ANAPHORA RESOLUTION IN SLOT GRAMMAR

Shalom Lappin and Michael McCord

IBM T. J. Watson Research Center
P.O.B. 704
Yorktown Heights, NY 10598

We present three algorithms for resolving anaphora in Slot Grammar: (1) an algorithm for interpreting elliptical VPs in antecedent-contained deletion structures, subdeletion constructions, and intersentential cases; (2) a syntactic filter on pronominal coreference; and (3) an algorithm for identifying the binder of an anaphor (reflexive pronoun or the reciprocal phrase "each other"). These algorithms operate on the output of a Slot Grammar parser, and, like the parser, they run in Prolog. The VP anaphora algorithm implements an S-structure analysis of VP ellipsis that we argue provides a more unified and empirically motivated treatment of VP anaphora resolution than analyses that attempt to interpret elliptical VPs at a level of logical form. Each algorithm can operate independently of the others, and we have incorporated each into an integrated anaphora resolution component. The interpreted elliptical VP structures that the VP anaphora algorithm produces provide the input to the two NP anaphora resolution procedures. The integrated anaphora resolution component provides a powerful syntactically driven module for generating partially interpreted representations that can serve as input to semantic and discourse interpretation systems.

1 INTRODUCTION

In this paper[1] we present algorithms for handling three different sorts of anaphora within Slot Grammar (McCord 1980, 1989b, 1990). These algorithms are second-pass procedures that operate on the output of a Slot Grammar parser. The parser and the algorithms constituting the anaphora resolution component run in Prolog. In Section 2 we present a brief overview of Slot Grammar and the parser that implements it. This section also includes a description of an alternative network representation of parser output on which the algorithms operate. In Section 3 we propose an analysis of VP anaphora that involves applying rules of interpretation directly to S-structure (parsed surface structure) rather than to LF (logical form), as required by several recent accounts. We provide in Section 3.1 theoretical motivation for preferring our analysis to an LF treatment. In Section 3.2 we present a schematic statement of the algorithm that implements this analysis in Slot Grammar, and illustrate the algorithm with examples of its output. Section 4 is devoted to a syntactic filter on pronominal anaphora that identifies noncoreferential NP-pronoun pairs within a sentence. A more detailed presentation of the filter algorithm is given in Lappin and McCord (1990). Section 5 contains a rule for locating possible NP antecedents for anaphors (reflexive pronouns and reciprocals). The conjunction of the latter two algorithms has roughly the same extension as Chomsky's (1981, 1986a) binding theory. However, while the conditions of the binding theory are stated in terms of the configurational relation of c-command, the coreference filter and anaphor binding algorithm employ the head-complement structures defined by Slot Grammar.[2]

The three algorithms that make up the anaphora resolution component of Slot Grammar are fully modular in that they apply independently of each other. Any two algorithms in this set can be conjoined. Moreover, both the pronominal noncoreference filter and anaphor binding algorithms have been combined with the VP anaphora algorithm to construct an integrated system of anaphora resolution in which the two NP anaphora rules apply to the results of VP anaphora interpretation.[3] In Section 6 we illustrate the operation of the integrated system with examples of the representations it generates.

2 SLOT GRAMMAR

The original work on Slot Grammar was done around 1976–1978 and appeared in McCord (1980). Recently, a new version (McCord 1989b, 1990) was developed in a logic programming framework, in connection with the machine translation system **LMT** (McCord 1989a, 1989c, 1989d).

Slot Grammar is lexicalist and is dependency oriented. Every phrase has a head word (with a given word sense and morphosyntactic features). The constituents of a phrase besides the head word, also called the *modifiers* of the head, are obtained by "filling" *slots* associated with the head. Slots are symbols like subj, obj, and iobj representing grammatical relations, and are associated with a word

(sense) in two ways. The lexical entry for the word specifies a set of *complement* slots, corresponding to logical arguments of the word sense, and the grammar specifies a set of *adjunct* slots for each part of speech.[4] A complement slot can be filled at most once, and an adjunct slot can by default be filled any number of times.

The phenomena treated by augmented phrase structure rules in some grammatical systems are treated modularly by several different types of rules in Slot Grammar. The most important type of rule is the (*slot*) *filler* rule, which gives conditions (expressed largely through unification) on the filler phrase and its relations to the higher phrase.

Filler rules are stated (normally) without reference to conditions on *order* among constituents. But there are separately stated *ordering rules*.[5] *Slot/head* ordering rules state conditions on the position (left or right) of the slot (filler) relative to the head word. *Slot/slot* ordering rules place conditions on the relative left-to-right order of (the fillers of) two slots.

A slot is *obligatory* (not *optional*) if it must be filled, either in the current phrase or in a raised position through left movement or coordination. Adjunct slots are always optional. Complement slots are optional by default, but they may be specified to be obligatory in a particular lexical entry, or they may be so specified in the grammar by *obligatory slot rules*. Such rules may be unconditional or be conditional on the characteristics of the higher phrase. They also may specify that a slot is obligatory *relative* to the filling of another slot. For example, the direct object slot in English may be declared obligatory on the condition that the indirect object slot is filled by a noun phrase.

One aim of Slot Grammar is to develop a powerful language-independent module, a "shell," which can be used together with language-dependent modules, reducing the effort of writing grammars for new languages. The Slot Grammar shell module includes the parser, which is a bottom-up chart parser. It also includes most of the treatment of coordination, unbounded dependencies, controlled subjects, and punctuation. And the shell contains a system for evaluating parses, extending Heidorn's (1982) parse metric. The Slot Grammar evaluator is used not only for ranking final parses, as with Heidorn's, but also for pruning away unlikely partial analyses *during* parsing, thus reducing the problem of parse space explosion. Parse evaluation expresses preferences for close attachment, for choice of complements over adjuncts, and for parallelism in coordination.

Although the shell contains *most* of the treatment of the above phenomena (coordination, etc.), a small part of their treatment is necessarily language dependent. A (language-specific) grammar can include for instance (1) rules for coordinating feature structures that override the defaults in the shell; (2) declarations of slots (called *extraposer* slots) that allow left extraposition of other slots out of their fillers; (3) language-specific rules for punctuation that override defaults; and (4) language-specific controls over parse evaluation that override defaults.

Currently, Slot Grammars are being developed for English (**ESG**) by McCord, for Danish (**DSG**) by Arendse Bernth, and for German (**GSG**) by Ulrike Schwall. **ESG** uses two lexicons: (1) a hand-coded lexicon of about 3,700 common words, and (2) the **UDICT** lexicon (Byrd 1983; Klavans and Wacholder 1989) having over 60,000 lemmas, with a heuristic interface that produces Slot Grammar–style entries.

Our anaphora algorithms apply in a second pass to the parse output; the remainder of this section describes Slot Grammar syntactic analysis structures.

A syntactic structure is a tree; each node of the tree represents a phrase in the sentence and has a unique head word. Formally, a phrase is represented by a term

```
phrase(X,H,Sense,Features,SlotFrame,Ext,Mods),
```

where the components are as follows. (1) **X** is a logical variable called the *marker* of the phrase. Unifications of the marker play a crucial role in the anaphora algorithms. (2) **H** is an integer representing the position of the head word of the phrase. This integer identifies the phrase uniquely, and is used in the anaphora algorithms as the way of referring to phrases. (3) **Sense** is the word sense of the head word. (4) **Features** is the feature structure of the head word *and* of the phrase. It is a logic term (not an attribute-value list), which is generally rather sparse in information, showing mainly the part of speech and inflectional features of the head word. (5) **SlotFrame** is the list of complement slots, each slot being in the internal form **slot(Slot,Ob,X)**, where **Slot** is the slot name, **Ob** shows whether it is an obligatory form of **Slot**, and **X** is the *slot marker*. The slot marker is unified (essentially) with the marker of the filler phrase when the slot is filled, even remotely, as in left movement or coordination. Such unifications are important for the anaphora algorithms. (6) **Ext** is the list of slots that have been *extraposed* or *raised* to the level of the current phrase. (7) The last component **Mods** represents the modifiers (daughters) of the phrase, and is of the form **mods(LMods,RMods)** where **LMods** and **RMods** are the lists of left modifiers and right modifiers, respectively. Each member of a modifier list is of the form **Slot:Phrase** where **Slot** is a slot and **Phrase** is a phrase that fills **Slot**. Modifier lists reflect surface order, and a given slot may appear more than once (if it is an adjunct). Thus modifier lists are not attribute-value lists.

Figure 1 shows a sample parse produced by **ESG** for the sentence *Who did John say wanted to try to find him?* The tree is displayed by a procedure that uses only one line per node and exhibits tree structure lines on the left. In this display, each line (representing a node) shows (1) the tree connection lines, (2) the slot filled by the node, (3) the *word sense predication,* and (4) the feature structure. The feature structure is abbreviated here by a display option, showing only the part of speech. The word sense predication consists of the sense name of the head word with the

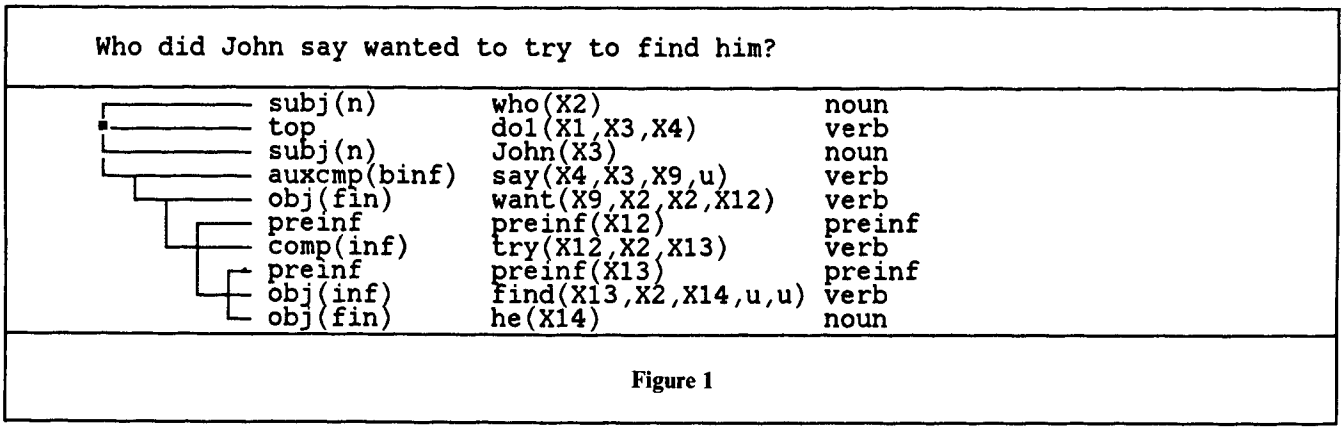

Figure 1

following arguments. The first argument is the marker variable for the phrase (node) itself; it is like an event or state variable for verbs. The remaining arguments are the marker variables of the slots in the complement slot frame (u signifies "unbound"). As can be seen in the display, the complement arguments are unified with the marker variables of the filler complement phrases. Note that in the example the marker X2 of the 'who' phrase is unified with the subject variables of "want," "try," and "find." (There are also some unifications created by adjunct slot filling, which will not be described here.)

For the operation of our anaphora algorithms, there is a preliminary step in which pertinent information about the parse tree is represented in a more convenient way for the algorithms. As indicated above, nodes (phrases) themselves are represented by the word numbers of their head words. Properties of phrases and relations between them are represented by unit clauses (predications) involving these integers (and other data), which are asserted into the Prolog workspace. Because of this "dispersed" representation with a collection of unit clauses, the original phrase structure for the whole tree is first grounded (variables are bound to unique constants) before the unit clauses are created.

As an example of this clausal representation, the clause **hasarg(P,X)** says that phrase P has X as one of its arguments; i.e., X is the slot marker variable for one of the complement slots of P. For the above sample parse, then, we would get clauses

```
hasarg(5,'X2'). hasarg(5,'X12').
```

as information about the 'want' node (5).

As another example, the clause **phmarker(P,X)** is added when phrase P has marker X. Thus for the above sample, we would get the unit clause

```
phmarker(1,'X2').
```

An important predicate for our algorithms is **pharg**, defined by

```
pharg(P,Q) ← phmarker(P,X) & hasarg(Q,X).
```

This says that phrase P is an argument of phrase Q. This includes remote arguments and controlled subjects, because of the unifications of marker variables performed by the Slot Grammar parser. Thus for the above parse, we would get

```
pharg(1,5). pharg(1,7). pharg(1,9).
```

showing that "who" is an argument of "want," "try," and "find."

3 VP Anaphora

3.1. The Resolution of VP Anaphora at S-Structure

Before presenting our algorithm for the interpretation of VP anaphora structures, we will provide motivation for the general view of VP anaphora that the algorithm implements. We characterize VP anaphora as a relation between the head V and selected arguments and adjuncts of a structured empty, or partially empty, elliptical VP, and the head A and corresponding adjuncts of an antecedent VP. This relation is identified on S-structure parse representations. The VP anaphora interpretation procedure copies the head A of the antecedent VP into the position of the head of the elliptical VP, and specifies which arguments and adjuncts of the antecedent A are inherited by the elliptical V. In this way, it provides an interpretation of the elliptical VP. It is important to recognize that this procedure operates on S-structure representations rather than on a more abstract level of LF.

Let us briefly consider the case for an LF-based approach to VP anaphora resolution. The elliptical VP in the relative clause of the object NP in 1 is contained in the matrix VP, which is its antecedent.

1. Dulles suspected everyone who Angelton did.

As May (1985) observes, if we copy the matrix VP into the position of the empty VP at S-structure, an interpretive regress results. The empty VP will reappear in the copied matrix VP. May proposes to solve this problem by applying the operation of quantifier raising (QR) to the object NP in 1. QR adjoins the quantified NP to the matrix sentence to derive the LF representation 2.[6]

2. $[_{IP'}[_{NP_1}$ everyone who$_1$ Angelton did $[_{VP}]][_{IP}$Dulles $[_{VP}$ suspected $t_1]]]$

The matrix VP of the IP in 2 is assigned to the empty VP of the adjoined NP to obtain 3, the desired interpretation of 1.

3. $[_{IP'}[_{NP_1}$ everyone who Angelton $[_{VP}$ suspected $t_1]]$ $[_{IP}$Dulles $[_{VP}$ suspected $t_1]]]$

May concludes that antecedent-contained deletion (ACD) structures can only be interpreted by a VP copying rule that applies at LF.

There are at least two serious difficulties with May's analysis of ACD structures.[7] First, as Haik (1987) points out, the wh-phrase in the relative clause of an ACD sentence such as 1 is constrained by subjacency.

4a. John read everything which Bill believes he did.
b. *John read everything which Bill believes the claim that he did.

On May's analysis, the VP in the relative clause in 1 and 4a–b is empty at S-structure, and the wh-phrase binds a trace only after VP copying has applied to the LF produced by the movement of the object NP. But it is generally agreed that subjacency is a condition that constrains operator-trace binding chains only at S-structure.[8] Given May's account, there is no trace at S-structure for the wh-operator to bind in 1 and 4a–b. Therefore, it is unclear how, on this analysis, subjacency can constrain wh-movement in ACD structures.

May (in press) seeks to avoid this problem by suggesting that subjacency does, in fact, apply at LF. The examples in 5 indicate that this is not the case.

5a. At least one critic reviewed Mary's biography of each author.
b. Who did at least one critic review $[_{NP}$a biography of t]
c. *Who did at least one critic review $[_{NP}$Mary's biography of t]

5a permits two scope readings for "each author" relative to "at least one critic." On the narrow scope reading, a single critic reviewed all of Mary's biographies. When "each author" receives wide scope, there is at least one (possibly different) critic for each of Mary's biographies of an author. If we accept May's view that the scope of a quantified NP is, in part, defined in terms of the constituent to which it is adjoined by QR, the fact that "each author" can take wide scope relative to "at least one critic" indicates that QR can move the former NP out of the NP "Mary's biography of each author." But 5b–c shows that the latter NP defines a syntactic island for wh-movement. It seems, then, that subjacency does constrain binding chains visible at S-structure, but not scope assignment.

The fact that antecedent-contained VP anaphora exhibits subjacency effects strongly suggests that the elliptical VP in these cases is not necessarily empty at S-structure, but may contain a trace bound by a wh-phase (or other operator).

Second, May's analysis does not extend to the subdeletion variety of ACD, where arguments and adjuncts of an empty verb are realized within the partially elliptical VP that it heads, as in the sentences in 6.[9]

6a. John writes more books than Bill does articles.
b. The university gives more money to the library for books than the city does to the orchestra for instruments.
c. The university gives more money to the library for periodicals than it does for books.
d. John wrote more articles for the journal about politics than he did about linguistics.
e. John showed everything to Mary which he did to Bill.
f. Mary argues about politics with everyone who she does about linguistics.
g. Mary arrived in London before Sam did in New York.
h. John reviewed the play for *The New York Times* shortly after Bill did for *The Washington Post*.

As these examples illustrate, subdeletion occurs in a variety of syntactic environments, including comparative NPs (6a–d), relative clauses (6e–f), and adverbial phrases (6g–h). Given that May's analysis treats VP anaphora as a global relation between an empty VP and an antecedent VP, it does not cover subdeletion, where an anaphoric relation holds between the head and selected constituents of an elliptical VP and its antecedent. But the full and subdeletion varieties of ACD are closely related phenomena, and an analysis that provides a unified explanation of both types of VP anaphora is clearly preferable to an account that handles only one type of antecedent-contained anaphora.

It is possible to capture the properties of ACD structures that create problems for May's LF analysis if we assume that the empty VP of a VP anaphora environment is structured, and may contain arguments or adjuncts of the head.[10] The arguments appearing in a partially empty elliptical VP can be lexically realized, or they may be traces (or their counterparts in Slot Grammar). If we apply this analysis to 1, we obtain 7a as its S-structure.

7a. Dulles suspected $[_{NP}[_{N'}$everyone $[_{CP}$who$_1$ Angelton did $[_{VP}[_{V}][_{NP}t_1]]]]]$
b. Dulles suspected $[_{NP}[_{N'}$everyone $[_{CP}$who$_1$ Angelton $[_{VP}[_{V}$suspected$][_{NP}t_1]]]]]$

In 7a, the object of the head of the elliptical VP is realized as a trace, and the VP is interpreted by copying only the head, "suspected," of the antecedent VP into the position of the empty head to yield 7b. Therefore, the interpretive regress is avoided without QR. Moreover, subjacency violations can be identified at S-structure by computing the relation between the wh-phrase and the trace that it binds.

Our proposal handles subdeletion in a natural and straightforward way. The only difference between the subdeletion structures in 6 and the ACD structure in 7a is that

arguments and adjuncts of the elliptical verbs in 6 are lexically realized, while the object of the elliptical verb in 7a is not. The interpretation procedure is the same for the two cases. The head of the antecedent VP is copied into the head of the elliptical VP. If any arguments or adjuncts are missing in the elliptical VP and corresponding arguments or adjuncts are realized in the antecedent VP, the latter are inherited by the head of the former.

This approach can be extended to intersentential VP anaphora cases like 8.[11]

8. John arrived yesterday, and Mary did too.

We simply treat the anaphoric relation in these cases as holding between the head and constituents of a structured empty VP, and the head and counterpart constituents of a full VP in another conjunct or sentence.

Subdeletion is generally marginal with intersentential VP anaphora when arguments are left in the elliptical VP.

9a. ??John writes articles, and Bill does books.
b. ?Mary spoke to Max, but Sam won't to Lucy.

However, subdeletion with adjuncts in these structures is considerably better.

10a. John arrived today, and Bill did yesterday.
b. Max spoke after Mary, and Sam will before Lucy.

The fact that adjuncts can remain in partially empty VPs in these cases provides motivation for applying the structured (partially) empty VP analysis to intersentential VP anaphora.

3.2. AN ALGORITHM FOR VP ANAPHORA INTERPRETATION

We define the predicate *P is contained in Q* recursively as follows. A phrase P is *immediately contained* in a head Q iff (i) P is an argument of Q, or (ii) P is an adjunct of Q. P is *contained* in Q iff (i) P is immediately contained in Q, or (ii) P is immediately contained in a head R, and (the phrase with head) R is contained in Q. The following is a schematic description of our algorithm for VP anaphora resolution.

VP ANAPHORA ALGORITHM

A. Identify an elliptical verb-antecedent verb pair ⟨V,A⟩ as follows.
   1. An elliptical verb V is identified by the presence of an auxiliary verb or the infinitival complementizer "to," where the auxiliary verb or the complementizer does not have a realized verb complement.
   2. A candidate A for an antecedent of V is a verb that is not elliptical and not an auxiliary verb with a realized complement.
   3. Check that A and V stand in at least one of the following relations:
      a. V is contained in the clausal complement of a subordinate conjunction SC, and the SC-phrase is either (i) an adjunct of A, or (ii) an adjunct of a noun N and N heads an NP argument of A, or N heads the NP argument of an adjunct of A.
      b. V is contained in a relative clause that modifies a head noun N, N is contained in A, and, if a verb A′ is contained in A and N is contained in A′, then A′ is an infinitival complement of A or of a verb contained in A.
      c. V is contained in the right conjunct of a sentential conjunction S, and A is contained in the left conjunct of S.

B. Generate a new tree in which A is substituted for V as the head of the elliptical verb phrase VP′ that V heads, and A is assigned the agreement features required by the head of VP′. (We will refer to this new occurrence of A as A′).

C. Consider in sequence each argument slot $Slot_i$ in the argument frame of A.
   1. If $Slot_i$ is filled by a phrase C, then
      a. If there is a phrase C′ in VP′ that is of the appropriate type for filling $Slot_i$, then fill $Slot_i$ in the argument frame of A′ with the marker variable of C′. Else,
      b. Fill $Slot_i$ in A′ with the marker variable of C, and list C as a new argument of A′.
   2. If $Slot_i$ is empty in the frame of A, it remains empty in the frame of A′.

D. For each adjunct Adj of A, if there is no adjunct of the same type as Adj in VP′, then list Adj as a new adjunct of A′.

Part A of the algorithm specifies the procedures for identifying pairs whose first element is the head of an antecedent VP and whose second element is the head of an elliptical VP. Elliptical VPs are identified by the presence of a bare auxiliary verb or the bare complementizer "to." In fact, for reasons of convenience, we take bare auxiliaries and bare complementizers as standing for the head of an elliptical VP, and so the algorithm treats them as surrogate VP heads. A.3 defines the structural relations that hold between the head of an elliptical VP and the head of a possible antecedent VP.[12]

Part B describes the operation of generating a new interpreted VP anaphora tree in which the head of the antecedent VP is substituted for the head of the elliptical VP, and the features of the new head of the interpreted elliptical VP are adjusted in accordance with the requirements of this VP.

Part C characterizes a procedure for filtering the arguments of the antecedent verb to determine which of them are inherited by the head of the interpreted elliptical VP. Similarly, Part D describes the filtering process that gives the adjuncts of the antecedent verb that are inherited by the interpreted elliptical verb. The combination of the new verb heading the elliptical VP and the lists of arguments and adjuncts it inherits from the antecedent verb provide the interpretation of the elliptical VP.

To illustrate the Prolog implementation of this algorithm

on the basis of the network representation, we will give the clauses pertinent to A.3.b, which identifies the case in which the elliptical verb V is contained in a relative clause.

The top-level predicate for testing that A is an antecedent of an elliptical verb V (used for implementing A.3) is anaph(A,V). The clauses for this predicate relevant to A.3.b are

```
anaph(A,V) ← arel(A,V).
arel(A,V) ←
   A=/V & relcont(V,N) &
   (pharg(N,A) | pharg(N,T) & phadjunct(T,A)).
```

(Here A=/B means "A is not equal to B" in IBM Prolog.) The relation relcont(V,N) holds if V is contained in a relative clause adjunct of noun N. The predicate pharg (P,Q), which says that P is an argument of Q, was defined in Section 2 in terms of the network representation. The relation phadjunct(P,Q) says that P is an adjunct of Q, and is also straightforwardly defined in terms of the network.

Let us consider several examples of the VP algorithm's results. The system produces the following output. For each ESG analysis of the input sentence, the parse tree is displayed, and then all pairs ⟨antecedent verb, elliptical verb⟩ found by the algorithm are displayed. Then, for each such pair, the following three things are displayed: (1) the new arguments inherited by the interpreted elliptical verb from its antecedent; (2) the new adjuncts inherited by the interpreted elliptical verb from its antecedent; and (3) the interpreted VP anaphora tree, as a modification of the original parse tree.

11 in Figure 2 shows the output for May's ACD example 1. In the parse tree, variable X5 in the object slot of the complement frame for the auxiliary "did" unifies with the phrase marker of the head of the relative "everyone" (and that of the wh-phrase "who"). Moreover, this variable is marked as a trace in the internal representation of the phrase structure from which the tree is projected. (Such a trace is marked in the feature structure of the verb—although it is not shown in the following abbreviated display.) Hence, the parse tree corresponds to the S-structure given in 7a. The VP anaphora algorithm identifies "suspected" as the antecedent of the elliptical verb (represented by the auxiliary), and substitutes it for the auxiliary in the interpreted VP anaphora tree. No arguments or adjuncts are inherited from the antecedent verb. The interpreted VP anaphora tree in 11 is the SG counterpart of the interpreted S-structure 7b.

A similar ACD case involving an elliptical VP that follows a bare occurrence of the complementizer "to" is given in 12 in Figure 3. Here, the interpreted verb "write" inherits the object argument "notes," while its indirect object argument is a trace bound by the wh-phrase corresponding to the head of the relative clause.

In the following examples, we give the output in an abbreviated way in order to save space. The uninterpreted tree is not shown, and the analysis resulting from the

```
11. Dulles suspected everyone who Angelton did.
Syntactic analysis time = 87 msec.

   subj Dulles(X4)          noun(prop,nom.X2,nwh)
   top  suspect(X1,X4,X5)   verb(fin(pers3,X2,past,X3))
   obj  everyone(X5)        noun(pron(all),acc.sg,X6)
      obj  who(X5)          noun(pron(wh),X10.sg,wh)
      subj Angelton(X9)     noun(prop,nom.X8,nwh)
      nrel do(X7,X9,X5)     verb(fin(pers3,X8,past,dep:dcl:wh))

Antecedent verb-elliptical verb pairs: suspect.2 - do.6
Elliptical verb-new argument pairs: none
Elliptical verb-new adjunct pairs: none

Interpreted VP anaphora tree:
   subj Dulles(X4)          noun(prop,nom.X2,nwh)
   top  suspect(X1,X4,X5)   verb(fin(pers3,X2,past,X3))
   obj  everyone(X5)        noun(pron(all),acc.sg,X6)
      obj  who(X5)          noun(pron(wh),X10.sg,wh)
      subj Angelton(X9)     noun(prop,nom.X8,nwh)
      nrel suspect(X7,X9,X5) verb(fin(pers3,X8,past,dep:dcl:wh))

Anaphora analysis time = 69 msec.
```

Figure 2

12. John wrote notes to everyone who asked him to.

```
subj     John(X3)              noun(prop,nom.sg,nwh)
top      write(X1,X3,X4,X5,u)  verb(fin(pers3,sg,past,X2))
obj      note(X4)              noun(cn,acc.pl,nwh)
iobj     to(X7,X5)             prep(to,X8,e(X5))
objprep  everyone(X5)          noun(pron(all),acc.sg,X8)
subj     who(X5)               noun(pron(wh),X13.sg,wh)
nrel     ask(X9,X5,X11,X12)    verb(fin(X10,sg,past,dep:dcl:wh))
obj      he(X11)               noun(pron(defprn),acc.sg,nwh)
comp     preinf(X12)           preinf
```

Antecedent verb-elliptical verb pairs: write.2 - preinf.9

Elliptical verb-new argument pairs: preinf.9 - note.3

Elliptical verb-new adjunct pairs: none

Interpreted VP anaphora tree:

```
subj     John(X3)              noun(prop,nom.sg,nwh)
top      write(X1,X3,X4,X5,u)  verb(fin(pers3,sg,past,X2))
obj      note(X4)              noun(cn,acc.pl,nwh)
iobj     to(X7,X5)             prep(to,X8,e(X5))
objprep  everyone(X5)          noun(pron(all),acc.sg,X8)
subj     who(X5)               noun(pron(wh),X13.sg,wh)
nrel     ask(X9,X5,X11,X12)    verb(fin(X10,sg,past,dep:dcl:wh))
obj      he(X11)               noun(pron(defprn),acc.sg,nwh)
comp     preinf(X12)           preinf
auxcmp   write1(X15,X11,X4,X5) verb(inf(bare))
```

Figure 3

algorithm is shown in an abbreviated linear form, consisting simply of head words and their arguments.

In 13, both "promise" and its infinitival complement "read" satisfy the condition given in A.3.b on the antecedent of an elliptical verb in a relative clause. Therefore, the algorithm correctly generates two possible interpreted VP anaphora trees. Tree 1 gives the reading on which "promise" is taken as the antecedent of the head of the empty VP in the relative clause, and the infinitival clause headed by "read" is inherited as a new argument of the interpreted verb. Tree 2 specifies the interpretation where "read" is substituted for the head of the empty verb, and the trace of the relative operator "which" is its (noninherited) argument. The algorithm correctly excludes "said" as a possible antecedent of the empty verb in the relative clause in 14. This is because it has a tensed rather than an infinitival complement that contains the verb that contains the noun modified by the relative clause. Therefore, the algorithm produces only two possible interpretations for 14, which are represented by the two interpreted VP anaphora trees that it generates.

13. John promised to read everything which Mary did.

Antecedent verb-elliptical verb pairs:
promise.2 - do.8, read.4 - do.8
Elliptical verb-new argument pairs:
promise.8 - preinf.3, promise.8 - read.4
Elliptical verb-new adjunct pairs: none
Interpreted VP anaphora tree1:
John(X3) promise(X1,X3,X4,u) preinf(X4)
read(X4,X3,X7,u) everything(X7)
which(X7) Mary(X9) promise(X8,X9,X4)

Interpreted VP anaphora tree2:
John(X3) promise(X1,X3,X4,u) preinf(X4)
read(X4,X3,X7,u) everything(X7)
which(X7) Mary(X9) read(X8,X9,X7)

14. John said that Mary promised to read everything which Max has.

Antecedent verb-elliptical verb pairs:
promise.5 - have.11, read.7 - have.11
Elliptical verb-new argument pairs:
have.11 - preinf.6, have.11 - read.7
Elliptical verb-new adjunct pairs: none
Interpreted VP anaphora tree1.
John(X3) say(X1,X3,X4,u,u) thatconj(X4,X9)
Mary(X10) promise(X9,X10,X11,u)
preinf(X11) read(X11,X10,X12,u)
everything(X12) which(X12) Max(X14)
have(X13,X14,X12) promise(X18,X14,X11)

Interpreted VP anaphora tree2:
John(X3) say(X1,X3,X4,u,u) thatconj(X4,X9)

Mary(X10) promise1(X9,X10,X11,u)
preinf(X11) read(X11,X10,X12,u)
everything(X12) which(X12) Max(X14)
have(X13,X14,X12) read(X18,X14,X12)

15 is a subdeletion case in which all of the arguments of the elliptical verb are filled locally within the elliptical VP. Hence, the algorithm substitutes the antecedent verb "write" for the auxiliary, and fills the direct and indirect argument slots in its frames with "notes" and "to Bill," respectively. It should be pointed out that the algorithm corrects ESG's parse of "to Bill" as a PP adjunct of "notes" in the original tree. This misparse is due to the fact that "do" does not allow an indirect object in its frame. The algorithm recognizes this adjunct as a possible filler for the indirect object slot in the frame of "write," and uses it to fill the slot when "write" is substituted for "do" in the interpreted tree.

15. Max writes more letters to Sam than Mary does notes to Bill.

Antecedent verb-elliptical verb pairs:
write.2 - do.9
Elliptical verb-new argument pairs: none
Elliptical verb-new adjunct pairs: none
Interpreted VP anaphora tree:
Max(X3) write(X1,X3,X4,X5,u)
more(X14) letter (X4) to l(X12,X5) Sam(X5)
than(X1,X7) Mary(X8) write(X7,X8,X9,X10)
note(X9) to(X9,X10) Bill(X10)

In 16, the indirect object slot of the interpreted verb "write" is filled locally by "to Bill," but "letters" is inherited from the antecedent. The algorithm also corrects the misparse of "to Bill" as a PP adjunct of the elliptical verb in the original tree by a strategy similar to the one used to correct the parse of the PP in 15

16. Max writes more letters to Sam than Mary does to Bill.

Antecedent verb-elliptical verb pairs:
write.2 - do.9
Elliptical verb-new argument pairs:
write.9 - letter.4
Elliptical verb-new adjunct pairs: none
Interpreted VP anaphora tree:
Max(X3) write(X1,X3,X4,X5,u) more(X13)
letter(X4) to1(X11,X5) Sam(X5)
than(X1,X7) Mary(X8) write(X7,X8,X4,X9)
to(X7,X9) Bill(X9)

Both the direct object "letters" and the indirect object "Sam" are inherited by the interpreted verb "write" in the fully empty ACD structure in 17.

17. Max writes more letters to Sam than Mary does.

Antecedent verb-elliptical verb pairs:
write.2 - do.9
Elliptical verb-new argument pairs:
write.9 - letter.4, write.9 - Sam.6
Elliptical verb-new adjunct pairs:
none
Interpreted VP anaphora tree:
Max(X3) write(X1,X3,X4,X5,u) more(X11)
letter(X4) to(X9,X5) Sam(X5)
than(X1,X7) Mary(X8) write(X7,X8,X4,X5)

18 and 19 show the operation of argument filtering in an ACD passive case.[13] As in 13 and 14, the algorithm corrects the misparse, in the original tree, of the second "by" phrase as a PP adjunct. It raises it to the status of the agent (deep subject) argument of the head of the new verb in the interpreted tree.

18. John was interviewed by Bill before Mary could have been by Max.

Antecedent verb-elliptical verb pairs:
interview.3 - be.10
Elliptical verb-new argument pairs: none
Elliptical verb-new adjunct pairs: none
Interpreted VP anaphora tree:
John(X3) be(X1,X3,X4) interview(X4,X13,X3)
by(X14,X13) Bill(X13) before(X1,X5)
Mary(X6) can(X5,X6,X7) have_perf(X7,X6,X8)
be(X8,X6,u) interview(X12,X10,X6)
by(X8,X10) Max(X10)

19. John was interviewed by Bill before Mary could have been.

Antecedent verb-elliptical verb pairs:
interview.3 - be.10
Elliptical verb-new argument pairs:
be.10 - Bill.5
Elliptical verb-new adjunct pairs:
Interpreted VP anaphora tree:
John(X3) be(X1,X3,X4) interview(X4,X11,X3)
by(X12,X11) Bill(X11) before(X1,X5) Mary(X6)
can(X5,X6,X7) have perf(X7,X6,X8) be(X8,X6,u)
interview(X10,X11,X6)

20 and 21 illustrate the adjunct filtering procedure of the algorithm in an intersentential case of VP anaphora. The adverbial "yesterday" is inherited in 20, but not in 21.

20. John arrived yesterday, and Mary did too.

Antecedent verb-elliptical verb pairs:
arrive.2 - do.7
Elliptical verb-new argument pairs:
none
Elliptical verb-new adjunct pairs:
arrive.7 - yesterday.3
Interpreted VP anaphora tree:
John(X9) arrive(X8,X9,u) yesterday(X11)
and(X1,X8,X13)
Mary(X14) arrive(X13,X14,u) too(X13)

21. John arrived yesterday, and Mary will tomorrow.

Antecedent verb-elliptical verb pairs:
arrive.2 - will.7
Elliptical verb-new argument pairs:
none
Elliptical verb-new adjunct pairs:
none
Interpreted VP anaphora tree:
John(X9) arrive(X8,X9,u) yesterday(X11)
and(X1,X8,X13)
Mary(X14) will(X13,X14,u) arrive(X19,X14,u)
tomorrow(X17)

22 and 23 exhibit the effects of adjunct filtering of a more complex variety in an ACD structure. Both the adverb "briefly" and the PP adjunct "with Bill" are inherited by the interpreted verb "walked" in 22, but only "briefly" is inherited in 23.

22. John walked briefly with Bill before Mary did.

Antecedent verb-elliptical verb pairs:
walk.2 - do1.8
Elliptical verb-new argument pairs:
none
Elliptical verb-new adjunct pairs:
walk.8 - briefly.3, walk.8 - with.4
Interpreted VP anaphora tree.
John(X3) walk(X1,X3,u,u) briefly(X1)
with(X1,X10) Bill(X10)
before(X1,X6) Mary(X7) walk(X6,X7,u)

23. John walked briefly with Bill before Mary did with Max.

Antecedent verb-elliptical verb pairs:
walk.2 - do1.8
Elliptical verb-new argument pairs:
none
Elliptical verb-new adjunct pairs:
walk.8 - briefly.3
Interpreted VP anaphora tree.
John(X3) walk(X1,X3,u,u) briefly(X1)
with(X1,X12) Bill(X12)
before(X1,X6) Mary(X7) walk(X6,X7,u)
with(X6,X10) Max(X10)

4 A Syntactic Filter on Pronominal Anaphora

The filter consists of six conditions for NP-pronoun noncoreference within a sentence. To state these conditions, we use the following terminology. The *agreement features* of an NP are its number, person, and gender features. We will say that a phrase P is in the *argument domain* of a phrase N iff P and N are both arguments of the same head. We will say that P is in the *adjunct domain* of N iff N is an argument of a head H, P is the object of a preposition PREP, and PREP is an adjunct of H. P is the *NP domain* of N iff N is the determiner of a noun Q and (i) P is an argument of Q, or (ii) P is the object of a preposition PREP and PREP is an adjunct of Q.

4.1 Filter on Pronominal Anaphora

A pronoun P is noncoreferential with a (nonreflexive or nonreciprocal) noun phrase N if any of the following conditions hold.

I. P and N have incompatible agreement features.
II. P is in the argument domain of N.
III. P is in the adjunct domain of N.
IV. P is an argument of a head H, N is not a pronoun, and N is contained in H.
V. P is in the NP domain of N.
VI. P is a determiner of a noun Q, and N is contained in Q.

Condition I rules out coreference between a pronoun and an NP with incompatible agreement features. It will identify the co-indexed expressions in 24a–c as noncoreferential.

24a. *He$_i$ said that they$_i$ came.
b. *The woman$_i$ said that he$_i$ is funny.
c. *I$_i$ believe that she$_i$ is competent.

The filter treats ⟨"he," "they"⟩ as a noncoreferring pair, which entails only that the intended denotation of "he" cannot be taken as identical to that of "they." The referent of "he" can, of course, be a part of the referent of "they," and, in appropriate contexts, a discourse interpretation system, like the **LODUS** system of Bernth (1988, 1989), should be able to recognize this possibility.

Condition II covers cases in which a pronoun and an NP are arguments of the same head, and so it rules out coreference between the coindexed expressions in 25a–d.

25a. *Mary$_i$ likes her$_i$.
b. *She$_i$ likes her$_i$.
c. *John$_i$ seems to want to see him$_i$.
d. *This is the girl$_i$ Mary said she$_i$ saw.

It is important to note that the conditions of the filter apply to pronouns and NPs regardless of whether they are lexically realized in argument position (25a–b), or bind the argument slots which they fill in their heads at a distance through control (25c) and unbounded dependency relations (25d). This is due to the fact that the variable that fills an argument slot is unified with the phrase marker of the head of the phrase to which it corresponds. Therefore, it is not necessary to incorporate empty categories such as traces and PRO into the parse output, and compute appropriate binding chains for these categories in order for the algorithm to handle noncoreference in cases involving control and wh-movement. Mechanisms of this kind are required for implementations of Chomsky's binding theory in Government Binding–based parsers, such as those described in Correa (1988) and Ingria and Stallard (1989).

Condition III rules out coreference between an argument of a verb V and the object of a prepositional adjunct of V, as in 26a–b.[14]

26a. *Mary$_i$ arrived with her$_i$.
b. *Who$_i$ did John say wants to sit near him$_i$?

Condition IV prevents coreference between a pronoun that is an argument of a head H, and a nonpronominal NP contained in H, as in 27a–c.

27a. *Who$_i$ did she$_i$ say Mary$_i$ kissed?
b. *This is the man$_i$ he$_i$ said Max$_i$ wrote about.
c. *He$_i$ likes John$_i$'s mother.

The filter does permit coreference in 28a–b. "His" in 28a is not an argument of "likes," and so ⟨"his," "John"⟩ do not satisfy Condition IV (or any other condition of the filter). An ordering constraint attached to Condition IV requires that a possessive adjunct of a noun contained in a head H follow a pronominal argument of H for this condition to apply to the pair.

28a. His$_i$ mother likes John$_i$.
b. John$_i$'s mother likes him$_i$.

Finally, V and VI in effect apply conditions II and III, respectively, to NP internal cases. They prevent coreference in 29a–c, while allowing it in 29d.

29a. *His$_i$ portrait of John$_i$ is interesting.
b. *John$_i$'s portrait of him$_i$ is interesting.
c. *His$_i$ description of the portrait by John$_i$ is interesting.
d. John$_i$'s description of the portrait by him$_i$ is interesting.

The filter on pronominal anaphora restricts the search space that a discourse system of anaphora resolution must consider. Bernth has integrated the filter into **LODUS** (Bernth 1988, 1989), which resolves pronominal anaphora and NP denotation through semantic and pragmatic rules of inference. The anaphora resolution component of **LODUS** applies only to the pronoun-NP pairs that the syntactic filter has not identified as noncoreferential.

An example of the filter algorithm's output is given in 30 in Figure 4. The list of noncoreferential pronoun-NP pairs appears after the parse tree. Ulrike Schwall has successfully implemented the algorithm in German Slot Grammar

30. This is the girl who she wanted Mary to talk to.

```
subj     this(X3)                noun(pron(defprn),nom.sg,nwh)
top      be(X1,X3,X4)            verb(fin(pers3,sg,pres,X2))
ndet     the(X7)                 det(sg,def)
predcmp  girl(X4)                noun(cn,X5.sg,X6)
objprep  who(X4)                 noun(pron(wh),X13.sg,wh)
subj     she(X9)                 noun(pron(defprn),nom.sg,nwh)
nrel     want(X8,X9,X10,X11)     verb(fin(pers3,sg,past,dep:dcl:wh))
obj      Mary(X10)               noun(prop,acc.sg,nwh)
preinf   preinf(X11)             preinf
comp     talk(X11,X10,X4,u)      verb(inf(full))
pobj     to(X16,X4)              prep(to,X17,e(X4))
```

Non-coreferential pronoun-NP pairs:

she.6 - girl.4, she.6 - who.5, she.6 - Mary.8

31. Paul gab Peter das Buch, um ihn zu beeindrucken.
(Paul gave Peter the book to impress him.)

```
subj     Paul(X2)                noun(prop,nom.sg.m.nda.X6,nwh)
top      geb(X1,X2,X3,X4,u)      verb(fin(pers3,sg,past,ind:dcl:nwh))
iobj     Peter(X3)               noun(prop,dat.sg.X14.nda.X15,nwh)
ndet     d(X13)                  det(naNS,def)
obj      buch(X4)                noun(cn,acc.sg.nt.na.nrflx,nwh)
sep      ,                       separator
preumzu  preumzu(X7)             preumzu
obj      er(X8)                  noun(pron(pers3),acc.sg.m.kda.X12,nwh)
preinf   preinf(X7)              preinf
umzu     beeindruck(X7,X2,X8,u)  verb(inf(ufull,a))
```

Non-coreferential pronoun-NP pairs: er.8 - Paul.1

Figure 4

(GSG). An example of its output for GSG is given in 31 in Figure 4.

5 An Anaphor-Binding Algorithm

We take the set of anaphors to include reflexive pronouns and the reciprocal NP "each other." The notion *higher argument slot* used in the formulation of the algorithm is defined by the hierarchy of argument slots given in 32.

32. subj > agent > obj > (iobj | pobj)

subj is the surface subject slot, *agent* is the deep subject slot of a verb heading a passive VP, *obj* is the direct object slot, *iobj* is the indirect object slot, and *pobj* is the object of a PP complement of a verb, as in "put NP on *NP*." We assume the definitions of *argument domain, adjunct domain,* and *NP domain* given in Section 4.

5.1 ANAPHOR-BINDING ALGORITHM

A noun phrase N is a possible antecedent binder for an anaphor A iff N and A do not have incompatible agreement features, and one of the following five conditions holds.

I. A is in the argument domain of N, and N fills a higher argument slot than A.
II. A is in the adjunct domain of N.
III. A is in the NP domain of N.
IV. N is an argument of a verb V, there is an NP Q in the argument domain or the adjunct domain of N such that Q has no noun determiner, and A is (i) an argument of Q, or (ii) A is an argument of a preposition PREP and PREP is an adjunct of Q.
V. A is a determiner of a noun Q, and (i) Q is in the argument domain of N and N fills a higher argument slot than Q, or (ii) Q is in the adjunct domain of N.

Conditions I and II cover anaphoric binding in cases like 33–34, respectively.

33a. They$_i$ wanted to see themselves$_i$.
b. Mary knows the people$_i$ who John introduced to each other$_i$.
34a. He$_i$ worked by himself$_i$.
b. Which friends$_i$ plan to travel with each other$_i$?

Condition III handles binding of an anaphor inside an NP by the determiner of the NP, as in 35, and IV deals with NP internal anaphors which are bound from outside of the NP, as in 36.

35. John liked Bill$_i$'s portrait of himself$_i$.
36. They$_i$ told stories about themselves$_i$.

Condition V applies to cases in which a reciprocal determiner is bound by an argument in the same clause as the NP containing the reciprocal. 37 is an example of this binding relation, and 38 illustrates the combined effect of IV and V.

37. [John and Mary]$_i$ like each other$_i$'s portraits.

38. [John and Mary]$_i$ like each other$_i$'s portraits of themselves$_i$.

An example of the anaphor binding algorithm's output is presented in 39 in Figure 5. Notice that the sentence in this example is ambiguous concerning antecedents for "himself," and the algorithm correctly identifies both "who" and "John" as possible binders of the reflexive. When a discourse interpretation system makes use of this algorithm, it must, of course, constrain the interpetation of anaphors by requiring that exactly one binding pair be selected from the list of pairs that the algorithm provides for any given anaphor, relative to the clause in which it appears.

Ulrike Schwall has implemented the algorithm in GSG, and 40 in Figure 5 illustrates its output for a German sentence.

6 An Integrated System for Anaphora Resolution

Any two of the algorithms described in Sections 4–6 can operate in conjunction with each other. Examples of the results provided by such combinations are given in 41–44 (see Figure 6 for 41). In 45, both the filter and anaphor binding algorithms have been integrated into the VP anaphora algorithm, and operate on the interpreted VP anaphora tree it generates.

6.1 VP ANAPHORA ALGORITHM WITH PRONOMINAL ANAPHORA FILTER

42. John talked to him, and Bill did too.

```
Antecedent verb-elliptical verb pairs:
   talk.2 - do1.8
Elliptical verb-new argument pairs:
   talk.8 - he.4
Elliptical verb-new adjunct pairs:
   none
Interpreted VP anaphora tree:
   John(X9) talk(X8,X9,X10,u) to(X12,X10)
   he(X10) and(X1,X8,X14)
   Bill(X15) talk(X14,X15,u,X10) too2(X14)

Noncoreferential pronoun-NP pairs:
   he.4 - John.1, he.4 - Bill.7
```

43. Mary sent John to everyone who he did.

```
Antecedent verb-elliptical verb pairs:
   send.2 - do2.8
Elliptical verb-new argument pairs:
   send.8 - John.3
Elliptical verb-new adjunct pairs:
   none
Interpreted VP anaphora tree:
   Mary(X3) send(X1,X3,X4,X5) John(X4)
   to(X6,X5) everyone(X5) who(X5)
   he(X9) send(X8,X9,X4,X5)
```

39. Who did John talk to about himself?

```
objprep who(X2)           noun(pron(wh),X7.sg,wh)
top     do(X1,X3,X4)      verb(fin(pers3,sg,past,ind]:wh))
subj    John(X3)          noun(prop,nom.sg,nwh)
auxcmp  talk(X4,X3,X2,u)  verb(inf(bare))
pobj    to(X11,X2)        prep(to,X12,e(X2))
vprep   about(X4,X10)     prep(about,X9,e(X10))
objprep himself(X10)      noun(pron(reflprn),acc.sg,X9)
```

Antecedent NP-Anaphor Binding Pairs: who.1 - himself.7, John.3 - himself.7

40. Dies ist der Mann, der ueber sich sprechen soll.
(This is the man who should speak about himself.)

```
subj    dies(X2)          noun(pron(defprn),nom.sg.X5.na.X6,nwh)
top     sei(X1,X2,X3,u)   verb(fin(pers3,sg,pres,ind:dcl:nwh))
ndet    d(X8)             det(nMgdFSgP,def)
predcmp mann(X3)          noun(cn,nom.sg.m.w.X7,nwh)
sep     ,                 separator
subj    d(X3)             relpro
pobj    ueber(X16,X15)    prep(uber.acc,nwh,e(X15))
objprep reflex(X15)       noun(pron(reflprn),acc.sg.X17.X18.X19,n wh)
auxcmp  sprech(X11,X3,u,X15) verb(inf(bare,a))
nrel    soll(X9,X3,X11)   verb(fin(pers3,sg,pres,dep:dcl:wh))
```

Antecedent NP-reflexive pairs: mann.4 - reflex.8

Figure 5

Noncoreferential pronoun-NP pairs:
he.7 - Mary.1, he.7 - John.3,
he.7 - everyone.5, he.7 - who.6

6.2 VP ANAPHORA ALGORITHM WITH ANAPHOR BINDING ALGORITHM

44. The girl will write a book about herself, and Mary might too.

Antecedent verb-elliptical verb pairs:
write.4 - may.12
Elliptical verb-new argument pairs:
may.12 - book.6, may.12 - about.7
Elliptical verb-new adjunct pairs:
none
Interpreted VP anaphora tree:
the(X11) girl(X9) will(X8,X9,X10)
write(X10,X9,X12,u,u) a(X15) book(X12)
about(X12,X16) herself(X16)
and(X1,X8,X18) Mary1(X19) may1(X18,X19,u)
write(X24,X19,X12) too(X18)

Antecedent NP-reflexive pairs:
girl.2 - herself.8, Mary.11 - herself.8

6.3 VP ANAPHORA ALGORITHM WITH THE PRONOMINAL ANAPHORA FILTER AND ANAPHOR BINDING ALGORITHM

45. They discussed each other's portraits of themselves before John and Mary did.

Antecedent verb-elliptical verb pairs:
discuss.2 - do1.12
Elliptical verb-new argument pairs:
discuss.12 - portrait.5
Elliptical verb-new adjunct pairs:
none
Interpreted VP anaphora tree:
they(X3) discuss(X1,X3,X4)
each.other(X10) 's portrait(X4,X9)
of(X11,X9) themselves(X9)
before(X1,X5) John(X7) and(X6,X7,X8)
Mary(X8) discuss(X5,X6,X4)

Noncoreferential pronoun-NP pairs:
they.1 - portrait.5, they.1 - John.9,
they.1 - coord(and,John,Mary).10,
they.1 - Mary.11

Antecedent NP-anaphor pairs:
they.1 - (each.other).3,
(each.other).3 - themselves.7,
coord(and,John,Mary).10 - (each.other).3

Our integrated system for anaphora resolution is syntactically based, and it must be supplemented by additional semantic procedures to yield fully adequate interpretations of elliptical VP structures. This can be seen quite clearly by considering the interpreted VP anaphora tree of 42. Here "him," the indirect object of "talk," is inherited by the interpreted verb in the second conjunct, and its marker

```
Pronominal Anaphora Filter with Anaphor Binding Algorithm

41. This is the man who he asked to talk about himself.

          subj     this(X3)              noun(pron(defprn),nom.sg,nwh)
          top      be(X1,X3,X4)          verb(fin(pers3,sg,pres,X2))
          ndet     the(X7)               det(sg,def)
          predcmp  man(X4)               noun(cn,X5.sg,X6)
          obj      who(X4)               noun(pron(wh),X11.X12,wh)
          subj     he(X9)                noun(pron(defprn),nom.sg,nwh)
          nrel     ask(X8,X9,X4,X10)     verb(fin(pers3,sg,past,dep:dcl:wh))
          preinf   preinf(X10)           preinf
          comp     talk(X10,X4,u,X15)    verb(inf(full))
          pobj     about(X16,X15)        prep(about,X17,e(X15))
          objprep  himself(X15)          noun(pron(reflprn),acc.X18,X17)

Non-coreferential pronoun-NP pairs:  he.6 - man.4, he.6 - who.5

Antecedent NP-reflexive pairs:  man.4 - himself.11, who.5 - himself.11
```

Figure 6

variable X10 is unified with the marker of the indirect object slot in the argument frame of this verb. Therefore, the interpreted VP anaphora tree correctly represents the fact that the second conjunct in 42 must be understood as asserting that Bill spoke to the same person as John did. The list of noncoreferential pronoun-NP pairs specifies that "him" is distinct in reference from both "John" and "Bill." However, in its present form, the VP anaphora algorithm unifies the marker variables of all inherited arguments with the appropriate slots in the frame of an interpreted verb. This will yield incorrect results for a sentence like 46, where "a book" is the inherited argument.

46. John read a book, and Mary did too.

On at least one possible reading of the sentence, John and Mary read distinct books. To complete the interpretation of elliptical VPs, it will be necessary to add procedures for substituting new marker variables for the occurrence of inherited arguments and adjuncts in the interpreted VP, when these expressions need not be taken as having the same denotations that they receive as arguments and adjuncts of the antecedent verb.

A related problem concerns scope assignment in sentences like 47.

47. Mary spoke to everyone after Max did.

Dalrymple, Shieber, and Pereira (1990) point out that 47 is ambiguous between a narrow scope reading on which Mary spoke to everyone after Max spoke to everyone, and a wide scope reading according to which everyone is such that Mary spoke to him/her after Max spoke to him/her. At this point, the VP anaphora algorithm generates only the former reading, as "everyone" is inherited as an argument by the interpreted head of the ellided VP.

We could capture the wide scope reading by modifying our S-structure copying analysis of VP anaphora to allow copying to apply to more abstract semantic representations. This approach involves adopting an interpolated copying theory of VP ellipses on which copying is permitted not only at the level of S-structure, but also after the antecedent clause has been assigned a partial or full semantic interpretation. In the case of 47, copying could apply after "everyone" has been assigned scope through the operation of NP storage and a semantic variable appears in its place.[15] The result of such copying would be an interpretation on which "everyone" would have wide scope by virtue of the fact that it binds variables in both the antecedent and interpreted VPs. The interpolated copying analysis could be implemented within Slot Grammar by permitting either expressions or simply their marker variables to be inherited. The former case corresponds to S-structure copying of a constituent, the latter to copying at a level of representation to which interpretation has already applied. If "everyone" is inherited in 47, it is, in effect, copied, and the narrow scope reading of the sentence results. When only its marker variable is inherited, the semantic variable within its scope is copied, which yields the wide scope interpretation.[16]

43 is particularly interesting. The sentence is a variant of an example that May (in press) claims provides evidence for this QR treatment of ACD structures. He maintains that only after QR has been applied to "everyone who he did" and the matrix VP "sent John to t" copied into the empty VP in the relative clause, can Principle C of Chomsky's binding theory rule out coreference between "he" and "John." In fact, the application of our filter to the interpreted VP anaphora tree provides the correct results for this case. This is due to the fact that the VP algorithm identifies "John" as the inherited object of the verb that it

substitutes for the elliptical verb in the new tree. Condition II of the filter algorithm is then satisfied by ⟨"he," "John"⟩.[17] This example provides strong support for our treatment of VP anaphora.

In the interpreted VP anaphora tree of 44, the substituted verb "writes" inherits "herself" as a new argument, and so we capture the "sloppy" reading of this sentence, on which each occurrence of the reflexive is bound by the subject of the clause in which it occurs. To obtain the "strict" interpretation, according to which "herself" is bound only by "the girl," it will be necessary to allow copying of the marker variable associated with "herself," in the manner required for the wide scope reading of "everyone" in 47 (see footnote 16).

The fully integrated algorithm provides the desired results for the sentence in 45. "They" is identified as noncoreferential with any of the NPs contained in the head of which it is the subject. "They" binds "each other" in the matrix clause, and "John and Mary" binds "each other" as the determiner of the inherited argument "portrait" in the adverbial phrase. "Each other" binds "themselves," and so, by transitivity of binding, "they" binds "themselves" in its occurrence in the object NP headed by "portrait" in the matrix clause, and "John and Mary" binds "themselves" in its occurrence in this NP as the inherited object of the substituted verb in the adverbial phrase.

7 Conclusion

We have presented three implemented algorithms for anaphora resolution in Slot Grammar. The conjunction of the two NP anaphora rules covers approximately the same phenomena as Chomsky's binding theory, but these rules do not require empty categories and the definition of binding chains in parse output. The VP anaphora algorithm implements an S-structure analysis of VP ellipsis that we argue offers a more unified and principled explanation of different sorts of VP anaphora than recent LF-based accounts. The success of the algorithm in providing appropriate representations for the VP anaphora cases that we used to motivate our theoretical approach supports this view. The combination of the three algorithms constitutes a powerful syntactically driven system for anaphora resolution in Slot Grammar. This system reduces the burden on modules of semantic and discourse interpretation by supplying partially interpreted representations to which the rules of the latter can apply.

References

Bernth, A. (1989). "Discourse understanding in logic." In *Proceedings of the North American Conference on Logic Programming.* MIT Press. 755–771.

Bernth, A. (1988). *Computational discourse semantics.* Doctoral dissertation, University of Copenhagen and IBM Research.

Bresnan, J. (1975). "Comparative deletion and constraints on transformations." *Linguistic Analysis,* 1: 25–74.

Byrd, R. J. (1983). "Word formation in natural language processing systems." *Proceedings of the 8th International Joint Conference on Artificial Intelligence,* 704–706.

Correa, N. (1988). "A binding rule for government-binding parsing." *COLING '88,* Budapest: 123–129.

Chomsky, N. (1986a). *Knowledge of Language: Its Nature, Origin, and Use.* Praeger.

Chomsky, N. (1986b). *Barriers.* MIT Press.

Chomsky, N. (1981). *Lectures on Government and Binding.* Foris.

Chomsky, N. (1977). "On wh-movement." In *Formal Syntax,* P. Culicover, T. Wasow, and A. Akmajian, edited by Academic Press.

Dalrymple, M.; Shieber, S; and Pereira, F. (1990). "Ellipsis and higher-order unification." SRI, Harvard University, and AT&T Bell Laboratories, unpublished ms.

Gazdar, G.; Klein, E.; Pullum, G.; and Sag, I. (1985). *Generalized Phrase Structure Grammar.* Blackwell.

Haik, I. (1987). "Bound VP's that need to be." *Linguistics and Philosophy* 11: 503–530.

Heidorn, G. E. (1982). "Experience with an easily computed metric for ranking alternative parses." *Proceedings of the 20th Annual Meeting of the Association of Computational Linguistics,* 1982: 82–84.

Hirshbuhler. (1982). "VP deletion and across-the-board quantifier scope." *Proceedings of NELS 12,* edited by J. Pustejovsky and P. Sells. GLAS.

Klavans, J. L. and Wacholder, N. (1989). "Documentation of features and attributes in UDICT." Research Report RC14251, IBM T. J. Watson Research Center.

Ingria, R. and Stallard, D. (1989). "A computational mechanism for pronominal reference." In *Proceedings of the 27th Annual Meeting of the Association of Computational Linguistics,* Vancouver: 262–271.

Jensen, K. (1986). "PEG: A broad-coverage computational syntax of English." Technical Report, IBM T. J. Watson Research Center.

Jensen, K. and Heidorn, G. (1990). "Post-syntactic processing of arguments and anaphora." Technical Report, IBM.

Lappin, S. (1984). "VP anaphora, quantifier scope, and logical form." *Linguistic Analysis* 13: 273–315.

Lappin, S. In press. "Concepts of logical form in linguistics and philosophy." In *The Chomskyian Turn,* edited by A. Kasher. Blackwell.

Lappin, S. and McCord, M. (1990). "A syntactic filter on pronominal anaphora in slot grammar." In *Proceedings, 28th Annual Meeting of the Association for Computational Linguistics:* 135–142.

Lappin, S.; Golan, I.; and Rimon, M. (1989). "Computing Grammatical Functions from Configurational Parse Trees." Technical report 88.268, IBM Scientific Center, Haifa.

Larson, R. (1988). "Scope and comparatives." *Linguistics and Philosophy,* 11: 1–26.

Larson, R. (1987). "Missing prepositions and the analysis of English free relatives." *Linguistic Inquiry,* 13: 273–315.

McCord, M. C. (1990). "SLOT GRAMMAR: A system for simpler construction of practical natural language grammars." To appear in *Natural Language and Logic, International Scientific Symposium,* edited by R. Studer, Lecture Notes in Computer Science, Springer Verlag, 118–145.

McCord, M. C. (1989a). "Design of **LMT:** A prolog-based machine translation system." *Computational Linguistics,* 15: 33–52.

McCord, M. C. (1989b). "A new version of slot grammar." Research report RC 14506, IBM Research Division, Yorktown Heights, NY 10598.

McCord, M. C. (1989c). "A new version of the machine translation system **LMT.**" *Journal of Literary and Linguistic Computing,* 4: 218–229.

McCord, M. C. (1989d). "**LMT.**" In *Proceedings of the MT Summit II,* Deutsche Gesellschaft für Dokumentation, Frankfurt, 94–99.

McCord, M. C. (1988). "A multi-target machine translation system." In *Proceedings, International Conference on Fifth Generation Computer Systems,* 1988, Institute for New Generation Computer Technology, Tokyo, Japan, 1141–1149.

McCord, M. C. (1987). "Natural language processing in prolog." In

Knowledge Systems and Prolog: A Logical Approach to Expert Systems and Natural Language Processing, edited by A. Walker, M. C. McCord, J. F. Sowa, and W. G. Wilson, Addison-Wesley.

McCord, M. C. (1986). "Design of a prolog-based machine translation system." In *Proceedings of the Third International Logic Programming Conference,* Springer-Verlag, Berlin, 350–374.

McCord, M. C. (1985). "Modular logic grammars." In *Proceedings, 23rd Annual Meeting of the Association for Computational Linguistics,* 104–117.

McCord, M. C. (1984). "Semantic interpretation for the EPISTLE system." In *Proceedings, Second International Logic Programming Conference,* Uppsala, Sweden, 65–76.

McCord, M. C. (1982). "Using slots and modifiers in logic grammars for natural language." *Artificial Intelligence,* 18: 327–367.

McCord, M. C. (1980). "Slot grammars." *Computational Linguistics,* 6: 31–43.

May, R. In press. "Syntax, semantics, and logical form." In *The Chomskyian Turn,* edited by A. Kasher. Blackwell.

May, R. (1985). *Logical Form: Its Structure and Derivation.* The MIT Press, Cambridge, MA.

Reinhart, T. (1984). *Anaphora.* Croom Helm.

Reinhart, T. (1981). "Definite NP anaphora and c-command domains." *Linguistic Inquiry,* 12: 605–635.

Reinhart, T. (1976). *The syntactic domain of anaphora.* Doctoral dissertation, MIT, Cambridge, MA.

Sag, I. (1976). *Deletion and logical form.* Doctoral dissertation, MIT, Cambridge, MA.

Walker, A.; McCord, M. C.; Sowa, J. F.; and Wilson, W. G. (1987). *Knowledge Systems and Prolog: A Logical Approach to Expert Systems and Natural Language Processing.* Addison-Wesley.

Wasow, T. (1972). *Anaphora in generative grammar.* Doctoral dissertation, MIT, Cambridge, MA.

Webber, B. (1978). *A formal approach to discourse anaphora.* Doctoral dissertation, Harvard University, Cambridge, MA.

Williams, E. (1977). "Discourse and logical form." *Linguistic Inquiry,* 8: 107–139.

Notes

1. Earlier versions of the paper were presented to the SRI natural language group at Menlo Park, CA in June, 1990, and to the AT&T Bell Laboratories natural language and speech generation group at Murray Hill, NJ in July, 1990. We are grateful to the participants of these two forums for their comments. We thank Mori Rimon for detailed and useful comments on an earlier version of the paper. We also very much appreciate the careful reading of the paper and the suggestions of three anonymous referees. We would particularly like to express our thanks to Fernando Pereira and Mary Dalrymple for extended discussion of this paper and the problems involved in VP anaphora resolution. Their own work in this area has provided us with considerable stimulation and insight.
2. See Reinhart (1976, 1981, 1984), and Chomsky (1981, 1986b) for alternative definitions of *c-command,* and discussions of the role of c-command in determining the possibilities for anaphora. See Lappin and McCord (1990) for comparisons between the pronominal anaphora filter in Slot Grammar and recent implementations of Chomsky's binding theory in GB-based parsers.
3. Shortly after we designed and implemented these three algorithms in Slot Grammar, Karen Jensen constructed three alternative procedures for anaphora resolution in the PEG grammar (see Jensen 1986 for a general description of PEG). Moreover, George Heidorn has implemented a version of our filter on pronominal anaphora in PEG. Jensen's procedures and Heidorn's implementation of our filter algorithm rely on and apply after a set of second-pass operations that comprise a module referred to as PEGASUS. This module computes deep grammatical roles from the surface configurational structures constituting the PEG parse. (See Jensen and Heidorn 1990 for a brief description of PEGASUS and an outline of Jensen's anaphora resolution procedures.) By contrast, in Slot Grammar deep grammatical roles are obtained directly in the course of parsing, through the unification of complement "marker variables" with variables in the argument frames of their heads. While PEGASUS reconstructs deep grammatical role information (primarily) from surface configurational relations, the representation of these roles in Slot Grammar is lexically driven and is an integral part of the parsing process. Therefore, where Jensen's anaphora resolution procedures operate on the output of a second-pass module (they are, in effect, third-pass rules), our algorithms are formulated in terms of the head-complement structures provided directly by the Slot Grammar parser. See McCord (1984) and Lappin et al. (1989) for earlier systems that compute deep grammatical roles from PEG's surface parse structures.
4. The list of complement slots in the argument frame of a verb includes its subject. Therefore, SG represents argument structure in a manner analogous to that of LFG in that it makes no structural distinction between the subject as an external argument of a VP and the internal arguments of the verb, as does Government Binding theory.
5. The distinction between slot filler rules and ordering constraints parallels the difference between immediate dominance rules and linear precedence rules in GPSG. See Gazdar et al. (1985) for a characterization of ID and LP rules in GPSG. See McCord (1989b) for more discussion of the relation of Slot Grammar to other systems.
6. IP is an inflectional phrase, the category to which sentences correspond in current versions of X′ theory. See Chomsky (1986b) for details of the IP analysis of sentences.
7. May's QR-based analysis of VP anaphora extends several of the ideas concerning the interaction of quantified NPs and VP anaphora originally proposed in Sag (1976). Webber (1978) adopts and modifies Sag's approach to VP anaphora within a computationally oriented framework. See Lappin (1984) for discussion of some of the difficulties that arise with Sag's original analysis. Lappin (in press) presents more detailed criticism of May's account, and of a variant of this analysis proposed in Haik (1987). This paper also deals with Larson's (1987, 1988) extension of May's account in ACD structures in adverbial phrases. Other treatments of VP anaphora are discussed, and motivation is given for the S-structure interpretation view adopted here. In the following, we limit ourselves to a brief presentation of two main arguments against the LF approach to VP anaphora resolution, and a summary of the S-structure alternative that we propose.
8. See Chomsky 1981 and 1986b for formulations of subjacency and arguments to the effect that it is an S-structure constraint. Haik presents an LF analysis of VP anaphora that classifies an empty antecedent-contained VP as a variable bound by a wh-(or empty) operator at S-structure. While Haik's account permits subjacency to constrain ACD structures, it requires that the VP variable be reanalyzed as an NP trace at LF in order to obtain a structure like 3. This is an ad hoc and otherwise unmotivated device. See Lappin (in press) for more detailed discussion of Haik's proposal.
9. See Bresnan (1975) and Chomsky (1977) for the classical discussion of subdeletion.
10. The idea that empty VPs are structured was initially proposed in Wasow (1972) and adopted in Williams (1977).
11. See Lappin (1984) and the references cited there for discussions of intersentential VP anaphora.
12. The VP anaphora algorithm identifies an elided VP by the presence of a bare auxiliary or infinitival complementizer. Therefore, it will not deal with elided VPs that are not introduced by auxiliaries or the complementizer "to," as in (i)a–b.

    (i)a. John wrote more papers than Mary.
    b. Bill arrived before Lucy.

    Complex syntactic and semantic factors must be invoked to distin-

guish "bare" VP ellipsis cases of this sort from structurally similar sentences that do not contain elided VPs, such as (ii)a–b.

(ii)a. John gave more papers than books to Mary.
b. Bill arrived before the beginning of the concert.

Extending the algorithm to cover "bare" VP ellipsis is clearly a nontrivial task, which is beyond the scope of our current work. We hope to take up this matter in future research.

13. Unlike the examples where the elliptical verb is identified by the auxiliary "do," the elliptical verb in 19 is represented by the auxiliary "be" rather than the antecedent verb in the list of elliptical verb–new argument pairs. This is due to the fact that with auxiliaries other than "do" the algorithm copies the elliptical verb immediately after the auxiliary in the interpreted tree, while in the case of "do" the antecedent is substituted for the auxiliary. The filtering component of the algorithm applies after the antecedent has been inserted into the new tree, and so it identifies the antecedent with the elliptical verb for purposes of listing new arguments and adjuncts when the antecedent replaces the auxiliary, but not otherwise.
14. Unfortunately, Condition III also incorrectly blocks coreference in cases like (i)a–b, (discussed in, for example, Reinhart (1981)) when "near him" is taken as an adverb modifying the head verb "saw."

(i)a. Dan_i saw a snake near him_i.
b. Near him_i, Dan_i saw a snake.

The problem with the cases of this kind is that the possibilities for pronominal coreference are notoriously susceptible to lexical variation, as indicated by (ii)–(iii).

(ii)a. John_i took a book with him_i.
b. *John_i took a walk with him_i.
(iii)a. Mary_i heard music near her_i.
b. ??Mary_i played music near her_i.

A variety of syntactic and lexical semantic factors seem to be involved in determining the possibility of coreference in these cases. However, when no complement intervenes between the subject of a verb and the pronominal object of its PP adjunct, coreference is always excluded. In light of this fact and the lexically governed complexity of the coreference patterns in structures like (i)–(iii), we have decided to retain Condition III in its present general form. Clearly, it would be desirable to refine it to allow for the distinctions illustrated in the examples given here.

15. Dalrymple, Shieber, and Pereira (1990) obtain both scope readings for 47 by permitting interaction between storage and an equational procedure for ellipsis resolution. This procedure involves solving an equation in which the interpretation of the source clause appears on one side and a predication containing a higher order property variable corresponding to the ellided VP is on the other. The representation of the source clause in the equation can contain either a released or an unreleased stored NP interpretation. Lappin (1984) employs an analogous free interaction between NP storage and interpretation to derive both the globally wide and locally (conjunct) wide readings of "many windows" in (i). Sentences of this kind were originally discussed in Hirshbuhler (1982).

(i) A Canadian flag was hanging in front of many windows, and an American flag was too.

16. It will be necessary to constrain copying at all levels by the filter on pronominal anaphora, as the possibilities for pronominal coreference in ellided VPs are restricted by the conditions that the filter implements. Thus, even if only the marker variable of a pronoun is inherited by an interpreted verb, it will be identified as the (denotation) marker variable of a pronoun, and subject to the pronominal coreference filter within the VP that the interpreted verb heads. Similarly, copying of reciprocal NPs must be restricted by the anaphor binding algorithm at every level of copying. The situation with respect to reflexives is less clear, given the possibility of strict as well as sloppy readings for reflexives in the interpretation of anaphoric VPs.
17. The variable of the wh-phrase (which unifies with that of the head noun of the relative clause) fills the object slot in the frame of the elliptical (auxiliary) verb in the original tree. It is correctly reanalyzed as filling the indirect object slot of the substituted verb.