumptions denotes the analysis node for the phrase being analyzed by the current rule. Note that the restrict assumptions introduced by clause, like the bind assumptions introduced in verb interpretations, refer to grammatical roles. By pairing the grammatical roles specified in a clause's interpretation with those specified in the interpretation of the clause's verb, Candide can enforce constraints relating grammatical role to arguments of the sense predicate. In particular, it can enforce the obvious requirement that the fillers of the argument positions in the main verb of a clause are the subject and complements of the clause.

Again, return to the example in Figure 6. After application of clause, the sense of the clause is $\mathit{fail}(x)$, the sense inherited from the verb. The assumption set consists of the assumptions inherited from the verb (the single bind assumption $\mathsf{bind}(x, \mathsf{arg\text{-}1}(2), \mathit{device})$), assumptions inherited from the verb's arguments (none), and the new restrict assumption $\mathsf{restrict}(\mathsf{arg\text{-}1}(n), j)$. This specifies that the first argument of the clause's sense, namely x, must be bound to the entity j, itself the sense of the filler of the arg-1 grammatical role of the clause. The restrict assumptions introduced by clause thus represent the connection between a phrase as a filler of a grammatical function in the syntax of the clause and the phrase's sense as the filler of an argument position in the sense of the clause's predicate. This connection between syntactic and semantic structure is analogous to that provided by projection functions in lexical-functional grammar [19].

The derivation in Figure 6 continues with the discharge of the two assumptions in the

initial conditional interpretation of the entire clause: the argument bind that is inherited from the interpretation of the predicate, and the restrict that is introduced by the clause rule. These two assumptions can be discharged successfully in parallel: binding x to j is legitimate because j has sort *jet*, and *jet* is a subsort of *device*. The discharge rule used for this is as follows:

$$\text{[arg-apply]} \quad P \rightsquigarrow A : (\lambda x.p)(y) \text{ if } P \rightsquigarrow A, \mathsf{bind}(x, \text{arg-}i(n'), s'), \mathsf{restrict}(\text{arg-}i(n), y_{\mathsf{e}}) : p$$

We require that node n' correspond to an appropriate grammatical-role filler for node n and that sort s' be a subsort of the sort s of y. Notice that the parameter y in the restrict assumption must be of the type entity, as indicated by the subscript e. A different discharge rule will apply when the parameter is a sort, as we will see in the discussion of relative clauses below.

The foregoing analysis applies only to situations in which all arguments in a clause are specified as such in the syntactic analysis of the clause. However, slight modifications will suffice to handle cases in which the least-commitment parser has attached verb complements too low as modifiers of other verb complements.

Every discharge of an argument-filler restrict assumption introduced by the clause rule is paired with the discharge of an argument bind assumption introduced by lex. This process effectively pairs the argument position of a predicate to its filler. More generally, we will see later that phrase references in restrict assumptions serve to identify the syntactic domain of candidates for modification by that assumption. As a result, if, in a derivation, a restrict assumption is not discharged with respect to a sense coming from the appropriate domain, the assumption will be "orphaned", that is, there will be no way to discharge the assumption in the rest of the derivation, which will thus be impossible to complete. While this is a perfectly good way of specifying declaratively the possible derivations, it is not reasonable computationally, because it leads too often to blocked derivations. For this reason, in the Candide system, we do not let a derivation proceed if it includes any orphaned assumptions.

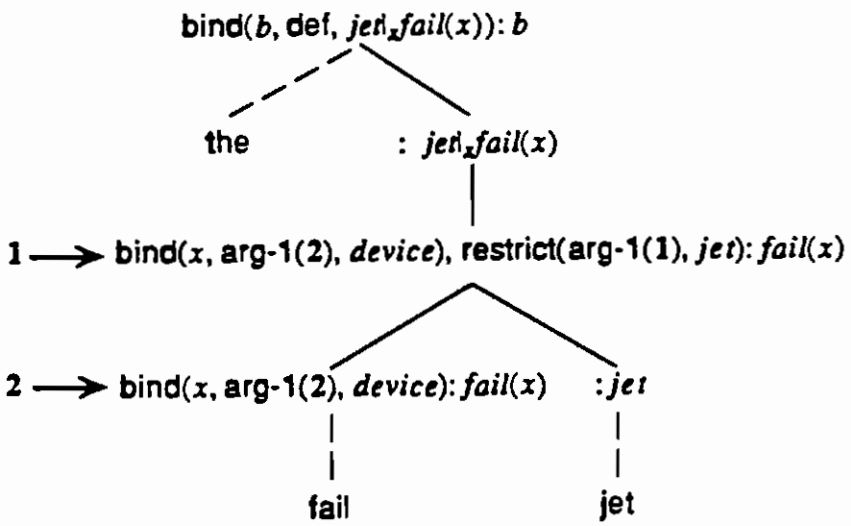

Figure 7: Interpreting a Relative Clause

3.2.2 Relative Clauses

We now show how the clause rule is used to interpret relative clauses. Figure 7 is a derivation of an interpretation for "the jet that failed". Our syntactic treatment of relative clauses is somewhat unorthodox: we analyze a relative clause just like a main clause, except that the position in the analysis corresponding to the relative-clause gap is filled with the nominal modified by the relative clause. In other words, the relative clause is "wrapped around" the nominal it modifies. Because the analyses for relative clauses and main clauses are so similar, the clause rule applies to both. In the current example, the resulting interpretation is:

$$\mathsf{bind}(x, \mathsf{arg\text{-}1(2)}, \mathit{device}), \mathsf{restrict}(\mathsf{arg\text{-}1(1)}, \mathit{jet}) : \mathit{fail}(x) \tag{6}$$

where 1 is the clause node and 2, the predicate node.

The difference between relative and main clauses is reflected in the discharge of argument bind assumptions against argument-filler restrict assumptions. Compare the restrict

assumption from Figure 7 (listed in (6)) with the corresponding restrict assumption in Figure 6. In (6) the filler for the first argument of the predicate *fail* is a sort term *jet* (its type is $\mathtt{e} - \mathtt{t}$), whereas in the earlier example it was an entity j (its type was $\mathtt{e}$). The discharge rule that combines a verb argument with its filler has two versions: one in which the filler sense is an entity, and the other in which it is a sort. The former case was covered by the rule arg-apply given earlier. In the latter case, the rule produces an interpretation whose sense is the filler sort restricted by the sense of the clause. The resulting interpretation for the current example is thus $jet|_x fail(x)$. The general form of the corresponding discharge rule is

[arg-restrict] $P \rightsquigarrow A : s|_y(\lambda x.p)(y)$ if

$$P \rightsquigarrow A, \mathsf{bind}(x, \mathsf{arg}\text{-}i(n'), s'), \mathsf{restrict}(\mathsf{arg}\text{-}i(n), s_{\mathtt{e}-\mathtt{t}}) : p$$

where n and n' are as for rule arg-apply.

Although our example above is for an intransitive verb, the same mechanism will apply without change to relative clauses with multiargument verbs in which the gap filler occupies any one of the argument positions. Only the argument that has been relativized will have an interpretation whose sense is a sort term; hence, that argument will be the one to which the arg-restrict rule applies; arg-apply will apply to the restrict assumptions associated with all the other arguments. The entire relative clause will thus receive an interpretation whose sense is a restricted sort term: its head sort is the sort of the modified noun, regardless of what argument position has been relativized.

The parallelism between restricting a sort and filling an argument, exemplified by rules arg-restrict and arg-apply, is central to our interpretation system and will reappear in several of our other discharge rules. In general, for any rule that fills an argument position of a predicate, there will be a parallel rule that abstracts over that argument position to provide a restriction to a sort. Which rule is used depends only on whether the argument filler is an individual or a sort. Thus a restrictive adjective, a prepositional phrase or a clause may be used to qualify an individual (predicate adjective, predicate prepositional phrase,

main clause) or to restrict a sort (prenominal adjective, postnominal prepositional phrase, relative clause).

3.2.3 Copular Clauses

In our application, copular clauses are defined to be those with main verb "to be" and a single predicate nominal, adjective phrase, or adverbial complement. The interpretation of copular clauses relies on a clausal structural rule, along with specialized structural rules for each of the types of copular complements handled by our system, and the argument discharge rules discussed previously (arg-apply and arg-restrict).

The copular clause rule is as follows:

$$\text{[cop-clause]} \quad \mathsf{clause}(\mathsf{pred} = \mathsf{be}, \mathsf{arg\text{-}1} = S, \mathsf{arg\text{-}2} = P) \rightsquigarrow A, B, \mathsf{restrict}(\mathsf{arg\text{-}1}(\uparrow), s) : p \text{ if } S \rightsquigarrow A : s \text{ and } P \rightsquigarrow B : p$$

The interpretations of predicate adjectives, predicate PPs and predicate nominals will have the same form as those of intransitive main verbs, that is, a proposition-type sense, subject to an argument bind assumption. This assumption will be discharged against the restrict assumption introduced in rule cop-clause for the subject of the copular clause. The following structural rules create the appropriate predicate interpretations:

$$\text{[pred-adj]} \quad \mathsf{pred\text{-}adj}(\mathsf{pred} = P) \rightsquigarrow A, \mathsf{bind}(x, \mathsf{arg\text{-}1}(\uparrow), \mathsf{e}) : p(x) \text{ if } P \rightsquigarrow A : p$$

$$\text{[pred-pp]} \quad \mathsf{pred\text{-}pp}(\mathsf{pred} = P, \mathsf{pobj} = O) \rightsquigarrow A, \mathsf{bind}(x, \mathsf{arg\text{-}1}(\uparrow), \mathsf{e}) : r(x, e) \text{ if } P \rightsquigarrow : r \text{ and } O \rightsquigarrow A : e$$

$$\text{[pred-nom]} \quad \mathsf{pred\text{-}nom}(\mathsf{arg\text{-}1} = P) \rightsquigarrow A, \mathsf{bind}(x, \mathsf{arg\text{-}1}(\uparrow), \mathsf{e}) : p(x) \text{ if } P \rightsquigarrow A : p$$

The patterns in these rules match only the copular complement position. The predicate nominal pattern matches an indefinite noun phrase with feature arg-1 representing the nominal.

3.3 Interpreting Modifying Expressions

Structural rules in the modification class are used for the interpretation of various constructions falling under the traditional label of modification: attributive restrictive adjectives, nominal modifiers and prepositional phrases. As was the case for clausal rules, each of the modification rules introduces a restrict assumption describing the interpretation constraints supplied by the modifying phrase. In contrast with the clausal rules, however, the phrase to be modified may not be fully specified syntactically.

We begin with the rule for attributive modification of a nominal by a restrictive adjective. The sense of a restrictive adjective is a property, that is, it has type $\mathbf{e} - \mathbf{t}$. The interpretation rule should thus have roughly the following form:

$$\mathsf{nom}(\mathsf{adj} = A, \mathsf{nom} = N) \rightsquigarrow P.Q.\mathsf{restrict}(\mathsf{adj}(\uparrow), r) : s \text{ if } A \rightsquigarrow P : r \text{ and } N \rightsquigarrow Q : s$$

If A is an adjective with interpretation $P : r$, which syntactically modifies a noun N with interpretation $Q : s$, then the interpretation of the nominal consisting of A and N should have the sense s of its noun; its assumptions should include a restrict assumption encoding the contribution of A's sense r.

With this structural rule, the interpretation of a noun phrase containing an adjective is straightforward, as shown by the derivation tree for "natural language" in Figure 8. The effect of discharging an adjectival restrict assumption is simply to introduce the adjective's sense (a property) as a restriction to the sort corresponding to the sense of the nominal being modified. A first approximation to the appropriate discharge rule would be

$$P \rightsquigarrow A : s|_x r(x) \text{ if } P \rightsquigarrow A, \mathsf{restrict}(\mathsf{adj}(i), r) : s \quad .$$

where i must be syntactically allowed as an adjectival modifier of the phrase with sense s.

In fact, the rules given above are not quite correct. To see why, let us first consider a similar rule for nominal modification:

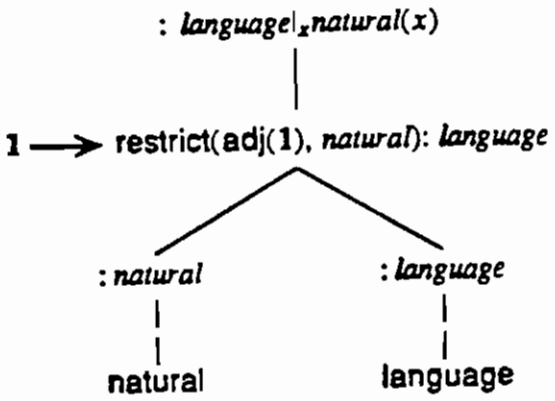

Figure 8: Adjectival Modification

$$\mathsf{nom}(\mathsf{nom\text{-}1} = N, \mathsf{nom\text{-}2} = N') \sim A, B, \mathsf{restrict}(\mathsf{nom}(\uparrow), r) : s \quad \text{if}$$
$$N \sim A : r \text{ and } N' \sim B : s$$

Figure 9 shows the use of this rule in the interpretation of "language processing". For the moment, we will leave unspecified the precise method by which we determine the appropriate modifying relation in the discharge of the restrict assumption; we will return to this question below. What is important to note here is that the structural rule for interpreting compound nominals specifies that the sense of the compound nominal is the sense of the rightmost noun (nom-2), under the assumption that that sort will be modified by the sense of the noun that precedes it (nom-1).

Now consider what happens when we apply the rules given so far to the phrase "natural language processing". The least-commitment parser gave the phrase the default right-branching analysis, and it is the job of the interpretation rules to compute alternative bracketings. Two alternative derivation trees are shown in Figures 10 and 11. In the first, the restrict assumption introduced by the interpretation of "language" is discharged immediately after introduction; in the second, its discharge is deferred until after the discharge of the restrict assumption introduced during the interpretation of "natural". In either case,

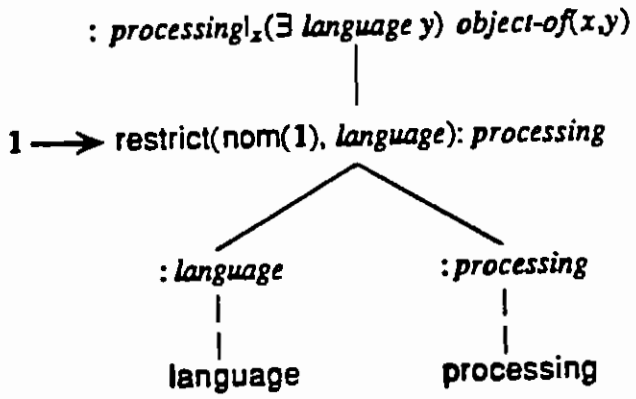

Figure 9: Nominal Modification

however, one gets the less-common reading, in which the processing is both natural and about language. The modification rules as given above are not capable of reconstructing alternative bracketings from the least-commitment parse, so we need some mechanism to derive the alternative interpretation in which "natural" modifies "language" rather than "processing".

Alternative bracketings for compound nominals are generated by *assumption nesting*, a general device used in our system to keep track of dependencies between assumptions introduced by nested constituents. Each assumption α has an additional component, a set of other assumptions *nested* in α. When the set of assumptions nested within some assumption is empty, we have omitted it, and will continue to do so. The assumptions nested in an assumption α arise from the conditional interpretation of some constituent of the phrase for whose interpretation α was introduced.

The default effect of discharging an assumption containing nested assumptions is to "pop" the nested assumptions into the main assumption set for the resulting conditional interpretation. All relevant discharge rules given earlier should be understood in this way. For example, the nested-assumption version of the ref-discharge rule would be

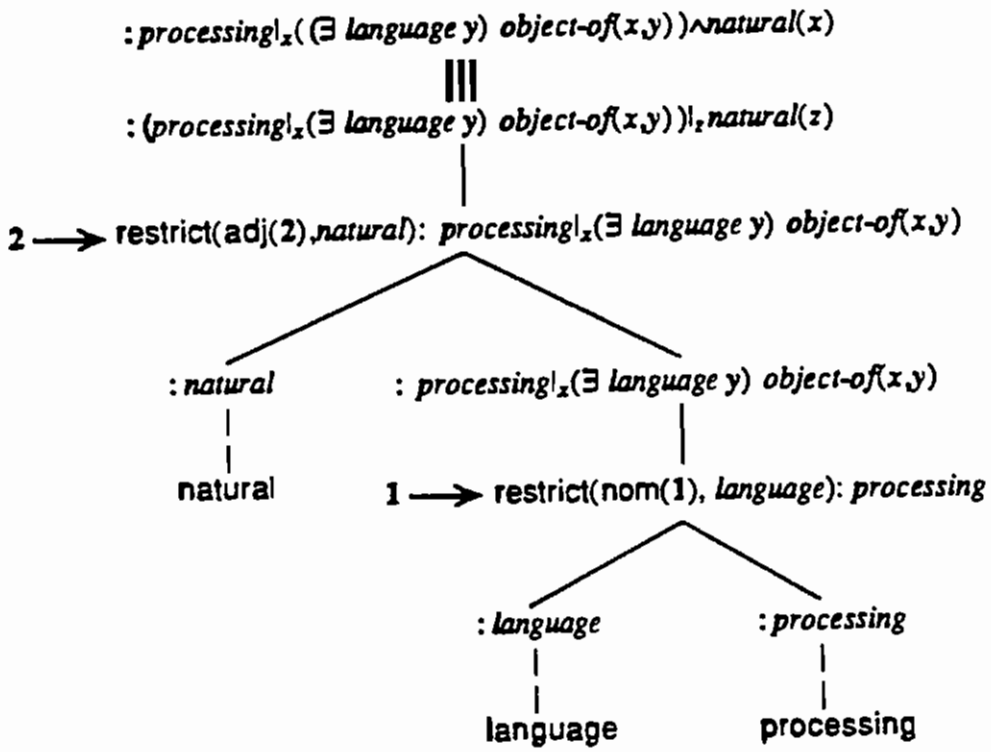

Figure 10: Mixed Modification—Early Discharge

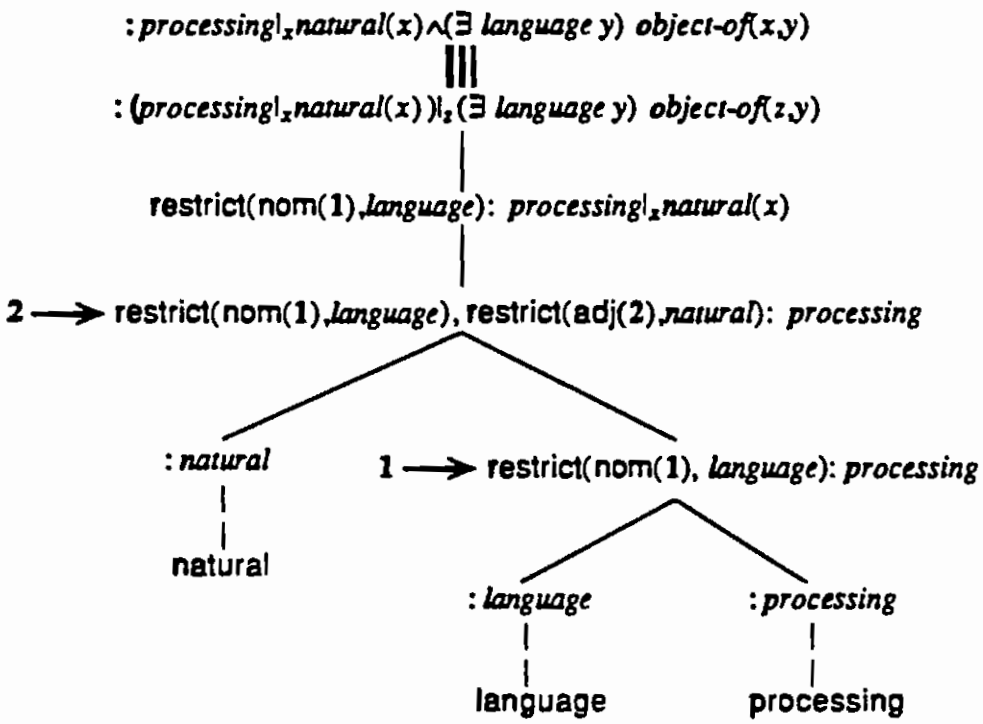

Figure 11: Mixed Modification—Late Discharge

[ref-discharge] $P \rightsquigarrow A[x/r], B[x/r] : s[x/r]$ if $P \rightsquigarrow A, \mathsf{bind}(x, C, t, B) : s$

We can now see how the nesting mechanism is used in interpreting modifying expressions. When a restrict assumption is introduced by the interpretation of either a modifying noun or adjective, all other nominal restrict assumptions that are inherited from the rule antecedent will be nested within the newly introduced assumption. Thus the correct rules for adjectival and nominal modification are as follows:

[adj-mod] $\mathsf{nom}(\mathsf{adj} = A, \mathsf{nom} = N) \rightsquigarrow P, Q - Q', \mathsf{restrict}(\mathsf{adj}(\uparrow), r, Q') : s$ if
$A \rightsquigarrow P : r$ and $N \rightsquigarrow Q : s$

[nom-mod] $\mathsf{nom}(\mathsf{nom\text{-}1} = N, \mathsf{nom\text{-}2} = N') \rightsquigarrow P, Q - Q', \mathsf{restrict}(\mathsf{nom}(\uparrow), r, Q') : s$ if
$N \rightsquigarrow P : r$ and $N' \rightsquigarrow Q : s$

where Q' is the set of nominal modifying assumptions in Q; $Q - Q'$ denotes set difference.

In discharging the assumptions that are introduced by rules adj-mod and nom-mod, the question of whether there are any nested assumptions becomes crucial. If there are not, then the discharge rule shown above still applies. However, in the case in which there are nested assumptions, the restriction specified in the top level of the assumption is taken to be a qualification of the first nested sort—not a qualification of the sense itself. Thus, the result of discharging a restrict assumption introduced by adj-mod or nom-mod with a nested assumption is a conditional interpretation with the same sense as that in the antecedent of the rule, but with a different assumption list. For the adjective case, the improved discharge rule is:

[adj-discharge] $P \rightsquigarrow A, \mathsf{restrict}(\mathsf{nom}(n), s|_x r(x), B) : t$ if
$P \rightsquigarrow A, \mathsf{restrict}(\mathsf{adj}(n), r, \mathsf{restrict}(\mathsf{nom}(n'), s, B)) : t$

where either $n' = n$ or n' is a daughter of n. The discharge rule for the nominal case is almost identical:

[nom-discharge] $P \rightsquigarrow A, \mathsf{restrict}(\mathsf{nom}(n), s|_x r'(x), B) : t$ if
$P \rightsquigarrow A, \mathsf{restrict}(\mathsf{nom}(n), r, \mathsf{restrict}(\mathsf{nom}(n'), s, B)) : t$

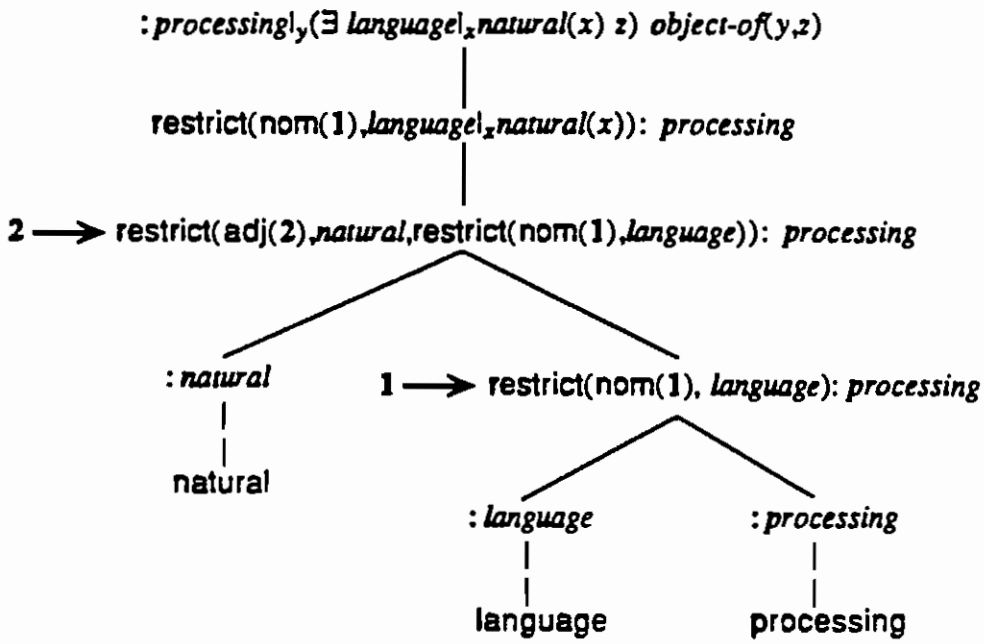

Figure 12: Modification with Nesting

With the above rules, we can now derive the more natural reading of "natural language processing", as illustrated in Figure 12.

It is worth noting that the nesting mechanism allows only those combination alternatives that respect the syntax of complex phrases. For example, when interpreting "computer language processing system", nesting will block the impossible alternatives in which "computer" modifies "processing" and "language" modifies "system".

It is reasonable to ask why we nest only nominal modification assumptions, and not those introduced by adjectives; after all, there are phrases like "dirty blonde hair" that exhibit a structural ambiguity analogous to that of "natural language processing", but in which the medial word is an adjective. The difficulty with that example is that the two alternative readings do not involve just structural ambiguity, but lexical-semantic ambiguity

as well. To interpret the bracketing "dirty [blonde hair]", the word "dirty" must be used as a restrictive adjective, with a sense of type $\mathsf{e} \multimap \mathsf{t}$, while to interpret the alternative bracketing "[dirty blonde] hair" the same word must be an adjective-modifying (adverbial) sense, that is, a sense of type $(\mathsf{e} \multimap \mathsf{t}) \multimap (\mathsf{e} \multimap \mathsf{t})$. A systematic relationship between the two senses is not obvious. A short-term solution, of course, is to include two distinct senses for "dirty" in the lexicon, as we do for other cases of lexical-semantic ambiguity.

When we discharge a nominal modifier, we need to find an implicit relation between the noun being modified and the noun modifying it; for instance, in the case of "language processing", we must determine that the *language* is the *object-of* the *processing*. In the nom-discharge rule, this relation is represented with the relation parameter r'. The are two perspectives one might take on the implicit relation represented by r'. For a certain class of modified nouns, namely, relational nouns and those derived from verbs, the modification relation can be seen as argument-filling, similar to the relation between verbs and their arguments (complements). On the other hand, nominal modification can in general be seen as a predication relation, similar to that between verbs and their adjuncts.

Consider first the case of relational and deverbal nouns. We can handle modification of these using a treatment similar to the one we use for verbs. The lexical interpretations of such nouns would then contain parameters and associated bind assumptions corresponding to the arguments that may be filled. For example, the conditional interpretation for the word "processing" would be:

$$\mathsf{bind}(x, \mathsf{arg\text{-}1}(n), \mathit{object}), \mathsf{bind}(y, \mathsf{arg\text{-}2}(n), \mathit{object}) : \mathit{processing}(x, y) \tag{7}$$

Restrict assumptions introduced by nom-mod would then be discharged against these bind assumptions, similar to the way in which restrict assumptions introduced by the clausal rules are discharged against the bind assumptions introduced by the lexical interpretation of verbs. Advantages of this approach include (1) the fact that the sortal (selectional) restrictions for the role-fillers are clearly indicated in the set of assumptions included in the conditional interpretation, and (2) the fact that the search for the relation between the

nouns is immediately constrained; rather than seeking an arbitrary relation, one looks for the appropriate argument-filling relation.

A primary disadvantage of the approach is that in compound nominals the argument-filling relationship is optional. One can, for example, say "language processing", without specifying what is doing the processing.[9] This optionality would result in undischarged bind assumptions. Another disadvantage is that this solution will not handle the modification of nouns that are neither relational nor deverbal. Consequently, in this paper, we have adopted the alternative approach, in which relations corresponding to relational nouns and deverbals are reified, and argument fillers are connected to such relation entities by binary predicates corresponding to argument roles [10, 31]. We limit ourselves to the predication-based approach only for ease of presentation. In fact, in the implementation of the Candide system, we adopted a hybrid approach, in which we treated relational nouns as they are treated in this paper, while we handled participles and deverbal nouns, by first attempting the alternative, argument-filling approach discussed above. If an argument-filling relation could not be found in such cases, then a predicating role was ascribed to the modifying noun.

With the rules just described, modifying nouns in compound nominals do not introduce new entities into the discourse model and thus cannot be subsequently referred to.[10] This accounts for the existential quantification of the entity of sort *language* in the interpretation of "language processing" in Figures 10, 11, and 12 above.

The treatment of prepositional phrases should now be clear. The structural rule for interpreting PPs is:

$$[\mathsf{pp}] \quad \mathsf{pp}(\mathsf{prep} = P, \mathsf{psubj} = S, \mathsf{pobj} = O) \rightsquigarrow A, \mathsf{restrict}(\mathsf{pp}(p, \uparrow), o, B) : s \text{ if } P \rightsquigarrow : p \text{ and } S \rightsquigarrow A : s \text{ and } O \rightsquigarrow B : o$$

[9] The recovery of such implicit argument fillers has been a major focus of the Pundit project [30].

[10] This effectively forces compound nominals to be anaphoric islands [34], which is in general too strong a restriction [43] but is nonetheless a reasonable design choice given the absence of a detailed theory of the relatively limited cases in which reference into putative anaphoric islands is possible.

Prepositional phrase modification, like adjectival and nominal modification, can result in nested PP restrict assumptions. However, there is a difference in the way in which these nested assumptions are discharged: when a PP-restrict with a nested assumption is discharged, the outermost sort does not restrict the first nested sort, as was the case with adjectival and nominal modification. Instead, the outermost PP restricts the sense of the current conditional interpretation, with the nested assumption then being "popped" to the top level. The following discharge rule achieves this:

[pp-discharge] $P \rightsquigarrow A, B : s|_x p'(x, o)$ if $P \rightsquigarrow A, \mathsf{restrict}(\mathsf{pp}(p, n), o, B) : s$

The relation p' is determined from the preposition and from the sorts of the modifying and modified sense, by using information in the knowledge base about sorts and the relations in which their members may participate. The discharge procedure must ensure that n is syntactically allowed as a modifier of the phrase with sense s.

The asymmetry between adjectival and nominal modification, on the one hand, and prepositional phrase modification, on the other, results from the left-to-right order of English, in which prepositional phrases generally follow the terms they modify, whereas adjectives and nouns (in general) precede them: this structure is reflected in our analyses. With both types of modification, early discharge results in a right-bracketed structure, while late discharge results in a left-bracketed structure. Figures 13 and 14 show two derivations of "man near the car in the park", the first with right bracketing and the second with left bracketing.

3.4 Interpreting Quantified Phrases

The structural rules for quantification are quite similar to the reference class rules given in Section 3.1: the interpretation of a quantified NP is a new parameter that is subject to a bind assumption. There are two rules, **gen-quant** for general NPs and **int-quant** for interrogative NPs. Rule **gen-quant** is as follows:

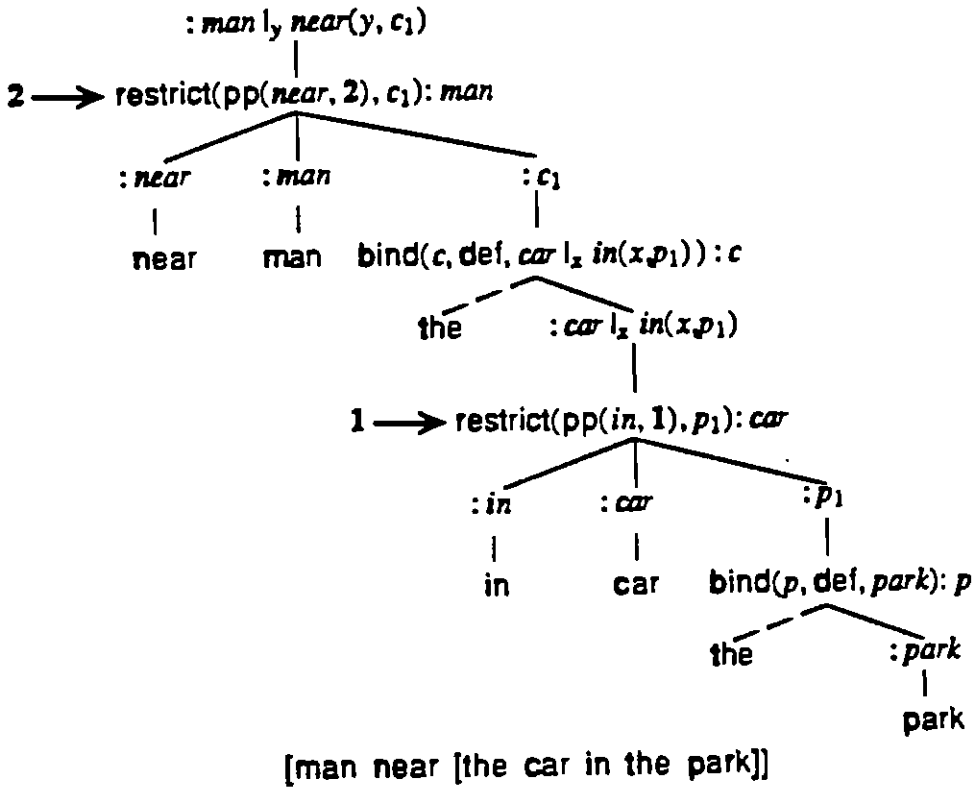

Figure 13: Prepositional Phrases—Right Bracketing

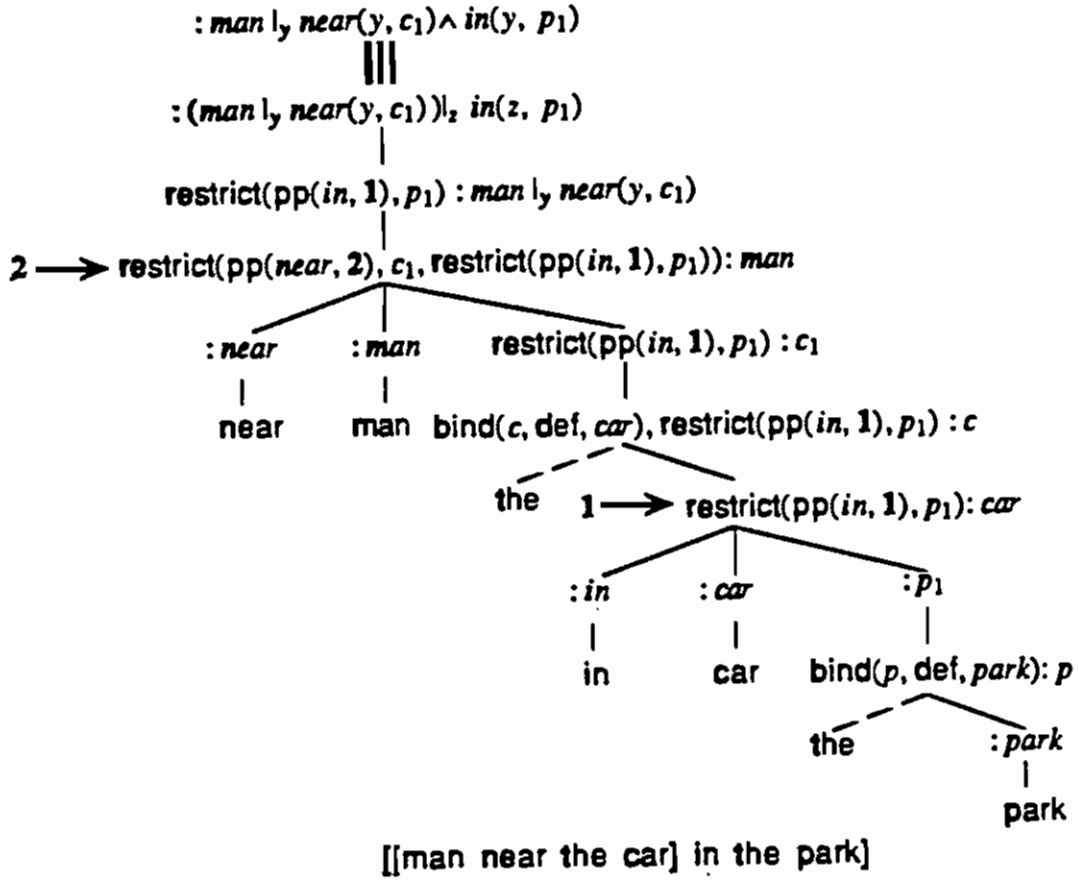

Figure 14: Prepositional Phrases—Left Bracketing

[gen-quant] $\mathsf{gen\text{-}quant}(\mathsf{quant} = D, \mathsf{nom} = N) \sim \mathsf{bind}(x, q, n, A) : x$ if

$D \sim : q$ and $N \sim A : n$

Rule int-quant is almost identical, so we will not discuss it further here.

General NPs include all quantified NPs accepted by Candide except for those with definite or referential indefinite determiners, which are interpreted by the def-np and indef-np rules discussed in Section 3.1.

The bind assumption $\mathsf{bind}(x, q, n, A)$ introduced by rule gen-quant can be seen as encoding a quantifier store element along the lines of Cooper storage [7]. The rule puts a quantifier into store. Appropriate discharge rules, discussed below, unstore quantifiers and apply them to their chosen scopes. As we shall see shortly, different orders of assumption discharge will produce different relative scopes for the quantifiers in the interpretation of a sentence.

Examining rule gen-quant it is clear that q is the sense of the determiner of some NP, while n is the sense of the nominal of the NP. Thus, the phrase "every jet" would receive the conditional interpretation $\mathsf{bind}(x, \mathit{every}, \mathit{jet}) : x$ (there are no nested assumptions here). Disregarding for the moment interactions with reference, the discharge of a general quantifier assumption $\mathsf{bind}(x, q, n, A)$ with respect to a sense S will yield the new sense $q(n, \lambda x.S)$, which we will more perspicuously notate as $(q\ n\ x)S$. Assume a derivation in which a decision is made to discharge the quantifier bind assumption during analysis of the entire phrase "every jet failed". Assuming all other assumptions have already been discharged, the discharge of the quantifier bind assumption will lead to the conditional interpretation $: (\forall\ \mathit{jet}\ x)\mathit{fail}(x)$. We can see that the determiner sense q is being taken as a generalized quantifier, that is, a relation between properties [4]. The contribution of the NP to the ultimate interpretation is the quantifier $Q = q(n)$, a property of properties. This is applied to the property $\lambda x.S$ that corresponds to the scope of the quantifier to give the clause sense $(q\ n\ x)S$.

Rule gen-quant stores all the assumptions A for the nominal of the general NP as nested assumptions of the single bind assumption it introduces. This ensures that the assumptions in A will not be discharged before the enclosing bind assumption is itself discharged. Assumption nesting serves two purposes. The first is to implement a form of *nested* Cooper storage enforcing the scoping constraint [42, 23] that a quantifier Q' appearing within another quantifier Q must either be scoped inside Q or have a scope than includes both Q and its scope. The second is to support the *capture* mechanism we use to interpret donkey sentences, further discussed below.

The discharge rule reflecting the first use of nested assumptions is

[quant-discharge] $P \rightsquigarrow A, B : (q\ s\ x)p$ **if** $P \rightsquigarrow A, \mathsf{bind}(x, q, s, B) : p_{\mathbf{t}}$

where x cannot occur free in A. This formal condition, which has been given a semantic explanation elsewhere [33], is necessary to block those derivations in which, roughly speaking, a variable occurs outside the scope of its binder. The type subscript $\mathbf{t}$ for the sense variable p should also be noted, indicating that this rule can apply only to interpretations whose sense part has the semantic type of a proposition. In general, scoping has to apply not only to proposition-type scopes but also to property-type scopes (meanings of common-NPs and verb-phrases) [35]. It would be straightforward to do this in Candide.

Under certain circumstances, rule quant-discharge should apply not only to the bind assumptions created for quantified NPs, but also to definite and indefinite bind assumptions that are to be given nonreferential readings. These arise when an NP is taken as having narrow scope with respect to another scoping operator. Examples include a generalized quantifier or a verb with an opaque argument:

(8) The thrust of each jet is measured by a sensor.

(9) John believes that a unicorn is approaching.

Let $a = \mathsf{bind}(x, k, s, B)$ be a definite or indefinite bind assumption. Q be the appropriate quantifier, $A, a : p$ be a conditional interpretation and let p have the appropriate type. Then, it will be possible to apply rule **quant-discharge** to the interpretation, resulting in a new interpretation $A : Q(\lambda x.p)$ whenever the derivation above this point includes the application of some scoping operator X whose scope includes $Q(\lambda x.p)$. This condition can be implemented in practice by letting definite or indefinite binds percolate up a derivation until a rule introducing a scoping operator is about to apply. Then, just before it does, the assumptions are optionally discharged as quantifiers.[11]

Given the foregoing refinement of the **quant-discharge** rule, we can see how alternative scopes for the interpretations of noun phrases in an utterance result from different orders of assumption discharge. This is exemplified by the abbreviated derivations in Figures 15 and 16, which give the two alternative scopings for the quantified noun phrases in the sentence

(10) Every driver controls a jet.

In the derivation shown in Figure 15, the indefinite bind assumption is discharged by rule **quant-discharge** as was just described, while in the derivation of Figure 16 the assumption is discharged by rule **ref-discharge**. In both cases, rule **gen-quant** introduces the bind assumption for the quantified NP "every driver" and rule **quant-discharge** discharges that assumption.

We noted above how assumption nesting is used to enforce a constraint on possible scopings. A second use of nesting in our system is to make indefinite noun phrase assumptions for the nominal in a quantified NP available to *capture* rules that may discharge the quantified NP's bind assumption. The purpose of capture rules is to allow the derivation of the intended interpretation for donkey sentences such as (11)

(11) Every driver that controls a jet closes it.

[11] The current version of Candide actually allows this only for indefinite bind assumptions nested within a quantified NP assumption about to be discharged. As a result, an indefinite NP can be interpreted in Candide only as referential (a discourse constant), as an existential quantifier immediately outscoped by a generalized quantifier, or as a captured quantifier.

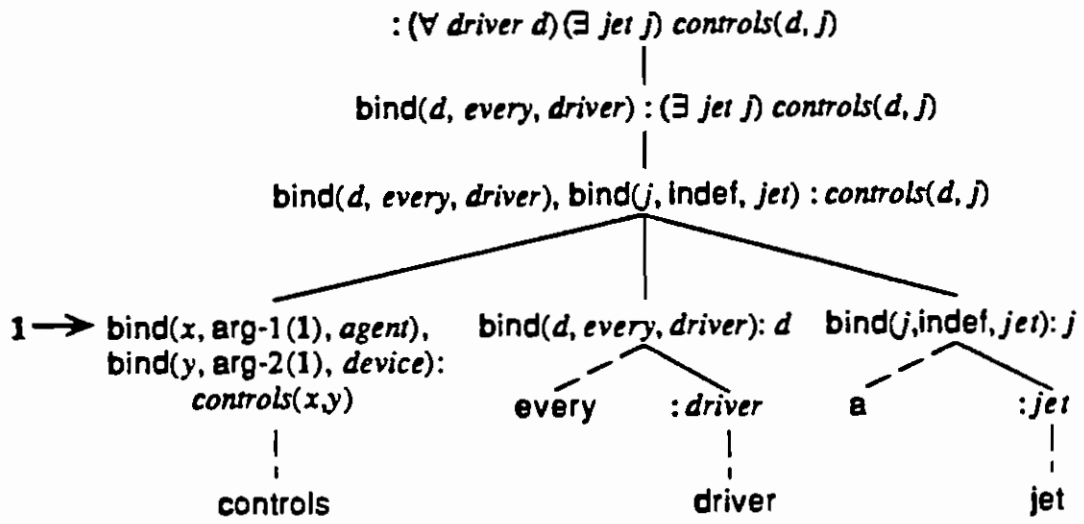

Figure 15: ∀∃ Interpretation

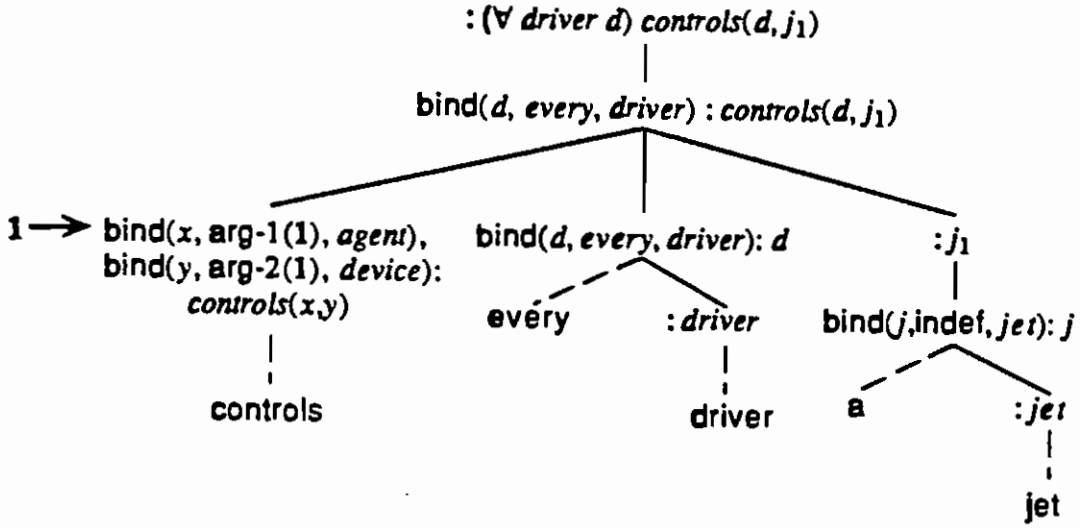

Figure 16: ∃∀ Interpretation

The basic operation of a capture rule is to discharge a bind assumption $\mathsf{bind}(x, q, s, A)$ for a quantified NP together with some subset A' of the indefinite bind assumptions in the nested assumption set A. Let $y_1, \ldots, y_k$ be the parameters bound by the assumptions in A'. Then capture will introduce an appropriate *polyadic quantifier* [41], which quantifies simultaneously over x and all the y_i. For example, the interpretation of (11) just before capture would be

$$\mathsf{bind}(x, \mathit{every}, \mathit{driver}|_d \mathit{controls}(d, y), \{\mathsf{bind}(y, \mathsf{indef}, \mathit{jet})\}) : \mathit{close}(x, y)$$

and after capture

$$: (\forall\ \mathit{jet}\ y)(\forall\ \mathit{driver}|_d \mathit{controls}(d, y)\ x)\mathit{close}(x, y) \tag{12}$$

In this case the polyadic quantifier reduces to two nested universal quantifiers. However, this interpretation of (11) is not without difficulties, as noted by Rooth [36] and Heim [21], among others. Furthermore, it will not generalize to the interpretation of sentences with a different generalized quantifier, for instance "most", instead of "every", since

$$: (\forall\ \mathit{jet}\ y)(\mathit{most}\ \mathit{driver}|_d \mathit{controls}(d, y)\ x)\mathit{close}(x, y) \tag{13}$$

is not an acceptable interpretation of

(14) Most drivers that control a jet close it.

(Consider a model with three drivers d_1, d_2 and d_3 and two jets j_1 and j_2 such that d_1 and d_2 control and close j_1, and d_3 just controls j_2. This model does not satisfy (13) but seems to satisfy the intuitive meaning of the sentence.) Since our purpose here is only to explain how such interpretations can be built incrementally, and not to argue for particular semantic analyses, we will not discuss these problems further.

Figure 17 shows a simplified derivation of interpretation (12) for sentence (11). At the clause node 4, the interpretation has two bind assumptions, one for the quantified NP

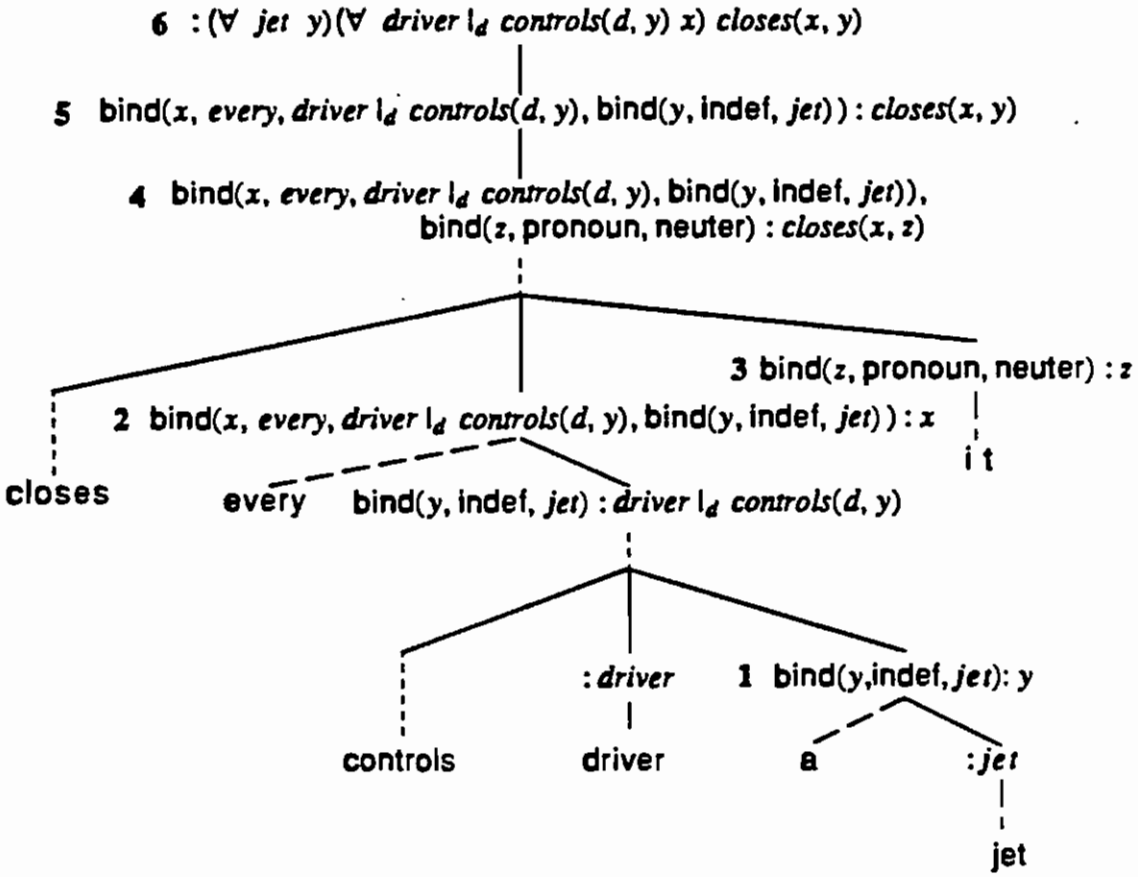

Figure 17: Derivation for Donkey Sentence

"every driver..." interpreted at node 2 and the other for the pronoun "it" interpreted at node 3. The bind assumption for the indefinite NP "a jet" introduced at node 1 is now nested within the quantified NP assumption. The derivation continues from node 4 with the identification of the indefinite noun phrase as the antecedent for the pronoun. This is done at node 5 by the application of rule ref-discharge, which can discharge a pronominal reference assumption against another undischarged bind assumption. Capture completes the derivation, discharging the quantified NP assumption and the nested indefinite assumption on node 5 to yield node 6. In general, several levels of nested assumptions may be unnested on capture, as in (15).

(15) Every friend of a member of a club visits it with him.

A (simplified) interpretation for (17) would be:

$$: (\forall\ \mathit{club}\ c)(\forall\ \mathit{member\text{-}of}(c)\ m)(\forall\ \mathit{friend\text{-}of}(m)\ f)\mathit{visits\text{-}with}(f, c, m)$$

Furthermore, definite assumptions must also be considered in capture. For example, to interpret (16) we need to discharge the assumptions in (17) so as to produce the intended interpretation in (18).

(16) Every friend of the author of a book praises it to her.

$$\begin{array}{l}\mathsf{bind}(f.\ \mathit{every}, \mathit{friend\text{-}of}(a),\\ \quad \mathsf{bind}(a, \mathsf{def}, \mathit{author\text{-}of}(b),\\ \quad\quad \mathsf{bind}(b, \mathsf{indef}, \mathit{book})) : \mathit{praises\text{-}to}(f, b, a)\end{array} \tag{17}$$

$$: (\forall\ \mathit{book}\ b)(\mathit{the}\ \mathit{author\text{-}of}(b)\ a)(\forall\ \mathit{friend\text{-}of}(a)\ f)\mathit{praises\text{-}to}(f, b, a) \tag{18}$$

In general, given a capturing bind assumption, the capture process involves selecting some binding assumptions to be captured from the set of assumptions nested in the capturing assumption, and recording the scoping constraints between those assumptions. It is

convenient to represent the result of this process as a *scope tree* that represents the captured assumptions and their scope relations. In the formal definition that follows, we represent a scope tree as a pair (α, T) where α is a bind assumption and T is a (possibly empty) set of scope trees whose bind assumptions must outscope α. An appropriate translation will map the capturing assumption and the selected scope tree to a polyadic quantifier.

Given a capturing assumption $\mathsf{bind}(x, q, s, A)$, we take a bind assumption $\mathsf{bind}(y, k, r, C)$ that is within A to be capturable just in case (1) the assumption type k is indefinite (indef) or (2) the assumption type is definite (def) and an assumption in the (nonempty) set of nested assumptions C is being captured. With this definition, all the definite and indefinite assumptions for the NPs in the subject nominal of sentence (16) are capturable provided that "a book" is captured. In this case, the derivation of the preferred interpretation (18) is allowed. If "a book" is not captured, neither of the indefinites will be, and the interpretation of the sentence will be that all the friends of the author of a specific book praise the book to the author.

The capture process may then be more precisely defined as follows:

> Given capturing assumption $\mathsf{bind}(x, q, s, A)$, construct a set of "leftover" assumptions B and a scope tree $t = (\mathsf{bind}(x, q, s), T)$ where
>
> - $A' = \{\alpha_1, \ldots, \alpha_m\} \subseteq A$ is a set of capturable assumptions for which scope trees t_i and leftover assumptions B_i have been determined
> - $B = (A - A') \cup \bigcup_i B_i$ and $T = \{t_1, \ldots, t_m\}$
> - assumption $\alpha_i = \mathsf{bind}(y_i, k_i, r_i, C_i)$ is capturable with scope tree $t_i = (\mathsf{bind}(y, k, r), U_i)$ and leftover assumptions B_i if either
>   - $k_i = \mathsf{indef}$, and C_i, B_i and U_i are empty, or
>   - k_i is def or indef, and α_i can capture a nonempty subset of assumptions from the nested assumption set C_i, with resulting leftover assumptions B_i and scope tree t_i.

As outlined earlier, the captured assumptions in a scope tree must be combined with the capturing quantifier to produce the appropriate polyadic quantifier. As was noted, in general the choice of quantifier is a difficult one, and all current proposals for interpreting donkey sentences have serious deficiencies. We will limit ourselves here to giving a usable translation for a universal quantifier with captured assumptions, the simplest case of which was originally considered in discourse representation theory (DRT) [24]. In this case, let $t = (\mathsf{bind}(x, \mathit{every}, s), T)$ be the quantifier tree built by the capture process. Such a tree encodes a partial order between assumptions: assumption α must outscope assumption β just in case α is a node in the subtree rooted at β. Any linearization of the partial order represented by the tree will lead to a polyadic quantifier of the form

$$\lambda R.(\forall\ s_1\ x_1)\cdots(\forall\ s_n\ x_n)R(x_1,\ldots,x_n)$$

where R will be filled by the matrix of the quantification.

Looking again at the interpretation of sentence (16), we see that the rules just given will translate the result of capturing both nested bind assumptions into the nested quantifiers in (18).

Given the foregoing discussion, the general form of the capture discharge rule is thus:

[capture] $\quad P \rightsquigarrow C, B : Q(\lambda x_1 \cdots x_n\ x.p)$ **if** $P \rightsquigarrow C, \mathsf{bind}(x, q, s, A) : p_t$

The application of this rule is subject to the following conditions: (1) q must be a general quantifier, (2) assumption $\mathsf{bind}(x, q, s, A)$ captures some assumptions from A resulting in a scope tree t and leftover assumptions B, (3) $x_1, \ldots, x_n$ are the parameters for the captured assumptions and (4) Q is the appropriate polyadic quantifier.

4 Conclusion

We have presented a system for the incremental interpretation of natural-language utterances in context. The main goal of the work was to account for the influences of context on the combinatorial aspects of interpretation, while preserving compositionality to the extent possible. To achieve this goal, we introduced a representational device, conditional interpretations, and a rule system for constructing them. Conditional interpretations represent the potential contributions of phrases to the interpretation of an utterance. Structural and discharge rules specify how phrase interpretations are combined, and how they are elaborated with respect to context. The control structure defined by the rules determines the points in the interpretation process at which sufficient information becomes available to carry out specific inferential interpretation steps.

Our interpretation system was conceived as a first step out of a dilemma of compositionality created by the fact that, in general, the choice of interpretation for a phrase depends on the phrase's context of utterance in complex and open-ended ways. The dilemma is represented by two opposing alternatives, which we now recapitulate.

The first alternative is represented by Montague grammar and its descendants. In these systems, the effects of context on the interpretation of a phrase must be built into the denotation of the phrase. This method preserves incrementality of interpretation, but leads to complex and otherwise unmotivated denotations. Furthermore, we know of no principled way to design such denotations to accommodate open-ended influences of context on interpretation.[12] Cooper's work on quantifier storage is probably the best-known attempt to provide such open-endedness. Since the scopes of quantifiers appearing in a phrase are underspecified, in Cooper's system phrase, denotations become pairs consisting of a Montague-style denotation and a store representing those quantifiers in the phrase whose scope is still to be determined. Quantifiers whose scope is determined are removed from storage and applied to their scope. Cooper denotations are thus hybrids, which cross over

[12]These difficulties are reminiscent of the problems of providing denotational semantics to imperative programming languages, in which the store plays the role of context.

from the strictly semantic domain of Montague denotations to the domain of the interpretation process. In Cooper's system, as in all the representatives of this first alternative, the directness of the semantics is sacrificed to maintain compositionality and incrementality in the face of contextual effects.

The other interpretation alternative is exemplified by discourse-representation theory (DRT) [24]. In DRT, what we would call the complete interpretation of an utterance is derived compositionally from an intermediate representation, called a discourse-representation structure (DRS). However, the rules for DRS construction presented by Kamp [24] are not compositional or incremental: the DRS for a phrase is found only as a by-product of finding the DRS for the embedding discourse. In particular, DRS-construction rules apply only after the relative scope of noun phrases and anaphoric bindings have been determined. Thus, Kamp's system does not provide a principled framework for incremental interpretation, that is, DRT sacrifices incrementality so that contextual effects and semantic simplicity may coexist.

This tension has been at least implicitly recognized by several authors. Barwise [3], in a formal development of ideas implicit in Heim's file-change semantics [20], proposes the direct interpretation of phrases as partial assignments of values to variables. His interpretation rules are compositional, but are applied to analyses in which scope and coreference relations have already been fully specified. Dynamic Montague grammar, introduced by Groenendijk and Stokhof [13] and further developed by Chierchia [6], attempts to treat context as a first-class citizen by adding a context-change dimension to the phrase denotations. This provides an elegant treatment of the interaction between quantification and reference, but again only after scopes and coreference relations have been fixed. Finally Zeevat [46] gives a set of incremental interpretation rules for a variant of DRT, but loses compositionality in the sense that his semantic operations are applied to formulas in a logical language rather than to denotations. Again, scoping and coreference relations must be fixed in advance.

Research on incremental interpretation in computational linguistics has been mostly concerned with its inferential rather than its combinatorial aspects. Neither that work nor work

on the combinatorial side of single sentence interpretation has paid much attention to the effects of context on combinatorial questions. Furthermore, our approach to interpretation offers a flexibility in the sequencing of interpretation decisions that has not been achieved in typical inferential interpretation systems. For example, the Pundit system [8, 9] deals with pragmatic problems in a fixed order, even though the necessary information may become available in a different order. The TACITUS system [22] avoids the ordering problem by eschewing incrementality and treating interpretation as global optimization. A higher degree of incrementality was achieved by Mellish [28] and Haddock [18], but the structure in their systems was insufficient to handle as wide a range of combinatorial problems as we do.

Our system suffers from a variety of limitations in linguistic coverage and discourse modeling that could be alleviated with further interpretation rules and the incorporation of more detailed discourse models. However, its main limitation is that discourse entities cannot depend upon undischarged interpretation assumptions. That kind of parameterization of the discourse model would be needed to interpret correctly the dependencies between multiple referring expressions that Ayuso [2] identified as a problem for Webber's type of discourse model [44].

It is fair to say that, even though the developments we have surveyed here bring more of context into the compositional fold, they do not address the central issue of the local indeterminacy of context, that is, they do not provide phrase interpretations that can be incrementally elaborated as contextual relations, such as coreference and scope, are discovered. This is, of course, the main feature of our system.

Acknowledgments

The development of the Candide system was funded by DARPA under Contract N000-39-84-C-0524. The preparation of this paper was partly supported by a gift from the System Development Foundation as part of a coordinated research effort with the Center for the Study of Language and Information, Stanford University.

We would like to thank David Israel, Ray Perrault, and Stuart Shieber for their many helpful discussions on this work, and Barbara Grosz, Julia Hirschberg, and David Israel for detailed comments on an earlier version of the paper. We would also like to thank Barney Pell, who contributed greatly to the implementation of the Candide system, and Lauri Karttunen, who wrote the grammar.

References

[1] A. V. Aho, R. Sethi, and J. D. Ullman. *Compilers: Principles, Techniques, and Tools.* Addison-Wesley, Reading, Massachusetts, 1986.

[2] D. M. Ayuso. Discourse entities in Janus. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, University of British Columbia, Vancouver, Canada, June 1989. University of British Columbia, Association For Computational Linguistics.

[3] J. Barwise. Noun phrases, generalized quantifiers and anaphora. In P. Gärdenfors, editor, *Generalized Quantifiers*, pages 1–29. D. Reidel, Dordrecht, Holland, 1987.

[4] J. Barwise and R. Cooper. Generalized quantifiers and natural language. *Linguistics and Philosophy*, 4:159–219, 1981.

[5] D. Carter. *Interpreting Anaphors in Natural Language Texts.* Ellis Horwood, Chichester, England, 1987.

[6] G. Chierchia. Dynamic generalized quantifiers and donkey anaphora. In M. Krifka, editor, *Proceedings of the 1988 Tübingen Conference*, pages 53–83. Seminar für natürlichsprachliche Systeme der Universität Tübingen, 1988.

[7] R. Cooper. *Quantification and Syntactic Theory.* D. Reidel, Dordrecht, Netherlands, 1983.

[8] D. A. Dahl. Focusing and reference resolution in Pundit. In *AAAI-86, Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 1083–1087. Philadelphia, Pennsylvania, 1986. University of Pennsylvania, American Association for Artificial Intelligence.

[9] D. A. Dahl, M. S. Palmer, and R. J. Passonneau. Nominalizations in Pundit. In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, pages 131–139, Stanford University, Stanford, California, 1987. Association For Computational Linguistics.

[10] D. Davidson. The logical form of action sentences. In N. Rescher, editor, *The Logic of Decision and Action*. University of Pittsburgh Press, 1967.

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

[12] M. P. Georgeff and A. L. Lansky. Procedural knowledge. *Proceedings of the IEEE. Special Issue on Knowledge Representation*, pages 1383–1398, 1986.

[13] J. Groenendijk and M. Stokhof. Dynamic Montague grammar. Presented at the workshop on Discourse Representation Theory, Stuttgart, December 1987.

[14] B. J. Grosz. The representation and use of focus in dialogue understanding. Technical Report 151, SRI International, Menlo Park, California, 1977.

[15] B. J. Grosz, K. S. Jones, and B. L. Webber, editors. *Readings in Natural Language Processing*. Morgan Kaufmann, Los Altos, California, 1986.

[16] B. J. Grosz, A. K. Joshi, and S. Weinstein. Providing a unified account of definite noun phrases in discourse. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, pages 44–50, Massachusetts Institute of Technology, Cambridge, Massachusetts, June 1983. Association For Computational Linguistics.

[17] B. J. Grosz and C. L. Sidner. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204, 1986.

[18] N. J. Haddock. Incremental interpretation and combinatory categorial grammar. In N. Haddock, E. Klein, and G. Morril, editors, *Edinburgh Working Papers in Cognitive Science, Vol. I: Categorial Grammar, Unification Grammar and Parsing*, pages 71–84. Edinburgh, Scotland, 1987.

[19] P.-K. Halvorsen and R. M. Kaplan. Projections and semantic description in lexical-functional grammar. In *Proceedings of the International Conference on Fifth Generation Computer Systems*, pages 1116–1122, Tokyo, Japan, November 1988. Institute for New Generation Computer Technology.

[20] I. R. Heim. *The Semantics of Definite and Indefinite Noun Phrases.* PhD thesis, Department of Linguistics, University of Massachusetts, Amherst, Massachusetts. September 1982.

[21] I. R. Heim. E-type pronouns and donkey anaphora. Presented at the workshop on Discourse Representation Theory, Stuttgart, December 1987.

[22] J. R. Hobbs. Interpretation as abduction. In *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics*, State University of New York, Buffalo, New York, June 1988. Association For Computational Linguistics.

[23] J. R. Hobbs and S. M. Shieber. An algorithm for generating quantifier scopings. *Computational Linguistics*, 13:47–63, 1987.

[24] H. Kamp. A theory of truth and semantic interpretation. In J. A. G. Groenendijk, T. M. V. Janssen, and M. B. J. Stokhof, editors, *Formal Methods in the Study of Language*, pages 277–322. Mathematisch Centrum, Amsterdam, Netherlands, 1981.

[25] D. K. Lewis. General semantics. In D. Davidson and G. Harman, editors, *Semantics of Natural Language*. D. Reidel, Dordrecht, 1972.

[26] W. D. Mann and S. A. Thompson. Rhetorical structure theory: A theory of text organization. Technical Report ISI/RR-87-190, USC/Information Sciences Institute, Marina del Rey, California, 1987.

[27] M. Marcus, D. Hindle, and M. Fleck. D-theory: Talking about talking about trees. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, University of Toronto, Toronto, Canada, June 1982. Association for Computational Linguistics.

[28] C. S. Mellish. *Computer Interpretation of Natural Language Descriptions.* Ellis Horwood, Chichester, England, 1985.

[29] R. Montague. The proper treatment of quantification in ordinary English. In R. H. Thomason, editor, *Formal Philosphy.* Yale University Press, 1973.

[30] M. S. Palmer, D. A. Dahl, R. J. Schiffman, L. Hirschman, M. Linebarger, and J. Dowding. Recovering implicit information. In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, pages 10–19, Columbia University, New York, New York, June 1986. Association For Computational Linguistics.

[31] T. Parsons. The progressive in English: Events, states and processes. *Linguistics and Philosophy*, 12(2):213–241, 1989.

[32] F. C. Pereira. *Logic for Natural Language Analysis*. PhD thesis, University of Edinburgh, Scotland, 1982. Reprinted as Technical Note 275, January 1983, Artificial Intelligence Center, SRI International, Menlo Park, California.

[33] F. C. N. Pereira. A calculus for semantic composition and scoping. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, pages 152–160, University of British Columbia, Vancouver, Canada, June 1989. Association For Computational Linguistics.

[34] P. M. Postal. Anaphoric islands. In R. Binnick et al., editors, *Papers from the Fifth Regional Meeting of the Chicago Linguistic Society*, pages 205–239, 1869.

[35] C. Roberts. *Modal Subordination, Anaphora and Distributivity*. PhD thesis, Department of Linguistics, University of Massachusetts, Amherst, Massachusetts, February 1987.

[36] M. Rooth. Noun phrase interpretation in Montague grammar, file change semantics, and situation semantics. In P. Gärdenfors, editor, *Generalized Quantifiers*, pages 237–268. D. Reidel, Dordrecht, Holland, 1987.

[37] S. J. Rosenschein and S. M. Shieber. Translating English into logical form. In *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics*, pages 1–8, University of Toronto, Toronto, Canada, 1982. Association for Computational Linguistics.

[38] L. Schubert and F. J. Pelletier. From English to logic: Context-free computation of 'conventional' logical translations. *American Journal of Computational Linguistics*, 10:165–176, 1982. Reprinted in [15].

[39] S. M. Shieber. Criteria for designing computer facilities for linguistic analysis. *Linguistics*, 23:189–211, 1985.

[40] C. L. Sidner. Focusing in the comprehension of definite anaphora. In M. Brady and R. Berwick, editors, *Computational Models of Discourse*. MIT Press, Cambridge, Massachusetts, 1983. Reprinted in [15].

[41] J. van Benthem. Polyadic quantifiers. *Linguistics and Philosophy*, 12(4):437–464, August 1989.

[42] J. van Eijck. *Aspects of Quantification in Natural Language*. PhD thesis, University of Groningen, Groningen, Holland, February 1985.

[43] G. Ward, R. Sproat, and G. McKoon. A pragmatic analysis of so-called anaphoric islands. In preparation.

[44] B. L. Webber. So what can we talk about now? In M. Brady and R. Berwick, editors, *Computational Models of Discourse*. MIT Press, Cambridge, Massachusetts, 1983. Reprinted in [15].

[45] W. A. Woods. Semantics and quantification in natural language question answering. In M. Yovits, editor, *Advances in Computers, Vol. 17*, pages 2–64. Academic Press, New York, 1978. Reprinted in [15].

[46] H. Zeevat. A compositional approach to Discourse Representation Theory. *Linguistics and Philosophy*, 12(1):95–131, February 1989.