MICHAEL T. WESCOAT

A PROCEDURAL PERSPECTIVE ON THE INTERPRETATION OF ELLIPSIS AND PRONOUNS*

The present paper discusses the semantic problems posed by ELLIPTICAL structures, such as the example in (1), and their effect on pronoun interpretation.

(1) John admires his mother. Bill too.

The general characteristics of the phenomenon are that *(a)* one clause, called the TARGET, appears in an abbreviated form and *(b)* the interpretation of the target clause is dependent on the accompanying, fully expressed clause, called the SOURCE, to fill in any elements of content left unspecified in the target. The syntactic manifestations of the phenomenon of ellipsis are quite varied; the following list is only partial:

(2) John admires his mother. Bill does too.

(3) John admires his mother. So does Bill.

(4) John admires his mother, and Bill his father.

In examples (1)–(3) the source and target clauses could alternatively have been coordinated. The construction exhibited in the target clause of (1) is what Kempson (1992) calls a BARE ARGUMENT FRAGMENT.[1] Examples (2) ad (3) are variants of the VP ELLIPSIS phenomenon, and (4) is an instance of GAPPING. The above examples display a variety of interesting ambiguities. As an example, let us go back to (1); we shall consider only those readings where the pronoun *his* is interpreted as coreferent with *John* in the source clause.

(5) John$_i$ admires his$_i$ mother. Bill too. [=(1)]

There are at least three readings for the target clause:

(6) READINGS FOR THE TARGET IN (5)

a. 'Bill admires John's mother.'

b. 'Bill admires his own mother.'

c. 'John admires Bill.'

Readings (6a,b) are distinguishable from that in (6c) on the basis of the fact that in the former cases the target-clause argument *Bill* is taken as corresponding to the source-clause subject, whereas it is correlated with the source-clause object in the latter case. I shall use expressions like the previous ones with 'correspond' and 'correlate' to describe the relation that holds between elements of the target and source clauses when the semantic representation of the source-clause element is lacking from the interpretation of the target clause, having been supplanted by the semantic representation of the target-clause element. The difference between (6a) and (6b) lies in the interpretation of the pronoun *his* under ellipsis. The reading in (6a) maintains the original referent

*The present work is a somewhat modified version of a paper presented at the 41st Machikaneyama Kotoba no Kai at Osaka University on January 14, 1993. I am grateful to all those who were present on that occasion, and I offer special thanks to Ruth M. Kempson, whose many insightful comments served greatly to improve this study. I would also like to acknowledge my debt of gratitude to Akiko Yoshimura, who first introduced me to the labeled-deductive-system approach through discussions of her own research. Of course, responsibility for any inadequacies in this work rests solely with me.

[1]This term is perhaps something of a misnomer, at least in the terminology of some linguistic theories, since elements explicitly mentioned in the elliptical, target clause need not correspond to subcategorized 'arguments' in the source clause.

(i) She yawned in front of John. Bill too.

The target clause in (i) can mean 'She yawned in front of Bill too.' In any case, the name should not be taken to imply a commitment to the position that the explicitly mentioned element from the target clause may not be interpreted as corresponding to an 'adjunct.'

for *his* from the source clause, i.e., *John*; this is called the STRICT reading. In contrast to the strict reading, the interpretation in (6b) represents the SLOPPY reading, where the target-clause argument *Bill* supplants not only *John*, but also the pronoun referring to *John*. The investigation of the interpretive possibilities for such pronouns involved in elliptical structures constitutes a large part of the empirical subject matter for this paper.

In this study I will support theoretical claims on two levels. On the more abstract plane, I shall present evidence that shows the superiority of a procedural approach to ellipsis phenomena that determines the semantic content needed to supplement the meaning of the target clause by building a new relation for the purpose, taking the semantic representation of the source clause as a starting point. Let us call this the CONSTRUCTIVE theory. This technique will be contrasted with one that borrows a ready-made component out of the semantic representation of the source clause in order to fill out the meaning of the target. The latter approach we may call the COMPONENTIAL theory. It is adopted by a number of researchers, from among whose proposals we shall be considering principally that of Gawron and Peters (1990). The two views differ on how they would treat ambiguities like those seen above in (6). The componential approach depends on the existence of a corresponding ambiguity in the source clause, while the constructive approach does not. Instead, the constructive theory allows for a mechanism to create more than one relation as the semantic contribution to the target clause. Dalrymple et al. (1991) have presented compelling arguments suggesting the superiority of the constructive approach, which I shall demonstrate to be equally applicable to the proposals set forth in the present study.

In addition to the general discussion of the merits of the basically procedural notion embodied in the constructive approach, I will also advance claims on the theory-internal level with respect to a particular implementation of the constructive analysis. I will examine a new approach to the study of natural language interpretation based on recent advances in the study of logic due to D. M. Gabbay (forthcoming). He proposes a general logical framework based on LABELED DEDUCTIVE SYSTEMS (LDSs). The application of this notion to natural language has been advanced principally by R. M. Kempson in coordination with Gabbay (1992b). In particular, Kempson (1992) employs the approach to analyze the interpretation of elliptical structures like those in (1)–(4). The constructive theory receives an elegant rendering in the LDS approach, one which automatically handles the differences in interpretation between sloppy and strict pronouns. Kempson also proposes a particular analysis of relative clauses and then shows that it interacts with her treatment of ellipsis to predict automatically that the following sentence may not have the indicated reading.

(7) The woman who likes Bill visited Joan. And Mary too. (Kempson 1992:22)
$\not\Rightarrow$ '... And the woman who likes Mary also visited Joan.'

The generalization that Kempson seeks to capture is that target-clause elements may not correspond to phrases embedded inside of relative clauses in the source. I believe there are two types of counter-examples to this generalization. However, since Kempson's proposals predict this generalization as an automatic consequence, I conclude that Kempson's analysis of relative clauses must be altered. With this change in place, I consider ways of handling data like (7), which motivated Kempson's analysis. Hence, the present study will hopefully make a contribution to the development of LDS-based linguistic analysis.

This paper will begin with a brief review in section 1 of some of the most important ideas in LDS theory. This is of course a general approach to the study of logic, with many diverse applications: we shall concentrate on one of these, devoted to natural language interpretation, a description of which is provided in section 2. From there, we will move in section 3 to the main emphasis of this study, the treatment of ellipsis and the interpretation of pronouns in such constructions.

1 LABELED DEDUCTIVE SYSTEMS

Gabbay (forthcoming) proposes the notion of labeled deductive systems as a measure to unify various different approaches to logic. The basic motivation lies in the observation that differences between separate logics tend to arise not so much in the direct manipulation of formulae, but rather in the meta-level considerations that control the flow of the deductive process. Gabbay's goal is to provide a representation of logical systems that integrates current meta-level features into the object level. The means of achieving this is provided by the use of LABELS connected to logical formulae.

The latter have their usual declarative content, and to that the labels add further information. The type of information encoded in the labels varies from one LDS application to another. As indicated above, the notion of an LDS is quite general, encompassing innumerable possible instantiations. We will later be considering one system out of this multitude, envisaged by Gabbay and Kempson (1992b), in which labels contain components of a semantic representation of a natural language utterance. As logical inferences are performed in an LDS, the labels of premises are combined in predefined manners to produce new labels for the implicata. We shall find that in the LDS to be employed here the inference process causes the components of semantic representation contained in the labels to combine into larger units of meaning. A final important aspect of LDSs to be considered here concerns the compartmentalization of information into structured, interconnected DATABASES. This facet of the system provides a means of modeling various sorts of locality effects in information and is particularly well suited to the analysis of natural language. The present section shall provide a brief and informal introduction to a few of the basic features of LDSs, viewed on a general level, without particular reference to the linguistic application to be highlighted later. A more precise, book-length introduction to the subject is of course furnished by Gabbay (forthcoming).

1.1 Labeled Formulae and Deduction

As already mentioned, LDSs differ from traditional forms of logic in that they associate annotations known as LABELS with formulae. Formulae equipped with labels are known either as LABELED FORMULAE or else as DECLARATIVE UNITS.

(8) Declarative Units

A declarative unit is a pair, written in the special notation $\alpha : \varphi$, where α is a label and φ is a formula.

Any LDS must define what is usable as a formula in declarative units, and also what is usable as a label. The former task is handled by including a logical language of the traditional sort in an LDS, with the understanding that only well formed formulae of that language may occur in declarative units. Similarly the set of acceptable labels is defined by an algebra also contained in the LDS. Furthermore, the LDS must specify what the legal associations of labels with formulae are. Of course, if one is to perform deductions, the LDS must also set down how such procedures are to be conducted, stipulating how inference rules affect both formulae and labels. An LDS is therefore a triple:

(9) Labeled Deductive Systems (LDSs)

A labeled deductive system, or LDS, is a triple $\langle L, \Gamma, \mathcal{M} \rangle$, where

L is a logical language,

Γ is an algebra of labels, and

$\mathcal{M}$ is a discipline of labeling formulae of the logic L (from the algebra of labels Γ), together with deduction rules and agreed ways of propagating the labels via the application of the deduction rules. (Gabbay forthcoming:6)

Let us exemplify all of these notions with a simple LDS, $\langle L_x, \Gamma_x, \mathcal{M}_x \rangle$. Let L_x be a sentential language with three atoms and $\rightarrow$:

(10) Formation Rules for L_x

There are three atomic formulae, P, Q, and R.

If φ and ψ are formulae, then $[\varphi \rightarrow \psi]$ is also a formula.

We shall allow one abbreviatory convention:

(11) Abbreviatory Convention for L_x

Outermost brackets may be omitted.

The algebra Γ_x takes the form $\langle S, \cup, - \rangle$, where S is the set of multisets whose members are drawn from among $\heartsuit$, $\clubsuit$, and $\spadesuit$, and $\cup$ and $-$ are multiset union and multiset difference, respectively. Finally $\mathcal{M}_x$ includes a labeling discipline that allocates fresh singleton sets as labels for assumptions, along with two deduction rules, modus ponens and the conditional proof rule, which essentially provide for the extraction and introduction of $\rightarrow$. The modus ponens rule has the following form:

(12) MODUS PONENDO PONENS (MPP)

$$\frac{\alpha : \varphi \rightarrow \psi \qquad \beta : \varphi}{\alpha \cup \beta : \psi}$$

The conditional proof rule takes the form seen in (13). Please ignore, for the moment, the box that figures in the statement of the rule: it will be explained momentarily.

(13) THE CONDITIONAL PROOF RULE (CP)

$$\boxed{\begin{array}{ll} \alpha : \varphi & \text{assume} \\ \vdots & \\ \beta_{\alpha \in \beta} : \psi & \end{array}}$$
$$\beta - \alpha : \varphi \rightarrow \psi$$

Notice that the above inference rules define not only the form that the formula of the derived declarative unit should take, but also the shape of the new label. These two rules will, by the way, be put to use in the linguistic LDS to be described below: the operations on labels will merely be altered to suit a different algebra.

To demonstrate how the two logical inference rules above function, let us consider a simple derivation. The following proof bears an obvious resemblance to inference procedures employed in traditional logic; consider how the labels associated with the formulae evolve through the course of the derivation.

(14) PROOF OF $\heartsuit : P \rightarrow [Q \rightarrow R] \vdash \alpha : Q \rightarrow [P \rightarrow R]$

$$\boxed{\begin{array}{ll} \text{goal: } \alpha : Q \rightarrow [P \rightarrow R] & \\ 1.\ \heartsuit : P \rightarrow [Q \rightarrow R] & \text{given} \\ \boxed{\begin{array}{ll} \text{goal: } \beta : P \rightarrow R & \\ 2.\ \clubsuit : Q & \text{assume} \\ \boxed{\begin{array}{ll} \text{goal: } \gamma : R & \\ 3.\ \spadesuit : P & \text{assume} \\ 4.\ \heartsuit\spadesuit : Q \rightarrow R & \text{MPP; 1, 3} \\ 5.\ \heartsuit\spadesuit\clubsuit : R & \text{MPP; 4, 2} \end{array}} & \\ 6.\ \heartsuit\clubsuit : P \rightarrow R & \text{CP; 3, 5} \end{array}} & \\ 7.\ \heartsuit : Q \rightarrow [P \rightarrow R] & \text{CP; 2, 7} \end{array}}$$

The goal expressed at the top level of (14) employs α as a variable over labels, since it is not known in advance what label is going to be associated with $Q \rightarrow [P \rightarrow R]$—although we eventually discover that $\alpha = \heartsuit$. Similar comments hold for β and γ in the subsequent goals. The two internal boxes are associated with applications of the conditional proof rule. Each introduces an assumption with its own atomic label—$\clubsuit$ for Q in line 2, and $\spadesuit$ for P in line 3. With the first application of MPP in line 4, we have multiset union of the label of the major premise, line 1, with that of the minor premise, line 3. The resultant label, $\heartsuit\spadesuit$, reflects the dependency of the formula so-labeled on the two premises labeled by $\heartsuit$ and $\spadesuit$. The same could be said for line 5, where the label $\heartsuit\spadesuit$ of the major premise in line 4 is combined the label $\clubsuit$ of the minor premise in line 2, yielding $\heartsuit\spadesuit\clubsuit$. With the completion of each conditional proof step in the derivation, assumptions are discharged, and this is represented in the labels by the suppression of assumption labels through multiset subtraction. For instance, in line 6, the assumption from line 3 is discharged, and the label employed reflects this, since the $\spadesuit$ has been eliminated. Similarly, $\clubsuit$ drops out of the label of line 7, reflecting the fact that the formula in this line depends only on the given premise in line 1, whose label, $\heartsuit$, remains.

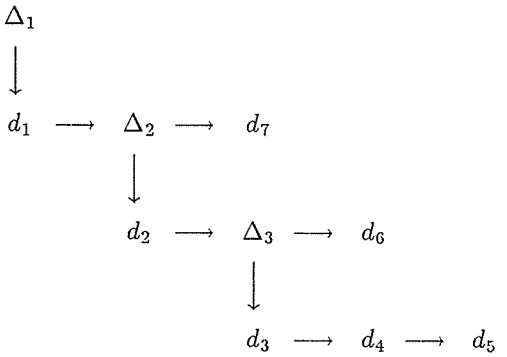
Hopefully, the above gives something of the flavor of the use of labels to supplement logical formulae with extra information, e.g., recording facts about the progress of a derivation. We next move on to issues of organization above the level of the declarative unit.

1.2 Databases and Linking

Declarative units are grouped together in DATABASES. Graphically, databases are rendered as boxes surrounding the declarative units that form their contents. Formally, databases may be any of a variety of structures, such as normal sets, multisets, partially ordered sets, etc., depending on the particular LDS in question. The utility of having structured databases lies in the potential for implementing a notion of locality of information. For instance, it will probably have been noticed that in (14) database boxes correspond to goal structure and to local domains in which assumptions are available. Recall also that the conditional proof rule was stated with the aid of a box: it is a box-exit rule associated with the retraction of a premise, whose domain was that database preceding the newly introduced implication.

One way to render the database structure underlying (14) would be with the following directed tree:

(15) The Database in (14) Viewed as a Graph-Theoretic Directed Tree



The vertices Δ_1, Δ_2, and Δ_3 represent the databases of (14), Δ_1 being the outermost and Δ_3 the innermost. The d_i vertices correspond to declarative units: in each case i matches the line number applied to the relevant declarative unit in (14). Notice that the tree has been so-arranged that the successor relation, symbolized with arrows, records the sequential order of the declarative units in a database. This observation is pointed out here simply to underline the fact that the contents of a database may be ordered and that relations like (immediate) predecessor or (immediate) successor are straightforwardly definable. This capability will be put to use in the linguistic application of LDS theory described below. Also a tree-structured database along the lines of (15) lends itself to a vertex nomenclature that uniquely identifies every element by means of the path that leads to it from the root. This facilitates effective definitions of various sorts; however, in this paper we do not aspire to that level of detail, so we will largely ignore refinements of this kind. The interested reader is referred to Gabbay and Kempson (1992b).

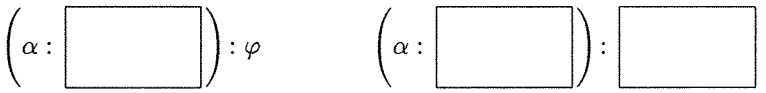
In some applications of LDS theory it is necessary to mark databases with information of a kind distinct from that recorded in the labels and formulae of the declarative units contained in those databases. For instance, one might wish to retain some indication of the source of a given group of assertions. Toward this end, databases are furnished with labels just as formulae are. A database label serves as a repository for information relevant to a whole body of information, i.e., the content of the database. Graphically, a labeled database will be denoted with the same colon-based convention as is employed with declarative units:

(16) Notation for a Labeled Database

The character and content of the labels varies according to the LDS application. In the linguistics-oriented LDS to be discussed in a subsequent section, we shall find that database labels hold information about tense.

It is not only the case that databases are provided with labels: labels may themselves in turn be databases. To appreciate what sort of considerations might motivate such a move, consider a case where one wishes to record the fact that a given formula depends in some way or other on a database. What better way to signal this fact than by actually using the database in question as the label for the dependent formula. The notation to be used in such a case is the following:

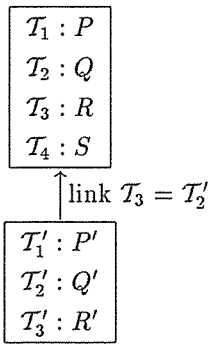
(17) NOTATION FOR A DATABASE USED AS A LABEL



Large parentheses have been introduced to make it easier to appreciate at a glance how labels, formulae, and databases are grouped. In the linguistic LDS application to be considered below, databases as labels are employed to model complement clauses—as opposed to the adjunct variety.

There is another way of expressing a relationship between databases, this time with an emphasis on connections mediated through particular declarative units. The technique is called LINKING and establishes a relationship between the labels from two declarative units in distinct databases. In this connection, imagine a system of databases containing declarative units whose labels represent times at which the associated formulae are true. Suppose furthermore, that these chronological labels are abstract, not decipherable into standard time. Even if the ordering of declarative units within a given database may be ensured, one would still face the problem of establishing how the contents of one database related to that of another. To accomplish this task, one could link the chronological labels of two declarative units, the implied semantics being that the labels so-connected are interpreted as being synchronous. Such an inter-label relation may be denoted as follows:

(18) DATABASES WITH TEMPORAL LABELS LINKED FOR SYNCHRONIZATION



We will find that in our linguistic application, linking is the vehicle for relating certain adjunct modifiers to the things they modify.

This completes our review of LDS basics, so in the next section we will turn to the application of LDS theory to natural language interpretation.

2 THE LINGUISTIC APPLICATION OF LDSS

Gabbay and Kempson (1992b) propose a linguistic application of LDS theory that accomplishes natural language interpretation through deductions on premises and procedural cues provided by lexical items. The procedural view that they put forth is claimed to circumvent the need for the combinatoric definitions of natural language structure in current use among linguists, actually predicting certain phenomena more elegantly than is possible on the usual approach. The present section sets out a brief review of the highlights of Gabbay and Kempson's techniques in view of subsequently examining how they apply to the case of ellipsis.

2.1 PARSING AS DEDUCTION

The LDS that handles parsing employs a very simple sentential language which we shall call L_0. The basic insight that motivates the design of L_0 is Pereira and Warren's (1983) observation that the parsing problem could be viewed as deduction on logical types associated with the semantic

representations of natural language expressions. The details of the LDS application of this concept are set out below.

We will assume a type system with the basic types e, for terms or 'entities,' and t, for formulae or 'truth-value-bearing' units. The usual functional theory of types also includes functions between types, e.g., $\langle e, \langle e, t\rangle\rangle$, which maps from the type e to the type $\langle e, t\rangle$, the latter being a function from e to t. The consequence is that when an expression of type $\langle e, \langle e, t\rangle\rangle$ applies to another expression of type e, the result is an expression of type $\langle e, t\rangle$. Furthermore, when the resultant expression combines with another expression of type e, the outcome is an expression of type t.[2] The various components of the functional theory of types are modeled in the language L_0 in the manner explained next.

The logical language L_0 is defined by the following two rules:

(19) FORMATION RULES FOR L_0

There are two atomic formulae, e and t.

If φ and ψ are formulae, then $[\varphi \rightarrow \psi]$ is also a formula.

We shall allow one abbreviatory convention:

(20) ABBREVIATORY CONVENTION FOR L_0

Outermost brackets may be omitted.

The two atomic formulae e and t correspond to the two basic logical types with identical names. The single connective $\rightarrow$ of L_0 models the effects of function application in the standard type theory. The functional turn of thought, e.g., '$\langle e, t\rangle$ applied to e yields t' gives way to a deductive style of thinking, e.g., 'the premises $e \rightarrow t$ and e allow one to infer t by modus ponens.' The goal of the deduction is to *prove* that a given linguistic form is a sentence, by proceeding from premises consisting of the L_0 representations of the types of the various lexical items involved and arriving at the conclusion t—viz. the logical type of formulae or sentences. Now, given only what has been said up to this point, we have described a deductive recognition system, capable of confirming that a string is indeed a sentence. However, yes-or-no well-formedness recognition is of limited interest, if one does not obtain, in the end, a parse tree or other artifact representing in some way or other the string one has taken pains to process. Toward the end of deriving a useful representation, we shall incorporate L_0 into an LDS by associating it with an algebra of labels which will evolve through the course of logical derivations into semantic representations expressed in a distinct logical language called L_1. In other words, when a proof in the proposed LDS ends with the conclusion $\alpha : t$, the label α will be the L_1 translation of the sentence being processed.

L_1 will be a higher order language of the following form. The non-logical constants of L_1 will be represented as English words in the following typeface: **John**, **respect**, etc. We will use the functional theory of types alluded to in the foregoing discussion to describe the syntax of L_1. Consequently, all non-logical constants will be associated with types; a partial list follows. Let $\leftarrow$ be interpreted as 'is assigned.'

(21) TYPE ASSIGNMENTS FOR CONSTANTS OF L_1

John, **Mary** $\leftarrow e$

fall $\leftarrow \langle e, t\rangle$

like, **respect** $\leftarrow \langle e, \langle e, t\rangle\rangle$

For the purposes of this paper we will be interested only in individual variables, i.e., of type e, which will be lower-case Roman letters drawn from the latter part of the alphabet. The formation rules for L_1 are as follows. A fully rounded out definition is provided, even though most of the features described below will not be used in this study.

(22) FORMATION RULES FOR L_1

A non-logical constant or variable of type a is a basic expression of type a in L_1.

[2] Type theory is familiar to linguists mainly through the work of R. Montague and his followers. However, the type theory used in that branch of formal semantics incorporates provisions for intensional types, which are absent from the type system set out here. A description readily accessible in the standard linguistics literature of the sort of more basic, non-intensional type theory employed here is provided by Dowty et al. (1981:83ff.).

If α is of type $\langle a, b\rangle$, and β is of type a, then $\alpha(\beta)$ is an expression of type b.

If φ is of type t, then $\neg\varphi$ is also an expression of type t.

If φ and ψ are of type t, then $[\varphi \rightarrow \psi]$ is also an expression of type t.

If φ is of type t, and v is a variable of any type, then $\forall v[\varphi]$ is also an expression of type t.

If α is of type a, and v is a variable of type b, then $\lambda v[\alpha]$ is an expression of type $\langle b, a\rangle$.

We also provide the following abbreviations for L_1:

(23) ABBREVIATORY CONVENTIONS FOR L_1

$[\varphi \vee \psi]$ abbreviates $[\neg\varphi \rightarrow \psi]$.

$[\varphi \wedge \psi]$ abbreviates $\neg[\varphi \rightarrow \neg\psi]$.

$[\varphi \leftrightarrow \psi]$ abbreviates $[[\varphi \rightarrow \psi] \wedge [\psi \rightarrow \varphi]]$.

$\exists v\varphi$ abbreviates $\neg\forall v\neg\varphi$.

$[\varphi \vee \psi \vee \ldots]$ abbreviates $[\varphi \vee [\psi \vee \ldots]]$, and similarly for $\wedge$.

Outermost brackets may be omitted.

$[[\varphi]]$ may be written as $[\varphi]$.

Since expressions of L_1 will be used as labels, the definition of LDSs requires us to form an algebra around them. This will consist of the set of all expressions in L_1 along with operations corresponding to the various formation rules for non-atomic expressions above. Hence, the algebra is $\langle L_1, f, \neg, \rightarrow, \forall, \lambda\rangle$, where f represents function application, i.e., $f(\alpha, \beta) = \alpha(\beta)$.

We must also consider a labeling discipline to associate labels with L_0 formulae. Basically, this will require that the formula of a declarative unit be the L_0 rendering of the logical type of the label. A couple of examples using logical constants of L_1 would be **respect** : $e \rightarrow [e \rightarrow t]$ and **Mary** : e.

Finally let us consider deduction rules for our LDS. These will include at least revised versions of modus ponens and the conditional proof rule introduced previously:

(24) MODUS PONENDO PONENS (MPP)

$$\frac{\alpha : \varphi \rightarrow \psi \qquad \beta : \varphi}{\alpha(\beta) : \psi}$$

(25) THE CONDITIONAL PROOF RULE (CP)

$\alpha : \varphi$	assume
$\vdots$	
$g(\alpha) : \psi$	

$\lambda x[g(x)] : \varphi \rightarrow \psi$

Notice that the treatment of derived labels is different from that employed before. Modus ponens stipulates that function application be applied to the labels of the premise. The g in the statement of the conditional proof rule represents some composition of algebraic operations applied to α and other labels to yield a complex label that contains α. In the final step of the rule, α is replaced with the abstraction variable x. As before, the conditional proof rule corresponds to the retraction of an assumption, viz. $\alpha : \varphi$.

Notice that the two logical inference rules described above, modus ponens and the rule of conditional proof, apply in this LDS in such a way that derived declarative units have well-formed expressions of L_1 as their labels, and the corresponding formulae are appropriate L_0 renderings of each label's logical type. For instance, take the two labeled formulae **respect** : $e \rightarrow [e \rightarrow t]$ and **Mary** : e as premises to modus ponens: application of that inference rule would yield as output **respect**(**Mary**) : $e \rightarrow t$. The label in this implicatum is a well formed L_1 expression of type $\langle e, t\rangle$, and that type is accurately reflected in the derived L_0 formula $e \rightarrow t$. Now let us consider the other inference rule. Suppose that one is given the premise **respect** : $e \rightarrow [e \rightarrow t]$. In anticipation of applying the conditional proof rule, one may then assume **Mary** : e. Suppose further that

one then derives **respect(Mary)** : $e \rightarrow t$: the conditional proof rule says we may now conclude λx[**respect**(x)] : $e \rightarrow [e \rightarrow t]$. Since **respect**($x$) is of type $\langle e, t\rangle$, and x is type e, λx[**respect**(x)] would be of type $\langle e, \langle e, t\rangle\rangle$: notice that the last type matches the derived L_0 formula $e \rightarrow [e \rightarrow t]$.

Now let us step back from the details of the LDS and consider how the system could be used to model the parsing process. The deduction that drives the parsing process starts with premises provided by lexical items. Thus, we require a lexicon listing declarative units associated with lexical items. A partial example follows.

(26) LEXICON

falls ← **fall** : $e \rightarrow t$

John ← **John** : e

likes ← **like** : $e \rightarrow [e \rightarrow t]$

Mary ← **Mary** : e

respects ← **respect** : $e \rightarrow [e \rightarrow t]$

With the lexicon in place, we conduct a parse by looking up the declarative units associated by the lexicon with each lexical item that occurs in the string under consideration. The declarative units collected in this lexical analysis step become the initial premises for a proof whose goal is $\alpha : t$, where α is a variable which stands in for the as yet unknown, final label. The proof then proceeds in the normal manner, advancing one step at a time through the application of modus ponens and the rule of conditional proof, as already outlined in the preceding discussion. As an example of a simple case, consider the following sentence in (27a) and the associated proof in (27b):

(27) a. John respects Mary.

b. PROOF OF {**John** : e, **respect** : $e \rightarrow [e \rightarrow t]$, **Mary** : e} $\vdash$ $\alpha : t$

goal: $\alpha : t$	
1. **John** : e	assume
2. **respect** : $e \rightarrow [e \rightarrow t]$	assume
3. **Mary** : e	assume
4. **respect(Mary)** : $e \rightarrow t$	MPP; 2, 3
5. **respect(Mary)(John)** : t	MPP; 4, 1

The foregoing derivation was successful, in that it concluded with a declarative unit where the label was an appropriate L_1 translation for the sentence under consideration. However, a little thought will reveal a variety of ways in which the initial premises specified by the lexical items could have been employed to derive a different conclusion $\alpha' : t$, where α' is an inappropriate L_1 rendering of the linguistic input. Obviously, how the derivation is carried out is of paramount importance. The next section will touch on this problem and sketch out ways of overcoming the difficulty.

2.2 PROCEDURAL RESTRICTIONS

This section describes restrictions on the processing procedure, which inhibit the derivation of various erroneous interpretations that might arise for a given linguistic input. Consider the sentence in (28) and the accompanying group of L_1 formulae, only one of which is a suitable rendering of the English expression. Let us use ⇒ informally to mean 'is interpreted as.'

(28) John respects Mary.

⇒ **respect(Mary)(John)**

⇏ **respect(Mary)(Mary)**

⇏ **respect(John)(John)**

⇏ **respect(John)(Mary)**

Given only what we have said so far, any of these formulae could be derived as an interpretation for the English expression.

The following is a demonstration of how the second of the L_1 renderings in (28) could be produced: the derivation for the third is analogous.

(29) a. John respects Mary.

b. PROOF OF $\{\mathbf{John} : e, \mathbf{respect} : e \rightarrow [e \rightarrow t], \mathbf{Mary} : e\} \vdash \alpha : t$

goal: $\alpha : t$	
1. $\mathbf{John} : e$	assume
2. $\mathbf{respect} : e \rightarrow [e \rightarrow t]$	assume
3. $\mathbf{Mary} : e$	assume
4. $\mathbf{respect}(\mathbf{Mary}) : e \rightarrow t$	MPP; 2, 3
5. $\mathbf{respect}(\mathbf{Mary})(\mathbf{Mary}) : t$	MPP; 4, 3

The two facts to note here are that $\mathbf{Mary} : e$ in line 3 was used twice for modus ponens and that $\mathbf{John} : e$ in line 1 is never used at all. The obvious observation to make here is that all premises provided by the lexical items should be used at least once in the derivation, and, furthermore, that they should be used at most once. The 'at least once' part of this constraint is useful not only for preventing the problem seen in the second and third L_1 renderings of (28), but also for predicting the unacceptability of the occurrence of unsubcategorized arguments, as in (30).

(30) a. *John falls Mary.

b. PROOF OF $\{\mathbf{John} : e, \mathbf{fall} : e \rightarrow t, \mathbf{Mary} : e\} \vdash \alpha : t$

goal: $\alpha : t$	
1. $\mathbf{John} : e$	assume
2. $\mathbf{fall} : e \rightarrow t$	assume
3. $\mathbf{Mary} : e$	assume
4. $\mathbf{fall}(\mathbf{John}) : t$	MPP; 2, 1

Notice that the conclusion on line 4 in (30) has t as its formula: the derivation yields this false positive result, because the lexical assumption in line 3 was never used.

As for the 'at least once' stipulation above, it might seem superfluous, if one considers only (29). That is to say, the 'at most once' constraint alone would suffice to rule out the problematic derivation. Alternatively the 'at most once' clause could be used in place of the 'at least once' version to achieve the same effect. However, the 'at least once' constraint was motivated by (30), and, similarly, the 'at most once' restriction serves to rule out cases where subcategorized arguments are lacking, as in (31).

(31) a. *John likes.

b. PROOF OF $\{\mathbf{John} : e, \mathbf{like} : e \rightarrow [e \rightarrow t]\} \vdash \alpha : t$

goal: $\alpha : t$	
1. $\mathbf{John} : e$	assume
2. $\mathbf{like} : e \rightarrow [e \rightarrow t]$	assume
3. $\mathbf{like}(\mathbf{John}) : e \rightarrow t$	MPP; 2, 1
4. $\mathbf{like}(\mathbf{John})(\mathbf{John}) : t$	MPP; 3, 1

The derivation in (31) yields a false positive, because line 1 was used twice. Thus both the 'at least once' and the 'at most once' stipulations are independently required.

The stipulation that assumptions provided by lexical items be used at least and at most once can be modeled using standard techniques developed within the study of logic. For instance, one branch of logic, namely RESOURCE LOGIC, requires that one keep track of which part of the database proves what. Hence, the logic needed for the present linguistic application would be a resource logic, because it requires that one pay attention to what lexical assumptions have

been used. Furthermore, there are established varieties of logic that place controls on the use of assumptions: RELEVANCE LOGIC accepts only derivations that use all assumptions, and LINEAR LOGIC accepts only derivations that use all assumptions exactly once. Hence, the logic required here would appear to be a linear resource logic. The constraints on the derivation process could be implemented in an LDS by means of label manipulations. However, we will not pursue this problem any farther, beyond noting that an effective solution is available. We will therefore proceed with further discussions, applying the 'at least once' and 'at most once' constraints intuitively.

Despite the adoption of restrictions to produce a linear resource logic, we still have not ruled out examples with too many arguments like (30a). In the examples considered so far, we have applied only the rule of modus ponens. If we go on to consider the conditional proof rule, we discover another potential problem. Consider the following derivation:

(32) a. *John falls Mary.

b. PROOF OF $\{\mathbf{John} : e, \mathbf{fall} : e \rightarrow t, \mathbf{Mary} : e\} \vdash \alpha : t$

goal: $\alpha : t$	
1. $\mathbf{John} : e$	assume
goal: $\beta : t$	
2. $\mathbf{fall} : e \rightarrow t$	assume
3. $\mathbf{Mary} : e$	assume
4. $\mathbf{fall}(\mathbf{Mary}) : t$	MPP; 2, 3
5. $\lambda x[\mathbf{fall}(x)] : e \rightarrow t$	CP; 2, 4
6. $\mathbf{fall}(\mathbf{John}) : t$	MPP; 5, 1

Notice that all the assumptions have been used exactly once. Despite this fact, we were able to come up with a derivation for a sentence with two nominal arguments for an intransitive verb. This was accomplished by using one of the nominal arguments and then canceling it out with the conditional proof rule. Obviously we must find a way to prevent this sort of derivation. A very natural solution presents itself, if we consider the database structure in (31). There are two database boxes, one embedded inside of the other. It was necessary to do this because the conditional proof rule that produced the λ-abstraction is a box-exit rule that retracts a local assumption. Thus, there is no way to avoid having an embedded box that contains the retracted assumption, if the conditional rule is to apply. Consequently, we can control the undesirable application of the conditional proof rule by getting a handle on the propagation of database boxes. We shall therefore limit the construction of new, embedded databases to contexts that correspond to embedded clause structures in the input utterance. The general technique involves recognizing the probable boundary of an embedded clause on the basis of the preceding syntactic context; when such a boundary is discovered, a subordinate database is constructed, and subsequently encountered lexical information flows into the new, embedded database. When the syntactic context decrees that the subordinate clause has been exhausted, the embedded database is exited and processing continues in the superordinate box. This turn of thought may very well seem alien in the context of currently popular linguistic theory, but notions very similar to this are routinely discussed in the computer science literature under the rubric of shift-reduce parsing.[3] By controlling the production of databases in this manner, all of the assumptions representing the arguments in a clause will be grouped together in a single box; consequently the undesirable application of the conditional proof rule cannot be used to exhaust unsubcategorized arguments and thereby admit ungrammatical sentences with inappropriate readings.

Let us now move on to the final problem posed in (28), that of preventing *John respects Mary* from being interpreted as $\mathbf{respect}(\mathbf{John})(\mathbf{Mary})$, in which the arguments of the predicate are reversed. The problem stems from the lack, so far, of any restrictions on the order in which premises are used in the course of a derivation. The ill-fated order in question is illustrated in (33).

[3] For more on this topic, the reader may consult any compiler design book, such as the one by Aho et al. (1986). To get a few ideas about the application of this style of thinking to ambiguous, natural-language inputs, see Tomita (1985).

(33) a. John respects Mary.

b. PROOF OF $\{\mathbf{John} : e, \mathbf{respect} : e \rightarrow [e \rightarrow t], \mathbf{Mary} : e\} \vdash \alpha : t$

goal: $\alpha : t$	
1. $\mathbf{John} : e$	assume
2. $\mathbf{respect} : e \rightarrow [e \rightarrow t]$	assume
3. $\mathbf{Mary} : e$	assume
4. $\mathbf{respect}(\mathbf{John}) : e \rightarrow t$	MPP; 2, 1
5. $\mathbf{respect}(\mathbf{John})(\mathbf{Mary}) : t$	MPP; 4, 3

The premise in line 1 is used before that in line 3, whereas the opposite would have given rise to the correct reading. The solution to this problem will once again lie in controlling the flow of the proof process. The repository for instructions to regulate progress through a proof will be the labels of the LDS, just as in the case of the constraints (alluded to above but not expressed) that gave rise to a linear resource logic. Obviously, labels will have to become somewhat more complex objects than we have indicated so far, and it is to this matter that we turn next.

Henceforth, all labels will be pairs, $\langle C, R \rangle$, where C is a CONTENT LABEL, and R is a RESOURCE LABEL. The content label is what we have been using up to this point as labels, i.e., an expression of L_1, while the resource label is a set of instructions to control the flow of a derivation. Graphically, labels will take on the form $\langle \alpha, \{\ldots\} \rangle$; however, to enhance visual clarity, labels of the form $\langle \alpha, \emptyset \rangle$ will be abbreviated simply as α. Of course, if we change the form of labels in our declarative units, we ought to go back and revise the definition of our algebra of labels. However, it probably would not be very enlightening to descend to that level of detail, so we shall proceed, assuming that all of the appropriate changes have been effected in the background. Now consider the following database, into which some resource labels have been inserted:

(34) a. John respects Mary.

b. PROOF OF $\{\mathbf{John} : e, \mathbf{respect} : e \rightarrow [e \rightarrow t], \mathbf{Mary} : e\} \vdash \alpha : t$

goal: $\alpha : t$	
1. $\langle \mathbf{John}, \{\text{use me last}\} \rangle : e$	assume
2. $\mathbf{respect} : e \rightarrow [e \rightarrow t]$	assume
3. $\langle \mathbf{Mary}, \{\text{use me first}\} \rangle : e$	assume
4. $\mathbf{respect}(\mathbf{Mary}) : e \rightarrow t$	MPP; 2, 3
5. $\mathbf{respect}(\mathbf{Mary})(\mathbf{John}) : t$	MPP; 4, 1

The processing instructions 'use me first' and 'use me last' indicate that the minor premises so-labeled should be respectively the first and last members of a chain of premises to be combined with the major premise $\mathbf{respect} : e \rightarrow [e \rightarrow t]$, leading to the conclusion $\alpha : t$. This addition would fix the order of derivation in a manner that would ensure the right order of arguments in the ultimate semantic rendering for *John respects Mary*. These processing cues inserted into the new complex labels can provide a solution to the problem of controlling the derivation, but we must specify how it is that they come to be in the database in the appropriate places.

Obviously the resource label 'use me last' is not inherently associated with **John**, since one could just as easily have said *Mary respects John*, in which case the proper resource label for **John** would have been 'use me first.' The obvious goal is to insert the resource labels 'use me first' and 'use me last' according to the content of the sentence, which means that they will have to be handled as a part of the processing procedure. To arrange for this to happen, we shall enrich the lexicon in such a way that some lexical items will carry processing instructions in addition to labels and formulae. A few relevant examples follow: the proper nouns *John* and *Mary* have no instructions, while the verb *respects* has two.

(35) MODIFIED LEXICON

John $\leftarrow \mathbf{John} : e$

Mary $\leftarrow \mathbf{Mary} : e$

respects ← **respect** : $e \rightarrow [e \rightarrow t]$

i. Insert 'use me last' into the resource label of the assumption associated with the lexical item immediately before me in the sentence.

ii. Insert 'use me first' into the resource label of the assumption associated with the lexical item immediately after me in the sentence.

The instructions associated with *respects* mention things that haven't really been given effective definitions, such as 'immediately before me' and so forth. We shall content ourselves with the observation that an effective definition is available and then proceed to apply these instructions intuitively.

The implementation of the lexical instructions assigned to *respects* (and other verbs) depends on some supporting features of the database structure and the processing procedure. First recall the discussion about databases, where it was observed that they could be modeled with a variety of mathematical structures, including a graph-theoretic directed tree. It was further observed that by associating declarative units with successive vertices along a path in such a tree, the tree structure of the database would define an order on its contents. Now, consider the lexical analysis procedure that initially collects declarative units for each lexical item in the input sentence: we can trivially make that procedure insert those units into a database in a compact, totally ordered list that respects the order of lexical items in the sentence. The procedure will traverse the sentence from left to right, making the declarative unit for the leftmost lexical item the graph-theoretic successor of the database vertex, and the declarative unit for each subsequent lexical item the successor of the most recently introduced vertex. Only after this lexical analysis step is completely finished would inference using the lexically derived assumptions proceed. Let us adopt, by the way, the graphic convention that databases will be pictured with declarative units descending from top to bottom in the order imposed by the structure of the database. All previously displayed databases adhere to this practice. The foregoing observations suggest that it would be straightforward to develop thoroughly formalized replacements for the English instructions in (35) on the basis of the structure of the database and the predictable distribution therein of the lexically based assumptions. Indeed such an alternative is provided by Gabbay and Kempson (1992b:88ff.).

With the above observations out of the way, we can return to a more intuitive consideration of the instructions in (35). They obviously specify that 'use me first' and 'use me last' be distributed exactly as they are in (34), thus providing a solution to the argument ordering problem that initiated this discussion. Of course, the specific rules presented here are hardly impressive in their empirical coverage: after all, such phenomena as inversion and heavy shift would suffice to confound them.[4] However, shortcomings of simplified, example rules are not really such a serious defect. Rather, the discussion of control over the flow of deduction conveys an idea of what could be achieved in a procedural account of sentence interpretation with a more highly developed set of resource labels and lexical instructions. On this subject it is worth raising the point that to some degree methods are known for translating knowledge about combinatoric properties of language into procedural routines for parsing. This is the basis of compiler-oriented practices such as the automated creation of action tables for shift-reduce parsing, after all. Indeed Gabbay, Kempson, and Pitt (1992) report on a prototype LDS natural language interpreter whose procedural constraints were composed partly from combinatoric grammars by automation of this kind. These comments are meant to suggest that there is reason to hope that procedural routines of considerably greater empirical accuracy than those displayed above may eventually be producible.

In any case, we shall henceforth leave the 'use me first' and 'use me last' labels unexpressed in our figures and move on to an examination of the labeling and structuring of databases.

2.3 Database Structuring and Linguistic Modeling

In the previous section we concentrated on the declarative units that are used in the LDS implementation of the parsing as deduction method: in the present section we turn our attention more

[4] Alternations in active, passive, and middle voice are not so worrisome in this regard, because such variants of the same verb would be furnished with different sets of instructions—all instantiated, we may presume, by some form of yet to be established lexical rules. By the way, the simplified, highly stipulative lexica displayed here should be taken merely as trivial examples. To flesh out a sizable lexicon in the manner shown so far would be to miss a number of generalizations and to throw economy to the wind.

to the database level.

Recall that databases are labeled, just as are declarative units. As already briefly mentioned above, the labels on databases in our LDS for natural language interpretation are repositories for information about tense. However, in English tense is marked on the verb, so the question arises as to how the relevant information comes to be correctly incorporated into the database. The method for accomplishing this relies on lexical instructions of the following sort; we once again modify the lexical entry for *respects*.

(36) *respects* $\leftarrow$ **respect** : $e \rightarrow [e \rightarrow t]$

i. Insert 'use me last' into the resource label of the assumption associated with the lexical item immediately before me in the sentence.

ii. Insert 'use me first' into the resource label of the assumption associated with the lexical item immediately after me in the sentence.

iii. Insert $\langle s_i, \{\Theta(s_i) = s_{utt}\}\rangle$ as the label of the database that contains me.

This leads to databases labeled as follows.

(37) John respects Mary.

$\langle s_1, \{\Theta(s_1) = s_{utt}\}\rangle$:

goal: $\alpha : t$	
1. **John** : e	assume
2. **respect** : $e \rightarrow [e \rightarrow t]$	assume
3. **Mary** : e	assume
4. **respect(Mary)** : $e \rightarrow t$	MPP; 2, 3
5. **respect(Mary)(John)** : t	MPP; 4, 1

Θ is the instantiation function, and s_{utt} refers to the time of utterance. Hence, the effect of the label inserted by the new instruction in the lexical entry for *respects* is to constrain the instantiation of the database label s_1 in such a way that it matches the time of utterance. Since this information is not really crucial for our exposition below, we shall drop tense constraints out of subsequent database figures for greater visual clarity.

Now, if tense information is recorded in database labels, then it follows from the general comments about databases and their labels set out in a previous section that tense constraints thereby become a shared feature of the whole body of information contained in the database. Not coincidentally, databases in our LDS correspond to clauses in the linguistic input, since the clause is the domain of tense. Hence, natural language clause structure is reflected by a network of interconnected databases, the relationships among them being of two types, corresponding to complements and adjuncts, as briefly mentioned above. We turn next to these two types of connections.

Complement taking verbs like *know* specify that their first minor premise be labeled with a database. The requisite declarative unit with an empty database as a label is therefore created and inserted into the current database, and the processor shifts down into the new database to begin filling it up with lexically based assumptions. The end result is something like the following:

(38) John knows that Mary succeeds.

s_1 :

goal: $\alpha : t$		
1. **John** : e		assume
2. **know** : $e \rightarrow [e \rightarrow t]$		assume
3. s_2 : goal: $\beta : t$		: e
4. **Mary** : e	assume	
5. **succeed** : $e \rightarrow t$	assume	
6. **succeed(Mary)** : t	MPP; 5, 4	
7. **know**$(s_2$: **succeed(Mary)**$)$: $e \rightarrow t$		MPP; 2, 3
8. **know**$(s_2$: **succeed(Mary)**$)$**(John)** : t		MPP; 7, 1

Next we shall consider the treatment of relative clauses, which differs somewhat from that of complement clauses just described. The following is the analysis which Kempson (1992) proposes for relative clauses in general; however, in what follows I shall propose that it be applied only to non-restrictive relatives. The treatment involves building a separate database to contain the information drawn from the lexical items that make up the relative clause, but, unlike the case of complement clauses, the database representing the relative clause is not incorporated into the database of the host clause. Instead, the two clauses are connected by linked labels, in the manner described in a previous section.

Relative pronouns like *who* have no labels, containing instead only procedural instructions. For instance, suppose that the processor is working through the following sentence and has just seen the word *Mary*:

(39) John respects Mary, who succeeds.

Encountering the word *who*, the processor looks up its lexical entry and finds instructions to build a database. However, there is an added twist to this case of database construction: the goal of the database is to prove the form $\beta[x] : t$, where $\beta[x]$ describes a formula of L_1 which crucially must contain an instance of the variable x; meanwhile, that variable x is linked to the label of the preceding declarative unit, i.e., **Mary**. In the end, one has the following as the finished derivation:

(40) John respects Mary, who succeeds.

s_1 :

goal: $\alpha : t$	
1. **John** : e	assume
2. **respect** : $e \rightarrow [e \rightarrow t]$	assume
3. **Mary** : e	assume
4. **respect**(**Mary**) : $e \rightarrow t$	MPP; 2, 3
5. **respect**(**Mary**)(**John**) : t	MPP; 4, 1

link $x =$ **Mary** ↑

s_2 :

goal: $\beta[x] : t$	
6. $x : e$	assume
7. **succeed** : $e \rightarrow t$	assume
8. **succeed**(x) : t	MPP; 7, 6

Note that the two databases above are separate, save for the link holding between two labels. Contrast this with the case of the database representing a complement clause, which, as a label, was wholly contained with the database of its host clause.

2.4 The Interpretation of Pronouns and Definite Descriptions

In all previous cases, we have examined lexical items that contributed constants of L_1 as content labels to the LDS database. In this section, however, we will examine expressions whose content is determined 'on line' as it were, as the interpretation procedure is at work. The forms in question are pronouns and definite descriptions. Their content labels are variables, and their resource labels provide constraints on how those variables may be instantiated. Consider the following lexical items for a pair of pronoun forms:

(41) Lexicon

$$him \leftarrow \langle v, \{\Theta(v) \notin s_i, \mathbf{male}(\Theta(v))\}\rangle : e$$

$$himself \leftarrow \langle v, \{\Theta(v) \in s_i, \mathbf{male}(\Theta(v))\}\rangle : e$$

As already mentioned, Θ is the instantiation function, which maps a variable to its eventual instantiation. The resource labels specified above help determine the proper instantiation. The expression $\Theta(v) \in s_i$ is a shorthand way of saying that the value ultimately assigned to the variable v, must be drawn from the local database, s_i. Also, the inclusion of statements like $\mathbf{male}(\Theta(v))$ indicates that the eventual referent of the pronoun must satisfy the predicate **male**.

Now let us consider what happens during processing. When the procedure encounters a pronoun, it puts the declarative pair specified in the lexicon into the database. This is seen in line 3 of (42).

(42) John likes himself.

	goal: $\alpha : t$	
	1. $\mathbf{John} : e$	assume
	2. $\mathbf{like} : e \rightarrow [e \rightarrow t]$	assume
s_1 :	3. $\langle v, \{\Theta(v) \in s_1, \mathbf{male}(\Theta(v))\}\rangle : e$	assume
	4. choose $\Theta(v) = \mathbf{John}$	
	5. $\mathbf{like}(\mathbf{John}) : e \rightarrow t$	MPP; 2, 3
	6. $\mathbf{like}(\mathbf{John})(\mathbf{John}) : t$	MPP; 5, 1

Then an instantiation for the variable is sought with the choose operation, care being taken to ensure that the constraints in the resource label for the pronoun are satisfied. The instantiation then stands in for the variable in subsequent algebraic manipulations during the course of the derivation.

Definite descriptions are treated in roughly the same way. The difference is that the principle constraint on instantiation with definites is that the eventual referent of the variable initially introduced in the database must uniquely satisfy the predicate associated with the head noun.

(43) The man fell.

	goal: $\alpha : t$	
	1. $\langle v, \{\mathbf{man}(\Theta(v)), \forall w[\mathbf{man}(w) \rightarrow w = \Theta(v)]\}\rangle : e$	assume
s_1 :	2. choose $\Theta(v) = \mathbf{John}$	
	3. $\mathbf{fall} : e \rightarrow t$	assume
	4. $\mathbf{fall}(\mathbf{John}) : t$	MPP; 3, 1

Though in the present examples we have chosen to instantiate the variable associated with the pronoun or definite description immediately upon encountering it, one could postpone instantiation, using the variable rather than the instantiated constant in subsequent derivational steps. In the subsequent discussion of ellipsis we shall find it necessary to postpone instantiation in order to derive certain readings.

This concludes our general overview of the LDS application for natural language interpretation, and we turn next to Kempson's (1992) analysis of ellipsis.

3 The Interpretation of Elliptical Structures

In this section we take up the main empirical concern of this paper, the interpretation of elliptical constructions. We will begin by considering the basics of Kempson's LDS-based treatment of ellipsis. We shall examine Kempson's analysis of the interaction of ellipsis with adjuncts, ultimately rejecting some of her assumptions. This will set the stage for a consideration of some rather challenging pronoun-related data.

3.1 Premise Withdrawal and Content Sharing

First consider forms like (44):

(44) Bill knows that John respects Mary. Sue too.

The constructive theory would approach the problem of interpreting such utterances by finding some way of subtracting a piece of the meaning of the left-hand, source clause in such a way that when the meaning of the partial, right-hand, target clause is added, the sum is a whole, well formed semantic representation for a clause. Kempson (1992) proposes to realize this goal with the premise withdrawal rule of conditional proof.

In the following figure, the database s_1 corresponds to the source clause, while s_3 is for the target clause:

(45) Bill knows that John respects Mary. Sue too.

s_1 :

goal: $\alpha : t$

1. **Bill** : e — assume
2. **know** : $e \rightarrow [e \rightarrow t]$ — assume
3. s_2 :
   - goal: $\beta : t$
   - 4. **John** : e — assume
   - 5. **respect** : $e \rightarrow [e \rightarrow t]$ — assume
   - 6. **Mary** : e — assume
   - 7. **respect(Mary)** : $e \rightarrow t$ — MPP; 5, 6
   - 8. **respect(Mary)(John)** : t — MPP; 7, 4

   : e
9. **know**$(s_2 :$ **respect(Mary)(John)**$)$: $e \rightarrow t$ — MPP; 2, 3
10. **know**$(s_2 :$ **respect(Mary)(John)**$)$ **(Bill)** : t — MPP; 9, 1

s_3 :

goal: $\gamma : t$

1. **Sue** : e — assume

s_1 :

goal: $\alpha : t$

2. **Bill** : e — assume
3. **know** : $e \rightarrow [e \rightarrow t]$ — assume
4. s_2 :
   - goal: $\beta : t$
   - 5. **John** : e — assume
   - 6. **respect** : $e \rightarrow [e \rightarrow t]$ — assume
   - 7. **Mary** : e — assume
   - 8. **respect(Mary)** : $e \rightarrow t$ — MPP; 6, 7
   - 9. **respect(Mary)(John)** : t — MPP; 8, 5

   : e
10. **know**$(s_2 :$ **respect(Mary)(John)**$)$: $e \rightarrow t$ — MPP; 3, 4
11. **know**$(s_2 :$ **respect(Mary)(John)**$)$ **(Bill)** : t — MPP; 10, 2

12. λx[**know**$(s_2 :$ **respect**(x)**(John)**$)$ **(Bill)**] : $e \rightarrow t$ — CP; 7, 11
13. **know**$(s_2 :$ **respect(Sue)(John)**$)$ **(Bill)** : t — MPP; 12, 1

The database s_3 consists of the assumption **Sue** : e, which is provided by the input, plus a certain manipulation of the database s_1, from the source clause. One assumption, namely **Mary** : e, is withdrawn from s_1 by the rule of conditional proof, yielding a λ-abstract in the s_3 database. The property so-created is then combined with **Sue** : e to finish off with a formula that accurately represents one of the possible readings of the target clause.

Note that nothing forced us to choose **Mary** : e as the assumption to withdraw from s_1 in the foregoing derivation. It would be equally possible to let the conditional proof rule operate on either of the remaining terms. The results of these alternative derivations are shown in (46). Felicitously enough, both are quite possible readings for the sentence in question.

(46) Alternative Readings for (45)

s_3 :

goal: $\gamma : t$

1. **Sue** : e — assume

s_1 :

goal: $\alpha : t$

⋮ ⋮

11. **know**$(s_2 :$ **respect(Mary)(John)**$)$ **(Bill)** : t — MPP; 10, 2

12. λx[**know**$(s_2 :$ **respect(Mary)**(x)**)(Bill)**] : $e \rightarrow t$ — CP; 5, 11
13. **know**$(s_2 :$ **respect(Mary)(Sue)**$)$ **(Bill)** : t — MPP; 12, 1

goal: $\gamma : t$			
1. **Sue** : e			assume
s_3 : s_1 : goal: $\alpha : t$			
	⋮		⋮
11. **know**$(s_2 :$ **respect(Mary)(John)**$)$**(Bill)** : t			MPP; 10, 2
12. $\lambda x[$**know**$(s_2 :$ **respect(Mary)(John)**$)(x)] : e \to t$			CP; 2, 11
13. **know**$(s_2 :$ **respect(Mary)(John)**$)$**(Sue)** : t			MPP; 12, 1

3.2 Ellipsis and Relative Clauses

At this point we can consider one of the claims about ellipsis interpretation advanced by Kempson (1992). She combines the foregoing approach to ellipsis with the database linking analysis of relative clauses to predict that (47) cannot have a reading where *Mary* corresponds to *Bill*; i.e., the target clause does not have the interpretation 'the woman who likes Mary visited Joan too.'

(47) The woman who likes Bill visited Joan. And Mary too. [=(7)]

goal: $\gamma : t$			
1. **Mary** : e			assume
s_3 : s_1 : goal: $\alpha : t$			
2. $\langle v, \{$**woman**$(\Theta(v)), \forall w[$**woman**$(w) \to w = \Theta(v)]\}\rangle : e$			assume
3. **visit** : $e \to [e \to t]$			assume
4. **Joan** : e			assume
5. **visit(Joan)** : $e \to t$			MPP; 3, 4
6. **visit(Joan)**(v) : t			MPP; 5, 2
	link $u = v$ ↑		
s_2 : goal: $\beta[u] : t$			
7. $u : e$			assume
8. **like** : $e \to [e \to t]$			assume
9. **Bill** : e			assume
10. **like(Bill)** : t			MPP; 8, 9
11. **like(Bill)**(u) : t			MPP; 10, 7
12. λx[**visit(Joan)**(v)] : $e \to t$			Cannot withdraw **Bill** : e from s_1!

The reason for the failure is that one needs to make a λ-abstraction out of the L_1 representation of the matrix clause of the source, which is in the last line of database s_1, whereas the assumption to be withdrawn, **Bill** : e, is contained in an entirely separate database, s_2. The conditional proof rule is simply unable to apply in these circumstances. In this section I shall take issue with Kempson's conclusion.

3.2.1 Counterexamples to Kempson's Claim. While Kempson's analysis does correctly predict the infelicity of the reading in question for (47), I claim that this approach implements an incorrect generalization about why it is that (47) cannot bear the above-described interpretation. If we adopt Kempson's analysis, we are committed to the position that arguments in the target clause cannot be made to correspond to source-clause elements from inside of relative clauses. However, consider in this regard constructions involving *the same goes for* and *the same is true of*.

(48) Mary is extremely fond of John. { The same goes for / The same is true of } Sue.

The right-hand clauses above are syntactically whole, but they nonetheless display the same sort of interpretation as is found with syntactically elliptical structures. Note also that the potential

for strict and sloppy readings for pronouns that characterizes ellipsis is present in the constructions currently under consideration.

(49) John$_i$ admires his$_i$ mother. $\left\{\begin{array}{l}\text{The same goes for}\\ \text{The same is true of}\end{array}\right\}$ Bill.

On the reading where *Bill* corresponds to *John*, the target clause in (49) can mean either 'Bill admires John's mother' or 'Bill admires his own mother.' I submit that to treat the interpretation of such utterances as those in (48) and (49) any differently than one would treat the interpretation of syntactically elliptical structures would be to miss a generalization. Consequently, Kempson's result, if valid, should hold for constructions involving *the same goes for* and *the same is true of* in the same way that it does for usual elliptical utterances. However, when one examines the relevant data, one discovers that the supposedly illicit type of reading is perfectly acceptable.

(50) The negotiators who are representing the United States will fight tooth and nail for their agenda. The same goes for Japan.

(51) The negotiators who are representing the United States want major concessions. The same goes for the European Community.

(52) Anyone who eats sashimi will be ecstatic about the taste. The same is true of tataki.

(53) Anyone who is a friend of Smith's is no friend of mine. And the same goes for that crooked partner of his.

(54) The last person who was a monolingual speaker of Manx is now dead. The same is true of Ainu.

Since Kempson's analysis would kill off any possibility of analyzing (50)–(54) on a par with ellipsis structures, I conclude that it is to be rejected.

This decision is further motivated by data regarding gapping. According to my judgments, all of the following sentences are fine, despite the fact that one or the other of the noun phrases mentioned in the target correspond to an element of the source clause that is embedded inside of a relative or other noun modifier.

(55) John knows someone who was acquainted with Sartre, and Bill Genet.

(56) John frequently referees papers that appear in *Linguistic Inquiry*, and Bill *Language*.

(57) John is responsible for the wall papering in the living-room, and Mary the dining-room.

(58) All those carrying British passports go left, and foreign passports right.

Now we are faced with the task of modifying the analysis so that it is able to account for all the observed data. This will be done in two steps. First we shall liberalize the analysis so that the rule of conditional proof is able to abstract elements out of certain relative clauses. This will allow us to provide appropriate interpretations for data like those in (50)–(58), but it will cause us to lose the explanation of the infelicity of the reading discussed above in reference to (47). Consequently, our second step will be to take measures to restore our ability to rule out that type of undesirable reading.

3.2.2 A Modified Treatment of Restrictive Relatives. Given the current state of the analysis of ellipsis and relative clauses, as developed by Kempson, explicitly mentioned elements from the target clause of an elliptical structure cannot correspond to phrases contained in a relative in the source, because the lexical assumptions used to build the representation of the relative are located in a linked database and are therefore inaccessible to the conditional proof rule. In order to allow the readings in (50)–(58), we shall endeavor to make those assumptions accessible, so as to allow the conditional proof rule to apply. I propose to accomplish this by incorporating databases corresponding to restrictive relative clauses into the databases representing the clauses that most immediately dominate those relatives. The technique to be employed here cannot be applied to non-restrictive relatives, and we shall see below that this state of affairs leads to certain correct empirical predictions.

Let us now consider a new treatment of restrictive relative clauses. First recall the analysis of simple definite descriptions like *the man*. Recall that the semantic contribution of a definite description is a variable with instantiation constraints:

(59) $\langle v, \{\mathbf{man}(\Theta(v)), \forall w[\mathbf{man}(w) \rightarrow w = \Theta(v)]\}\rangle : e$

The variable in (59) must be instantiated by the unique individual that satisfies the predicate **man**. Now consider a definite description with a restrictive relative, such as *the man who teaches Bill*. An expression of this kind does not pick out the unique individual in a given discourse situation that 'is a man,' but rather the unique individual in that situation that both 'is a man' and 'teaches Bill.' Note that the presence of other men in the relevant discourse context need not affect the appropriateness of the *the man who teaches Bill*. If we are to model these facts in the LDS representation, the content of the restrictive relative clause must be made a part of the instantiation constraints on the variable in the representation of the definite description. I propose to do this by inserting an appropriately linked database into the resource label:

(60) $\langle v, \{\mathbf{man}(\Theta(v)), s_1 :$

link $u = v$	
goal: $\beta[u] : t$	
1. $u : e$	assume
2. $\mathbf{teach} : e \rightarrow [e \rightarrow t]$	assume
3. $\mathbf{Bill} : e$	assume
4. $\mathbf{teach}(\mathbf{Bill}) : e \rightarrow t$	MPP; 2, 3
5. $\mathbf{teach}(\mathbf{Bill})(u) : t$	MPP; 4, 1

$, \ldots\}\rangle : e$

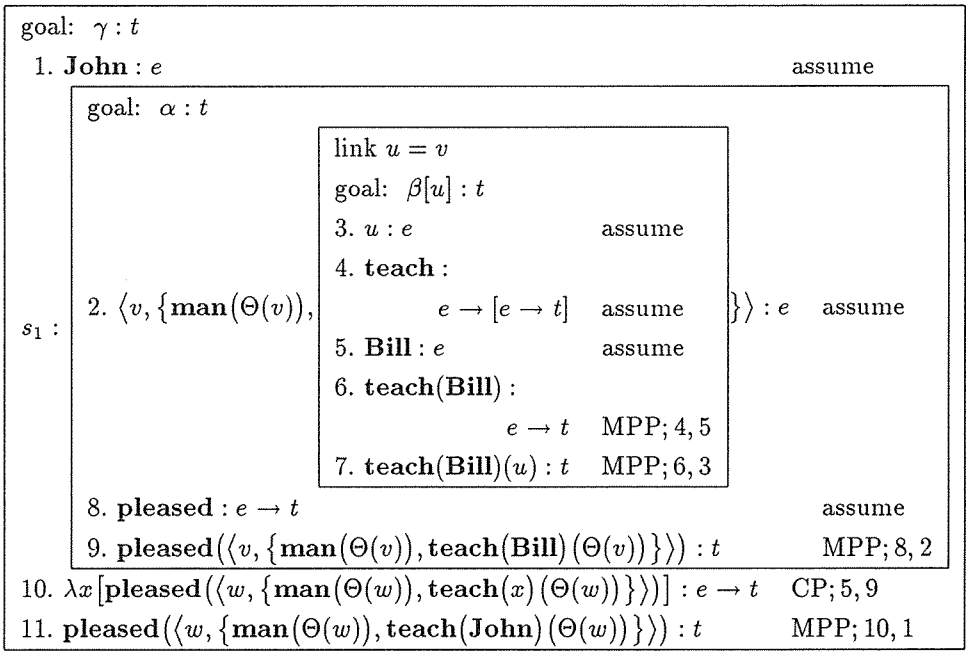
Despite graphic differences, the database in (60) is to be regarded as linked in the usual manner illustrated in previous sections. The form in (60) is to be interpreted as meaning that the L_1 expression derived in the database must be made true by the instantion of the variable which is the content label. Due to the bulkiness of the notation, I have omitted the universally quantified uniqueness constraint from the resource label of (60) in order to save space. However, the reader should assume here and below that such a constraint is implicitly present.

It is essential to note here that only databases corresponding to restrictive relatives are to be included in resource labels. This distinction is semantically motivated. For instance, if the relative in *the man, who teaches Bill* is non-restrictive, then there can be only one man in the relevant discourse situation. This state of affairs is accurately modeled if the content of the non-restrictive relative is omitted from the resource label.

Of course, we cannot simply carry on with the LDS as it currently stands. Rather, the algebra of labels will have to be suitably altered so as to accommodate the new elements to be allowed in the resource label. However, just as in previous areas where modifications of the label albegra were needed, I propose to keep the discussion on an intuitive level and simply assume that the formal definitions have been suitably adjusted to meet our requirements. The processing instructions that accompany relative pronouns in the lexicon will also have to be altered. In the majority of cases, the instruction to shift processing into a new, linked database which receives the information from the subsequent relative clause will have to be accompanied by an *optional* instruction to insert that new database into the resource label of the variable to which the database is linked. Since relative clauses introduced by *that* are exclusively restrictive, the instruction to insert the relative's database into the resource label will have to be *obligatory* with respect to that particular lexical item.

This modification to the analysis of relative clauses not only improves the semantic analysis of restrictive modifiers, but also provides us with a means of admitting *the same goes for* structures and gapping where the target-clause items correspond to elements in relative clauses. Since the databases for restrictive relatives now reside within labels, the assumptions contained in these databases are henceforth accessible for application of the rule of conditional proof in superordinate databases. This parallels the foregoing treatment of complement clauses in (45). Recall that the databases for such clauses were similarly contained in labels. We can now handle interpretations like (61); in a situation where it is known in advance that Bill and John are pupils of two male tutors and that the tutor's satisfaction with their pupils' scholastic progress is under consideration, it is fairly easy to get the reading 'the man who teaches John is pleased' for the target clause.

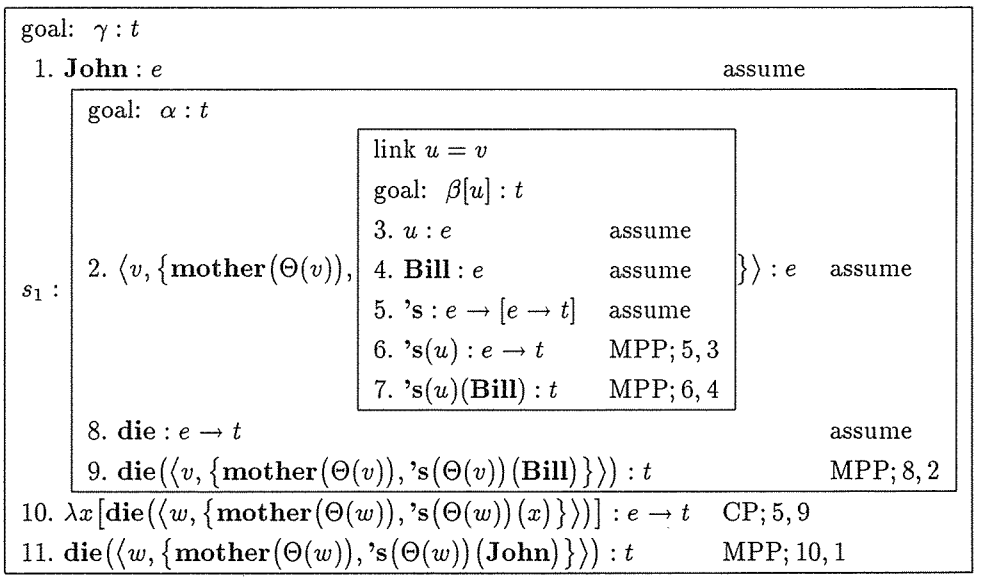
(61) The man who teaches Bill is pleased. The same goes for John.



The structure in (61) is the LDS representation of the target clause of the accompanying utterance. Though not drawn out separately here, s_1 is the LDS representation of the source clause. The database label for the relative clause and the uniqueness constraint on the definite description have been dropped in order to conserve space. Note that the algebra of labels will have to be specified in such a way as to ensure that variables in definite descriptions are changed when abstraction affects their constraints.

Let us also note that a parallel analysis is available for possessives.

(62) Bill's mother died. The same goes for John.



Now consider a possessive form like *John's mother*. I propose to treat possessives as constrained variables, just as in the case of definite descriptions. I will not attempt to discuss the many difficult aspects of the semantics of possessives; let us simply use the predicate 's to represent the complex relation that holds between the possessor and possessee. In any case, we have now achieved our first goal, which was to allow for interpretations where an element of the target could correspond to phrases inside of NP modifiers in the source, as was the case with the data in (50)–(58).

3.2.3 A Brief Look at Non-Restrictive Relatives. Let us now turn our attention for a moment to the question of how ellipsis interacts with non-restrictive relative clauses. Recall that we made a slight modification to the analysis of restrictive relative clauses in order to allow for the readings observed with regard to (50)–(58). However, that change does not affect non-restrictive relatives. Consequently the original treatment that Kempson proposes for relatives in general still applies unchanged to non-restrictive relatives. As a result, Kempson's prediction that elements of the target clause may not correspond to items inside of source-clause relatives is still in force as far as non-restrictive modifiers are concerned. Indeed, this prediction is born out by the data. Consider the contrasts below:

(63) a. The doctor who is treating *John* performs tests every day. The same goes for *Bill.*

b. *Dr. Smith, who is treating *John*, performs tests every day. The same goes for *Bill.*

(64) a. John hates cities that have *polluted air.* The same goes for *polluted water.*

b. *John hates Los Angeles, which has *polluted air.* The same goes for *polluted water.*

The relative clauses in the (b) examples are necessarily non-restrictive, because they modify proper names. Even if it is known that Dr. Smith treats both John and Bill and that Los Angeles has both polluted air and polluted water, the (b) utterances above cannot have interpretations where the two emphasized elements correspond. However, the comparable (a) examples with restrictive relatives clearly allow the readings in question. The analysis of restrictive and non-restrictive relatives presented in this study predicts exactly this contrast. Hence, we may count these observations as evidence in favor of the present approach.

3.2.4 Constraints on Target-Source Correspondence. Now it is time to return to the second of the two problems posed above. If we liberalize the analysis in a manner that allows for the problematic readings in (50)–(58), which featured the expressions *the same goes for* and *the same is true of* as well as gapping, how then does one rule out the same kind of reading in the case of (47), which features a bare argument fragment? I suggest that the proper way to handle (47) will be revealed through examination of the nature of the adverb *too*, which accompanies the bare argument fragment. It will also prove necessary to posit a constraint on the interpretation of VP ellipsis structures. However, let us set the stage for this discussion with a brief survey of the range of possible interpretations available for the various constructions under consideration.

Let us begin with a raw empirical generalization. Among elliptical structures and the expressions involving *the same goes for* and *the same is true of*, one can recognize at least three distinct categories in terms of the constraints that each structure imposes on the selection of an argument from the source clause to abstract with the conditional proof rule. The following data illustrate the division; the first item in (65) should be taken as a source clause, and to it the three different target clauses in (65a–c) are to be applied.

(65) John's lawyer attacked the prosecution.

a. So did Bill.

i. $\Rightarrow$ 'Bill attacked the prosecution.'

ii. $\not\Rightarrow$ 'John's lawyer attacked Bill.'

iii. $\not\Rightarrow$ 'Bill's lawyer attacked the prosecution.'

b. Bill too.

i. $\Rightarrow$ 'Bill attacked the prosecution.'

ii. $\Rightarrow$ 'John's lawyer attacked Bill.'

iii. $\not\Rightarrow$ 'Bill's lawyer attacked the prosecution.'

c. The same goes for Bill.

i. $\Rightarrow$ 'Bill attacked the prosecution.'

ii. $\Rightarrow$ 'John's lawyer attacked Bill.'

iii. $\Rightarrow$ 'Bill's lawyer attacked the prosecution.'

All of the (i) entries in (65) represent interpretations of the corresponding target clauses where *Bill* is interpreted as being correlated with the subject of the source clause. In the (ii) and (iii) interpretations, *Bill* corresponds to the object and the subject possessor, respectively. The cases marked with $\Rightarrow$ are those where my intuitions indicate that the subsequent interpretation is possible, given adequate contextualization; I take the $\not\Rightarrow$ interpretations to be impossible. With respect to the foregoing array of data and various other examples like them, I would propose the following generalizations about the restrictions imposed by the three different constructions under scrutiny.

So did NP. NP *did too.*	NP must be interpreted as corresponding to the subject of the source clause. Neither non-subjects nor elements embedded within the subject are acceptable correspondents for NP.
NP *too.*	NP must be interpreted as corresponding to some argument or adjunct of the source clause. The choice of correspondents must respect a kind of '*A*-over-*A*' constraint, which prevents the selection of any element embedded inside of an argument or adjunct of the source clause.
The same goes for NP *The same is true of* NP	NP is free to correspond to any element of the source clause, whether or not it is a direct argument or adjunct of that clause.

Though omitted from the above table, gapping falls into the last category, as (55)–(58) suggest. Not all of the details listed in the above set of generalizations are motivated by (65). The reader is invited to verify that the following data fill in some of the missing pieces of evidence. The judgments below reflect the possibility of an interpretation where the emphasized elements of the source and target clauses are interpreted as corresponding to each other:

(66) The school lost *John*'s records at the time of the fire.

a. *So did *Bill.*

b. **Bill* too.

c. The same goes for *Bill.*

(67) The courts will try any corporation that discriminates against *minorities.*

a. *So do *women.*

b. **Women* too.

c. The same goes for *women.*

(68) I wrote several letters to *John.*

a. *So did *Bill.*

b. *Bill* too.

c. The same goes for *Bill.*

(69) Over the years I wrote many notes of apology to *John*'s teachers.

a. *So did *Bill.*

b. **Bill* too.

c. The same goes for *Bill.*

(70) I always laugh my head off with *John.*

a. *So does *Bill.*

b. *Bill* too.

c. The same is true of *Bill.*

(71) The D.A. decided to prosecute despite the sworn statement of someone who says he was with *John* at the time in question.

a. *So did *Bill.*

b. **Bill* too.

c. The same goes for *Bill.*

To account for the different patterns of possible interpretations for VP ellipsis, bare argument fragments, gapping, and constructions involving *the same goes for* and *the same is true of*, it will suffice to propose two constraints. One would apply to VP ellipsis, and the other would account for the bare-argument-fragment data. The remaining phenomena represent the full potential for abstraction by the rule of conditional proof when unfettered by constraints.

Now let us sketch out (albeit in slight detail) how one might implement the above constraints in the LDS framework. First we require a way of reigning in the application of the conditional proof rule with VP ellipsis. The constraint will require that the element of the source clause to be abstracted by the conditional proof rule must be the subject. This constraint could be invoked during the processing of the input string when one encounters an auxiliary verb not followed by another verbal construction. The subject may be identified within the source database by means of a mark placed in the resource label of the relevant argument. Notice that this mark is distinct from the 'use me last' indicator. The latter picks out a semantic notion similar to that of 'agent' or 'actor,' in the sense in which these terms have been used in the generativist literature. The mark identifying the subject can be applied to a given term as it is identified in the input scanning process, where the 'use me first' and 'use me last' labels are attached. I will not attempt any more specific proposal than this. Instead I will simply content myself with the observation that identifying the subject of a clause is surely no more difficult a task than determining the proper application of the 'use me first' and 'use me last' labels. Hence, the eventual implementation of the present proposal is no more demanding than the commitment to argument differentiation already undertaken as a fundamental part of LDS-based utterance interpretation.

We have seen approximately how a constraint on VP ellipsis could be imposed, so let us move on to a consideration of how one might limit the interpretations available with bare argument fragments. In fact, I would like to claim that the real source of the constraint is probably not the bare-argument-fragment construction itself, but rather the accompanying adverb *too.*

It was claimed above that the bare-argument-fragment construction obeys an A-over-A-type constraint with regard to selecting source-clause correspondents for the phrase explicitly mentioned in the target. Thus, the source-clause correspondent cannot be selected from within a relative clause, for instance. I would like to emphasize here that the same sort of constraint is at work, even when *too* is used in non-elliptical constructions, such as those below.

(72) John left. Bill left too.

(73) I gave a book to John. I gave one to Bill too.

In these sentences the effect of the application of *too* is to correlate *John* and *Bill.* However, it would be impossible to use *too* to correlate two items that were embedded in relative clauses. The demonstration of this fact is somewhat elusive, because sentences like the following have perfectly acceptable interpretations:

(74) All of the thieves who robbed banks are in jail.
All of the thieves who robbed jewelry stores are in jail *too.*

The only difference between the two sentences above, other than the addition of *too* in the latter one, lies in the juxtaposition of *banks* with *jewelry stores.* However, these noun phrases are inside of relative clauses. Intuitively, though, one may perceive that *too* is correlating not the pair consisting of *banks* and *jewelry stores*, but rather the pair consisting of the larger phrases *all of the thieves who robbed banks* and *all of the thieves who robbed jewelry stores.* The point I wish to make here comes across when the above reading is removed by contextual information. The two sentences in (74) are fine together, as long as it is assumed that the sets of bank robbers and jewelry store

robbers are distinct. Therefore, consider the following, revised version of (74), where the identity of the two sets is explicitly asserted:

(75) All of the thieves who robbed banks are in jail.
All and only the thieves who robbed banks robbed jewelry stores.
Consequently, all of the thieves who robbed jewelry stores are in jail (**too*).

The felicitous use of an utterance of the general form 'x has the property P, and y has the property P *too*' depends on x and y's being distinct. Obviously the unacceptability of (75) results from the fact that no distinct x and y were available to satisfy this basic semantic requirement of *too*. Yet *banks* and *jewelry stores* are clearly different. Therefore, if we were unable to perceive any acceptable reading for (75), it must have been because the only two distinct items in the juxtaposed sentences were inaccessible for comparison with *too*. I therefore hypothesize that *too* may not be used to compare items which are embedded inside of relative clauses.

Now, if the adverb *too* bears an A-over-A-type constraint, there is no need to duplicate the restriction on the bare-argument-fragment construction itself. Consequently the analysis of elliptical structures per se is relieved of responsibility for handling the infelicity of the reading discussed in reference to (47). Therefore, I will not give any more space to the matter here, except to offer a brief comment about how the A-over-A-type constraint might eventually be implemented in a lexical restriction on *too*. Semantically, this adverb is used to emphasize the fact that two things share some property. Let us say that phrases whose designata are asserted to be similar in this manner are 'affected' by *too*. This adverb affects subcategorized arguments and elements of sentential or VP adjuncts, but not items within noun modifiers. Furthermore, *too* may affect items embedded within complement clauses. This generalization is motivated by the previous examples and the ones listed below.

(76) (He knows that) I like my students.
(He knows that) I like my colleagues too.

(77) (He knows that) I have made mistakes in front of my students.
(He knows that) I have make mistakes in front of my colleagues too.

Now, given a database s_i, the items that *too* may affect are distinguishable in the following manner. Subcategorized arguments of s_i will be present within s_i itself. As for sentential adjuncts, under standard Montagovian assumptions these would be immediately present in s_i as well. For instance, *with* in the fragment *leave with John* would be of type $\langle e, \langle\langle e,t\rangle, \langle e,t\rangle\rangle\rangle$.

(78) ... leave with John ...

	⋮	⋮
	leave : $e \rightarrow t$	assume
	with : $e \rightarrow [[e \rightarrow t] \rightarrow [e \rightarrow t]]$	assume
s_i :	**John** : e	assume
	with(John) : $[e \rightarrow t] \rightarrow [e \rightarrow t]$	MPP
	with(John)(leave) : $e \rightarrow t$	MPP
	⋮	⋮

Furthermore, databases representing complement clauses are realized as content labels. Consequently the items which may be affected by *too* can all be found in s_i itself or else in databases accessible from s_i through a series of one or more content labels. In contrast, noun modifiers are linked to variables; at best, restrictive modifiers are included in the resource labels of the variables to which they are linked. Therefore, any eventual constraint on *too* should require that this adverb be barred from affecting anything in a resource label. Although I have no analysis of the semantics of *too* at this time, when such an account is given, the above observations may provide the basis for a constraint that would adequately limit the set of arguments and adjuncts which could be affected by *too*.

3.2.5 TWO DIFFERING POSITIONS. The present section argues that the interaction of ellipsis phenomena and noun modifiers like relative clauses is rather different from the way it is portrayed by Kempson (1992). Kempson's generalization that items in the target clause can never be correlated with elements embedded in source-clause relatives appears to be too strong. I claim that it holds only in cases of non-restrictive modification. Turning from the abstract level to theory-internal matters, I proposed an independently motivated distinction between the treatment of restrictive modifiers and that of non-restrictive ones, which gave rise to a more liberal analysis of the range of possible interpretations for elliptical structures and semantically related constructs. Indeed, this move gave rise to excessively free interpretations in some of the cases that led Kempson to the restrictive view of the interpretation of ellipsis which I have contested here. I proposed that we regard the cases where my analysis allowed too many readings as being the result of parochial constraints on VP ellipsis and the adverb *too*. Hence, with regard to the treatment of the unavailable readings in question, the analytical fulcrum has been moved from the level of theorems about database structure to that of low-level stipulation in the interest of greater empirical accuracy.

3.3 PRONOUNS UNDER ELLIPSIS

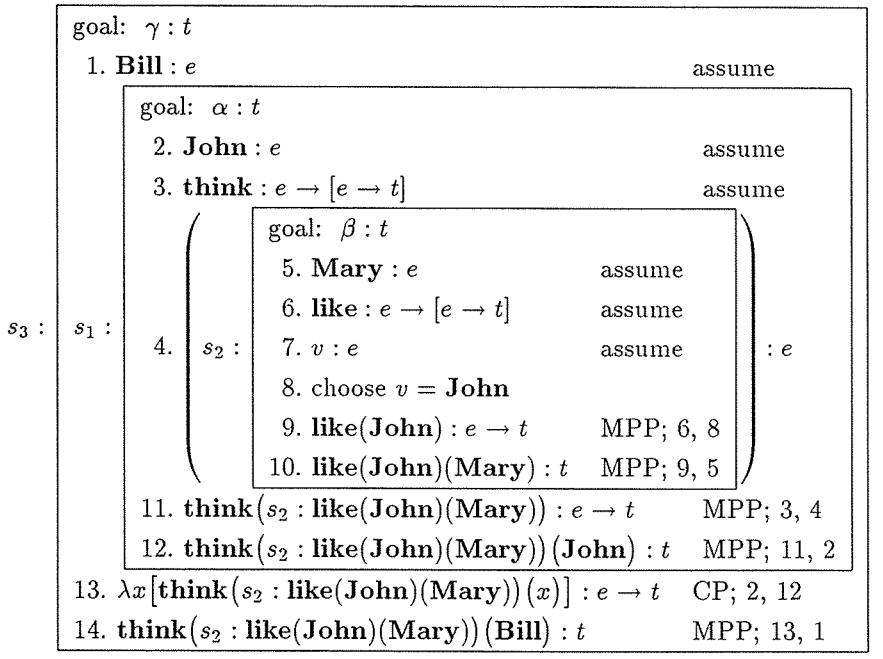
We next consider the interaction of ellipsis and pronouns. We begin by considering the basic LDS approach to strict and sloppy readings. Then we will go on to some challenging data that have made their appearance in the linguistics literature over the last few years.

3.3.1 PRONOUN INTERPRETATION BASICS. Example (79) admits both strict and sloppy readings:

(79) John thinks that Mary likes him, and Bill too.

In other words, the second clause may mean (among other things) either that Bill thinks that Mary likes John—the strict reading—or else that Bill thinks that Mary likes Bill—the sloppy reading. Kempson (1992) notes that either of these readings may be derived using the LDS analysis of elliptical structure. First we display the derivation for the strict reading of (79) in (80).

(80) John thinks that Mary likes him, and Bill too.



The crucial point that distinguishes the strict reading from the sloppy one resides in the λ-abstraction step in line 13. Compare this line with line 13 in the derivation for the sloppy reading, shown in abbreviated form in (81).

(81) SLOPPY READING FOR (79)

s_3 :

goal: $\gamma : t$

1. $\mathbf{Bill} : e$ assume

s_1 :

goal: $\alpha : t$

$\vdots$ $\vdots$

12. $\mathbf{think}(s_2 : \mathbf{like}(\mathbf{John})(\mathbf{Mary}))(\mathbf{John}) : t$ MPP; 11, 2

13. $\lambda x[\mathbf{think}(s_2 : \mathbf{like}(x)(\mathbf{Mary}))(x)] : e \rightarrow t$ CP; 2, 12

14. $\mathbf{think}(s_2 : \mathbf{like}(\mathbf{Bill})(\mathbf{Mary}))(\mathbf{Bill}) : t$ MPP; 13, 1

When one withdraws the assumption $\mathbf{John} : e$ from s_1, one has a choice about how to construct the λ-abstraction. The constant **John** crops up twice in $\mathbf{think}(s_2 : \mathbf{like}(\mathbf{John})(\mathbf{Mary}))(\mathbf{John})$. One may insert the abstraction variable in place of just one instance of the constant or else in place of both, either choice satisfies the exigencies of the conditional proof rule.

Having seen the basics of pronoun interpretation under ellipsis, we shall now go on to an examination of some challenging issues.

3.3.2 GAWRON AND PETERS SENTENCES. Gawron and Peters (1990) report on some data that have proven challenging for standard theories of ellipsis to handle. Consider the following sentence, disregarding any readings where *his* does not refer to *John* in the matrix clause of the left conjunct:

(82) John$_i$ revised his$_i$ paper before the teacher did, and so did Bill.
(Gawron and Peters 1990:122)

There are two readings of this sentence that basically any theory can account for. One is the completely sloppy reading where all the individuals named revised their own papers, and the other is the completely strict one, which has the three participants all revising John's paper. Gawron and Peters point out that (82) is capable of at least one more interpretation, i.e., 'John revised John's paper before the teacher revised John's paper, and Bill revised Bill's paper before the teacher revised Bill's paper.' This reading is beyond the reach of some theories, but Gawron and Peters provide a componential theory that can successfully predict the availability of the last sort of interpretation.

Since the Gawron and Peters approach is componential, it naturally enough deals with (82) by proposing ambiguities in the interpretation of the VP *revised his paper before the teacher did.* Their analysis is couched in situation theory, but the points of interest to us may be rendered in a simple logical shorthand with λ-abstraction. The infix relation R means 'revise,' the 'paper of' function p maps individuals to their associated papers, $\prec$ is the temporal 'precedes' relation, and j, b, and t stand for 'John,' 'Bill,' and 'the teacher' respectively. Gawron and Peters construct VP interpretations by abstracting the subject argument. Hence, the VP *revise his paper* could be rendered as either of the following, with the variable being used to represent the pronoun *his.*

(83) a. $\lambda y[yRp(x)]$
b. $\lambda x[xRp(x)]$

One is not compelled to use a fresh variable for the λ-abstraction, so the possessor may get bound, as in (83b), or it may not, as in (83a). This indeterminacy in variable choice provides for the difference between strict readings (83a) and sloppy ones (83b). When we move on to the complex VP *revise his paper before the teacher did*, either of the two VP meanings in (83) may be shared between the matrix and *before*-clause VPs. Thus, we get the following possibilities.

(84) ... revise his paper before the teacher did

a. i. $\lambda z[\lambda y[yRp(x)](z) \prec \lambda y[yRp(x)](t)]$
ii. $\lambda x[\lambda y[yRp(x)](x) \prec \lambda y[yRp(x)](t)]$
b. $\lambda z[\lambda x[xRp(x)](z) \prec \lambda x[xRp(x)](t)]$

If we assume that x in (84a i) can be instantiated with j—the details of how this is done are unimportant—then we get a completely strict reading, where each person revises John's paper.

(85) a. John revised his paper before the teacher did, and so did Bill.

b. $\lambda z[\lambda y[yRp(j)](z) \prec \lambda y[yRp(j)](t)](j) \wedge \lambda z[\lambda y[yRp(j)](z) \prec \lambda y[yRp(j)](t)](b) \implies [jRp(j) \prec tRp(j)] \wedge [bRp(j) \prec tRp(j)]$

In contrast, if we employ (84b) as the VP interpretation, we would get a completely sloppy reading, where each person revises his or her own paper.

(86) a. John revised his paper before the teacher did, and so did Bill.

b. $\lambda z[\lambda x[xRp(x)](z) \prec \lambda x[xRp(x)](t)](j) \wedge \lambda z[\lambda x[xRp(x)](z) \prec \lambda x[xRp(x)](t)](b) \implies [jRp(j) \prec tRp(t)] \wedge [bRp(b) \prec tRp(t)]$

The third reading, i.e., 'John revised John's paper before the teacher revised John's paper, and Bill revised Bill's paper before the teacher revised Bill's paper,' comes from using (84a ii).

(87) John revised his paper before the teacher did, and so did Bill.
$\lambda x[\lambda y[yRp(x)](x) \prec \lambda y[yRp(x)](t)](j) \wedge \lambda x[\lambda y[yRp(x)](x) \prec \lambda y[yRp(x)](t)](b) \implies [jRp(j) \prec tRp(j)] \wedge [bRp(b) \prec tRp(b)]$

Thus, all of the three readings for (83) that Gawron and Peters discuss are derivable under their analysis.

Dalrymple et al. (1991) report on an analysis of ellipsis that implements a variant of the constructive theory based on the technique of unification. They provide an interesting comparison between their own constructive approach and the componential analysis of Gawron and Peters. The analysis of Dalrymple et al. potentially attributes six readings to the sentence in (83). They are listed in (88) in a pseudo-logical shorthand with an accompanying English description.

(88) a. $[jRp(j) \prec tRp(t)] \wedge [bRp(b) \prec tRp(t)]$
Each person revised his own paper.

b. $[jRp(j) \prec tRp(j)] \wedge [bRp(j) \prec tRp(j)]$
Each person revised John's paper.

c. $[jRp(j) \prec tRp(j)] \wedge [bRp(b) \prec tRp(b)]$
John and then the teacher revised John's paper; Bill and then the teacher revised Bill's paper.

d. $[jRp(j) \prec tRp(t)] \wedge [bRp(j) \prec tRp(t)]$
John and bill both revised John's paper before the teacher revised the teacher's paper

e. $[jRp(j) \prec tRp(j)] \wedge [bRp(b) \prec tRp(j)]$
John and Bill revised their own papers before the teacher revised John's paper.

f. $[jRp(j) \prec tRp(j)] \wedge [bRp(j) \prec tRp(b)]$
John and then the teacher revised John's paper; Bill revised John's paper before the teacher revised Bill's paper.

Dalrymple et al. point out that componential theories, of which Gawron and Peter's is probably the most robust, cannot derive the readings in (88d–f). A little reflection on the Gawron and Peters proposal will reveal that the reason that their analysis cannot handle (88d–f) is that VP meanings are shared, and non-subjects will consequently be identical unless they are bound variables. This makes for a notable difference in predictions between constructive and componential theories of ellipsis. One must then ask which theory is empirically adequate. If the readings in (88d–f) are impossible for (82) and all structurally similar sentences, then these observations provide evidence in favor of the componential theory. However, if any of the interpretations in (88d–f) is valid, then the componential theory will find itself in the position of being unable to account for the full range of readings available in elliptical structures. Admittedly, the readings in (88d–f) may seem somewhat hard to imagine, but I believe that the following example with its supporting context is a fairly natural application of the interpretation in (88d), mutatis mutandis.

(89) (John and Bill had each noticed engine trouble in their respective cars. Bill was fond of bragging about his automotive expertise, and he told John that he would have the problem with his car all figured out before John could even open the hood on his. However, this time Bill was put to shame.) John diagnosed the problem in his car before Bill did, and so, for that matter, did the inexperienced teenager that John lets hang around while he works on his car.

In light of this datum, the constructive theory appears stronger than the componential one.

It is debatable whether the readings in (88e,f) are actually available for (82) or sentences like it. If no such interpretation is possible, then the analysis can be augmented with constraints to rule out the unwanted readings. Even if forced to this step, the constructive theory is in a better position than the componential approach, because the former can clearly be fixed in a manner that accounts for all of the relevant facts. It is not at all clear that the componential theory's coverage can be extended in a manner that would allow it to account for the reading just attributed to (89).

Now let us consider where Kempson's LDS-based approach stands in relation to the debate described above. Here I will briefly demonstrate that the LDS approach predicts the same range of interpretations as did Dalrymple et al.'s analysis. First, I will show in some detail how the LDS analysis can handle the reading in (88c), which was pointed out by Gawron and Peters. The procedure begins by making up a representation for *John revised his paper* and then uses that in constructing the interpretation for *John revised his paper before the teacher did* by means of the conditional proof rule. The latter representation is in turn used in another application of the conditional proof rule to make the representation of the whole sentence.

We begin with the first step described above.

(90) John revised his paper...

	goal: $\alpha : t$	
	1. $\mathbf{John} : e$	assume
	2. $\mathbf{revise} : e \rightarrow [e \rightarrow t]$	assume
s_1 :	3. $\langle u, \{\mathbf{paper}(\Theta(u)), \mathbf{'s}(\Theta(u))(\Theta(v)), \mathbf{male}(\Theta(v))\}\rangle : e$	assume
	4. choose $\Theta(v) = \mathbf{John}$	
	5. $\mathbf{revise}(\langle u, \{\mathbf{paper}(\Theta(u)), \mathbf{'s}(\Theta(u))(\mathbf{John})\}\rangle) : e \rightarrow t$	MPP; 2, 3
	6. $\mathbf{revise}(\langle u, \{\mathbf{paper}(\Theta(u)), \mathbf{'s}(\Theta(u))(\mathbf{John})\}\rangle)(\mathbf{John}) : t$	MPP; 5, 1

For the sake of simplicity, one may think of line 6 in (90) as *John revised John's paper*. Now we must consider the interpretation of the elliptical clause *the teacher did*. To produce an interpretation for this clause, we shall have to borrow information from the representation of the previous, non-elliptical clause in box s_1 in (90). We will use the conditional proof rule to produce a λ-abstraction equivalent to $\lambda x[x$ *revised John's paper*$]$. This abstraction will, of course, result in a strict reading. Specifically, the following derivation results:

(91) ...the teacher did,...

	goal: $\beta : t$		
	1. $\langle w, \{\mathbf{teacher}(\Theta(w))\}\rangle : e$	assume	
	goal: $\alpha : t$		
	2. $\mathbf{John} : e$	assume	
	3. $\mathbf{revise} : e \rightarrow [e \rightarrow t]$	assume	
s_2 : s_1 :	4. $\langle u, \{\mathbf{paper}(\Theta(u)), \mathbf{'s}(\Theta(u))(\Theta(v)), \mathbf{male}(\Theta(v))\}\rangle : e$	assume	
	5. choose $\Theta(v) = \mathbf{John}$		
	6. $\mathbf{revise}(\langle u, \{\mathbf{paper}(\Theta(u)), \mathbf{'s}(\Theta(u))(\mathbf{John})\}\rangle) : e \rightarrow t$	MPP; 3, 4	
	7. $\mathbf{revise}(\langle u, \{\mathbf{paper}(\Theta(u)), \mathbf{'s}(\Theta(u))(\mathbf{John})\}\rangle)(\mathbf{John}) : t$	MPP; 5, 7	
	8. $\lambda x[\mathbf{revise}(\langle u', \{\mathbf{paper}(\Theta(u')), \mathbf{'s}(\Theta(u'))(\mathbf{John})\}\rangle)(x)] : e \rightarrow t$	CP; s_1	
	9. $\mathbf{revise}(\langle u', \{\mathbf{paper}(\Theta(u')), \mathbf{'s}(\Theta(u'))(\mathbf{John})\}\rangle)$ $(\langle w, \{\mathbf{teacher}(\Theta(w))\}\rangle) : t$	MPP; 8, 1	

In (91), a representation for *the teacher* is provided by line 1, then the above described λ-abstraction is produced, and finally, the latter is applied to the former. The result could be paraphrased as $\lambda x[x$ *revised John's paper*$]$(*the teacher*), which is of course equivalent to *the teacher revised John's paper*. Next, let us assume that the representation of *John revised his paper* and that of *the teacher did* are joined by *before*. The details of how this is accomplished are not of interest here, so I merely display the outcome in (92).

(92) John revised his paper before the teacher did, ...

$$\begin{aligned}&\mathbf{before}(s_2 : \mathbf{revise}(\langle u', \{\mathbf{paper}(\Theta(u')), \mathbf{'s}(\Theta(u'))(\mathbf{John})\}\rangle)(\langle w, \{\mathbf{teacher}(\Theta(w))\}\rangle))\\ &\qquad(s_1 : \mathbf{revise}(\langle u, \{\mathbf{paper}(\Theta(u)), \mathbf{'s}(\Theta(u))(\mathbf{John})\}\rangle)(\mathbf{John})) : t\end{aligned}$$

The above expression could be paraphrased as *John revised John's paper before the teacher revised John's paper.* This is obvously the strict reading for *John revised his paper before the teacher did.*

We may now move on to the interpretation of the elliptical clause *so did Bill.* Briefly, this clause provides the lexical assumption $\mathbf{Bill} : e$, but the rest of the interpretation must be borrowed from the form in (92). Consequently, we apply the conditional proof rule to retract the assumption $\mathbf{John} : e$ from (92), and the result is the following λ-abstraction:

(93) $$\begin{aligned}&\lambda y[\mathbf{before}(s_2 : \mathbf{revise}(\langle u''', \{\mathbf{paper}(\Theta(u''')), \mathbf{'s}(\Theta(u'''))(y)\}\rangle)(\langle w, \{\mathbf{teacher}(\Theta(w))\}\rangle))\\ &\qquad(s_1 : \mathbf{revise}(\langle u'', \{\mathbf{paper}(\Theta(u'')), \mathbf{'s}(\Theta(u''))(y)\}\rangle)(y))] : e \rightarrow t\end{aligned}$$

The label in (93) is equivalent to $\lambda y[y$ *revised y's paper before the teacher revised y's paper*$]$. This may then be combined with the lexical assumption $\mathbf{Bill} : e$ by the rule of modus ponens. The result is seen in (94).

(94) ... so did Bill.

$$\begin{aligned}&\mathbf{before}(s_2 : \mathbf{revise}(\langle u''', \{\mathbf{paper}(\Theta(u''')), \mathbf{'s}(\Theta(u'''))(\mathbf{Bill})\}\rangle)(\langle w, \{\mathbf{teacher}(\Theta(w))\}\rangle))\\ &\qquad(s_1 : \mathbf{revise}(\langle u'', \{\mathbf{paper}(\Theta(u'')), \mathbf{'s}(\Theta(u''))(\mathbf{Bill})\}\rangle)(\mathbf{Bill})) : t\end{aligned}$$

The form in (94) may of course be paraphrased as *Bill revised Bill's paper before the teacher revised Bill's paper.* Now we have in hand the representation for the two conjuncts of *John revised his paper before the teacher did, and so did Bill,* and it only remains to put them together. The details of conjunction do not concern us here, so I show only the result of the operation (indeed, (95) is merely the label of the result).

(95) John revised his paper before the teacher did, and so did Bill.

$$\begin{aligned}&\mathbf{before}(s_2 : \mathbf{revise}(\langle u', \{\mathbf{paper}(\Theta(u')), \mathbf{'s}(\Theta(u'))(\mathbf{John})\}\rangle)(\langle w, \{\mathbf{teacher}(\Theta(w))\}\rangle))\\ &\qquad(s_1 : \mathbf{revise}(\langle u, \{\mathbf{paper}(\Theta(u)), \mathbf{'s}(\Theta(u))(\mathbf{John})\}\rangle)(\mathbf{John}))\wedge\\ &\mathbf{before}(s_2 : \mathbf{revise}(\langle u''', \{\mathbf{paper}(\Theta(u''')), \mathbf{'s}(\Theta(u'''))(\mathbf{Bill})\}\rangle)(\langle w, \{\mathbf{teacher}(\Theta(w))\}\rangle))\\ &\qquad(s_1 : \mathbf{revise}(\langle u'', \{\mathbf{paper}(\Theta(u'')), \mathbf{'s}(\Theta(u''))(\mathbf{Bill})\}\rangle)(\mathbf{Bill}))\end{aligned}$$

The above expression means roughly *John revised John's paper before the teacher revised John's paper, and Bill revised Bill's paper before the teacher revised Bill's paper.*

Having seen various details of the analysis, we are now prepared to demonstrate that all of the readings that Dalrymple et al. describe are available in our LDS approach. As described above, the interpretation of the sentence in question is a three stage process. In the table below, $jRp(j)$ represents the first stage, which is common to all readings, the determination of the interpretation of *John revised his paper.* The entries in (96a) and (96b) represent the second stage, i.e., that at which the interpretation of *John revised his paper before the teacher did* is handled. Here the conditional proof rule could be applied so to get either a sloppy reading (96a) or a strict one (96b). We will assume that the some constraint on VP ellipsis of the sort hinted at in the foregoing section prevents a third interpretation where the conditional proof rule abstracts only the object possessor. The (i)–(iv) entries in (96) represent the third stage in the process, where the reading for the whole sentence is determined. Once again the conditional proof rule is free to abstract any number of instances of j, so long as the source-clause subject is among them. In (96a), this gives rise to two different interpretations, and in (96b) there are four possibilities.

(96) $jRp(j)$

a. sloppy: $[jRp(j) \prec tRp(t)]$

   i. sloppy: $[jRp(j) \prec tRp(t)] \wedge [bRp(b) \prec tRp(t)]$

   ii. strict: $[jRp(j) \prec tRp(t)] \wedge [bRp(j) \prec tRp(t)]$

b. strict: $[jRp(j) \prec tRp(j)]$

i. sloppy: $[jRp(j) \prec tRp(j)] \wedge [bRp(b) \prec tRp(b)]$
ii. partially sloppy$_1$: $[jRp(j) \prec tRp(j)] \wedge [bRp(b) \prec tRp(j)]$
iii. partially sloppy$_2$: $[jRp(j) \prec tRp(j)] \wedge [bRp(j) \prec tRp(b)]$
iv. strict: $[jRp(j) \prec tRp(j)] \wedge [bRp(j) \prec tRp(j)]$

Consequently Kempson's LDS approach can account for as many readings as the unification-based analysis of Dalrymple et al. By the reasoning set out above, the LDS-based approach provides a better analysis of pronoun interpretation under ellipsis than does the Gawron and Peters analysis, and by extension most or all other componential analyses. This is so because the LDS-based, constructive approach predicts all of the readings that we have verified to exist, while the competing analysis cannot.

4 Conclusion

The present paper hopefully will make a contribution on two levels. On an abstract plane, in view of the demonstration regarding pronoun interpretations under ellipsis, this work may be viewed as adding to the growing body of research that points to the superiority of constructive theories of ellipsis over componential ones. Of course this study is couched in a particular formalism, the LDS-based approach to utterance interpretation pioneered by Gabbay and Kempson (1992b). Although many of the proposals advanced here are described informally, the logical base furnished by LDS theory should give us the means of providing the various notions employed above with effective definitions. The view of ellipsis phenomena adopted here differs in several points from that presented by Kempson (1992), so the present study should contribute to the debate on how to develop this new branch of linguistic research.