BUILDRS: AN IMPLEMENTATION OF DR THEORY AND LFG

Hajime Wada
Department of Linguistics
The University of Texas at Austin

Nicholas Asher
Department of Philosophy
Center for Cognitive Science
The University of Texas at Austin

ABSTRACT

This paper examines a particular PROLOG implementation of Discourse Representation theory (DR theory) constructed at the University of Texas. The implementation also contains a Lexical Functional Grammar parser that provides f-structures; these f-structures are then translated into the semantic representations posited by DR theory, structures which are known as Discourse Representation Structures (DRSs). Our program handles some linguistically interesting phenomena in English such as (i) scope ambiguities of singular quantifiers, (ii) functional control phenomena, and (iii) long distance dependencies. Finally, we have implemented an algorithm for anaphora resolution. Our goal is to use purely linguistically available information in constructing a semantic representation of discourse as far as is feasible and to forego appeals to world knowledge.

Introduction ; DR Theory

DR theory, versions of which have been developed independently by Hans Kamp (1981) and Irene Heim (1982), has several motivations, but certainly one of the principal ones was to examine the anaphoric properties of definite and indefinite noun phrases. Out of this study emerged a novel theory of indefinites and definites that has provided, among other things, a semantic theory of intersentential anaphora. One of our goals in implementing DR theory for a fragment of English was to investigate how a purely linguistic theory of anaphora might help in the real time processing of anaphors by automated natural language understanding systems.

DR theory has two basic components: the **DRS construction algorithm** and the **correctness definition.** The DRS construction algorithm is a mapping from natural language discourses to DRSs, the semantic representations that the theory posits. The correctness definition maps DRSs into models that (i) provide the truth conditions of one sentence in isolation and (ii) show how the content of one sentence contributes to the content of the antecedent discourse it follows. We have implemented only a fragment of the presently developed DRS construction algorithm.

We want to argue that for philosophical reasons it makes little sense to try to implement on a computer not only the DRS construction algorithm but also the correctness definition. To understand why we hold this view, however, we have to provide at least a very sketchy overview of these two components of DR theory.

The language in which DRSs are described is to begin with quite simple. Its vocabulary consists of a set of individual **reference markers** (x, y, z, x_1, etc.), a set of mental state markers (p, p_1,p_2, etc.), and a set of n-ary **predicates,** for which English nouns, verbs and intersective adjectives will serve fine. We also have certain logical symbols in the DRS language: $\neg$, $\vee$, $\Rightarrow$.

Next, we define **conditions** and DRSs by a simultaneous recursion. (I will use boldfaced letters of the appropriate type as metalinguistic variables for reference markers.)

Definition 1:

1. Suppose that φ is an n-ary predicate and $\mathbf{x_1}, \ldots, \mathbf{x_n}$ are reference markers. Then $\varphi(\mathbf{x_1}, \ldots, \mathbf{x_n})$ is an atomic condition.
2. Suppose $\mathbf{x_1}$ and $\mathbf{x_2}$ are reference markers. Then $\mathbf{x_1} = \mathbf{x_2}$ is an atomic condition.
3. A DRS K is a pair of sets <U, Con>, where U is a set of reference markers and Con a set of conditions.
4. Let K_1 and K_2 be DRSs and let $\mathbf{p}$ be a mental state reference marker. Then $\neg K_1$, $K_1 \vee K_2$, $K_1 \Rightarrow K_2$, and $\mathbf{p}{:}K_1$ are conditions.

This framework yields a treatment of indefinite and definite noun phrases that has made an important contribution to understanding the anaphoric and "pseudo referential" behavior of indefinites. Let us first briefly sketch the theory's treatment of indefinites. When processed by the DRS construction algorithm, a singular indefinite introduces into a DRS a reference marker that functions essentially as a free variable. This reference marker can be identified with other reference markers that are introduced by anaphoric pronouns at potentially unlimited distances from the original indefinite NP. Indefinites get their existential force by the way a DRS is assigned truth conditions.

To take a simple example that uses only indefinites and anaphoric pronouns, the DRS construction algorithm yields the DRS in (2) for

(1) A man loves a woman. She is beautiful.

(2)

<1> x,<2> y, <5> z
<0> [A man loves a woman]
<1> [loves a woman(x)]
<2> woman(y)
<3> loves(x, y)
<4> [She is beautiful]
<5> z = y
<5> [is beautiful(z)]
<6> beautiful(z)

The informal interpretation of (2) is that there are three objects (corresponding to x, y and z) that have the respective properties ascribed to x, y and z in the conditions <1> - <6>. The numbers to the left of the reference markers and conditions indicate how the construction algorithm might proceed from step to step in a top down algorithm on some sort of standard parse tree like an LFG c-structure. Initially, we begin with the unanalyzed sentence within a DRS K. The subject N(oun) P(hrase) node in the parse tree introduces a condition into the DRS, the condition man(x)', into Con_K and a reference marker x into U_K. That exhausts the content of the indefinite noun phrase. The

V(erb) P(hrase) node introduces an intermediate step in the DRS that is further broken down by the algorithm as it goes down the parse tree; we shall write this intermediate step as an as yet not fully analyzed condition -- 'loves a woman(x)'. The reference marker introduced by the NP already processed (the subject NP) is an argument of that condition. The unanalyzed condition is then broken down by the algorithm as it goes down the parse tree; the NP that is a consituent of the VP introduces another reference marker and a condition 'woman(y)' into K; and finally, the verb itself introduces a condition into Con_K.

After step <3>, the algorithm has finished with the first sentence in (1). It now processes the second sentence, using the background DRS constructed from the first as a context for the interpretation of the new input. The subject noun phrase of the second sentence is simply an anaphoric pronoun. Because the pronoun is an NP, the algorithm requires that it introduce a new reference marker into the DRS. Because the pronoun is anaphoric, the reference marker it introduces must be linked to some already introduced reference marker in U_K. Thus, U_K provides a set of contextually "discourse individuals" (objects that have been talked about in the discourse) that can be referred to by NPs in subsequent discourse. Once, the algorithm is finished with 'she', the rest of the second sentence is processed in the same sequence of steps used to process the first.

Our implementation in general follows the left to right process described above. Using f-structures as inputs to the construction algorithm allows us to bypass the steps that are enclosed within the square brackets, since we have already available in the f-structure the basic predicate argument structure of this simple sentence. The f-structures play a more important role in more complex sentences like those involving relative clauses; there they make transparent the way in which the predicate argument structure of subordinate clauses fits together with the predicate argument structure of the main clause.

Accessibility

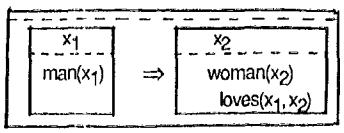
The simple example in (1) already introduces the problem of anaphoric coreference. DR theory, like all linguistic theories, provides constraints on what are the possible, antecedent, coreferential NPs for an anaphoric, pronominal NP. We shall say that a reference marker **u** introduced by an antecedent NP can be linked anaphorically to a reference marker **v** introduced by a pronominal NP just in case **u** is *accessible* to **v**. **u** is accessible to **v** just in case <**u**,**v**> belongs to the transitive closure of the relation that is defined as follows: a) **u** has already been introduced into U_K prior to the introduction of **v**, b) $u \in U_{K'}$, and there is a K" such that K' $\Rightarrow$ K is a condition in K", c) K occurs as a component of some condition in K' and **u** has already been introduced into $U_{K'}$ prior to the introduction of the condition containing K.[1] Our implementation uses this notion of accessibility to constrain the process of anaphora resolution.

While singular indefinite NPs simply introduce, when processed by the construction algorithm, reference markers and atomic conditions into the DRS, "quantificational" NPs (those involving determiners like 'every', 'each', and also 'many') will introduce logically complex conditions (see Kamp, (1981), Frey & Kamp (1985) for details).

(3) Every man loves a woman

yields on the "default" left-right scope reading the following DRS:

(4)



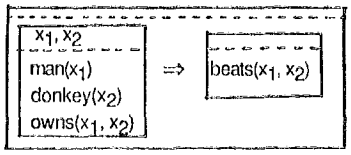
As we shall see shortly, the truth conditions of such a DRS are essentially this: pick any man; then there's a woman that loves him. The DRS in (4) differs from the one that is the result of processing the first sentence in (1), in that the determiner 'every' makes a decidedly different contribution from a pure indefinite. 'Every' introduces a logically complex structure between two DRSs; it corresponds in first order logic to a universal conditional.

Our implementation also countenances relative clauses and the truth functional sentential conjunctions that correspond to the DRS connectives already mentioned. Given the original design of the algorithm reproduced in our implementation and the notion of accessibilty, the following two sentences turn out to have almost exactly the same DRS:

(5) Every farmer who owns a donkey beats it.
(6) If a farmer owns a donkey he beats it.

The DRS in (7) is what (5) yields. The DRS for (6) is almost identical, except that the DRS in the consequent of the conditional $\Rightarrow$ contains two new reference markers in its universe **z** and **w** and the conditions $z = x_1$ and $w = x_2$.

(7)



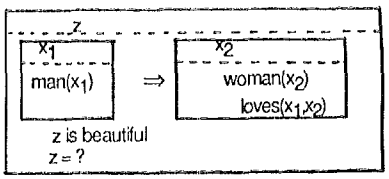
The construction algorithm thus shows how to arrive by a completely effective means at the same DRS and the same truth conditions for (5) and (6) and in so doing solves a longstanding puzzle concerning the "donkey sentences." See Kamp (1981), Heim (1982).

So far we have seen only left-right scope assignments to sentences with indefinites and true quantificational NPs. But of course, there are other possible scope assignments, which our implementation also generates. We generate the left-right scope assignment as a default, but our implementation can generate the other scope assignments as well. Sometimes these are needed in resolving anaphoric links. For instance, consider the discourse in (8):

(8) Every man loves a woman. She is beautiful.

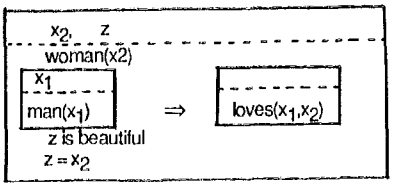
On the default left-right scope assignment, (8) yields:

(9-a)



which is a DRS that is incoherent; z cannot be identified with a reference marker, since neither x_1 nor x_2 are accessible to z. If 'a woman' takes wide scope, however, an anaphoric link is possible, and it is one that our implementation finds:

(9-b)



Definite noun phrases, which include definite descriptions and directly referential expressions like proper names, demonstratives, and indexicals, have a quite complex role in DR theory. All definites generate **adequacy** or **felicity** conditions for discourses containing them. When a definite NP

is used in a discourse, the discourse will be adequate if and only if the recipient has sufficient information given by the context and background assumptions to link the reference marker introduced with some contextually available individual and/or some reference marker already introduced in the discourse. This thesis is developed and defended at length in Heim (1982). If such a link is not possible, then the discourse as a whole may lack a determinate truth value. The intuition behind this requirement is that whenever the speaker correctly employs a definite NP, he has a "definite individual in mind" that the recipient must in at least some minimal way be able to isolate.

These claims lead to a special role for definites in the construction algorithm. They always have in effect wide scope over those logical operators that serve as barriers to anaphoric links, since their denotation must, if the felicity conditions are to be accomodated, already have been introduced in the discourse. To take an example, consider the case of a proper name. Insofar as a proper name is a definite, the recipient of an utterance containing one must have some information enabling him to link the reference marker introduced with the appropriate, contextually available individual. The sort of information at issue may be often quite minimal and the set of contextually available individuals quite large (it includes at least potentially everything that can be named). But once this link is made, the logical barriers to further anaphoric links between the reference markers introduced by name and those introduced by anaphoric pronouns that underlie the constraint of accessibility have no effect. That is, the reference marker introduced by a proper name is always accessible to a subsequently introduced reference marker. The reason for explaining the way definites work is to make a case for distinguishing reference markers introduced by definites from those introduced by indefinites as we have done in our implementation.

Correctness Definition

The model theory or **correctness definition** for the extensional fragment surveyed is very simple. We define a *model for* the DRS "language" to be an ordered pair <**D**, []>, where **D** is a set of entities (the "domain" of the model), and [] an interpretation function that takes n-ary predicates into sets of n-tuples in $\wp(\cup(D^n))$. Further we define an **embedding function** f for a DRS K in a model M to be a map from reference markers in U_K into the domain of M. We will also define an **extension** of an embedding function f to an embedding function g for a DRS K_1 to be the function: g: (Dom(f) $\cup$ U_{K1}) $\rightarrow$ **D**. We shall abbreviate 'g extends f to an embedding of K as '$f \subseteq_K g$'. We now define, again by simultaneous recursion, the notions of a **proper embedding** of a DRS K in the model M and the **satisfaction** of a condition C in a DRS K in the model M by an embedding function. I will abbreviate 'f is a proper embedding of K in M' by '$[f,K]^M = 1$' and 'the model M satisfies C under an embedding function f for K' as $M \models_{f,K} C$'.

Definition 2: Let $x_1, \ldots, x_n$ be reference markers, ψ an n-ary DRS predicate, K, K_1 and K_2 DRSs, and let $[\psi]^M$ be the extension of ψ in M. Then

1. If φ is an atomic condition of the form $\psi(x_1, \ldots, x_n)$, $M \models_{f,K} \varphi$ iff $\langle f(x_1), \ldots, f(x_n)\rangle \in [\psi]^M$.

2. Ifφ is an atomic condition of the form $x_1 = x_2$, $M \models_{f,K} \varphi$ iff $f(x_1) = f(x_2)$.

3. If $\varphi \in Con_K$ is a complex condition of the form $\neg K_1$, then $M \models_{f,K} \varphi$ iff $\neg \exists g \supseteq_{K1} f\ [g, K_1]_f^M = 1$

4. If$\varphi \in Con_K$ is a complex condition of the form $K_1 \vee K_2$, $M \models_{f,K} \varphi$ iff $\exists g \supseteq_{K1} f\ [g, K_1]_f^M = 1 \vee \exists g \supseteq_{K2} f\ [g,K_2]_f^M = 1$

5. If $\varphi \in Con_K$ is a complex condition of the form $K_1 \Rightarrow K_2$, then $M \models_{f,K} \varphi$ iff $\forall g \supseteq_{K1} f\ ([g,K_1]_f^M = 1 \rightarrow \exists h \supseteq_{K2} g\ [h,K_2]_g^M = 1)$

6. $[f,K]_g^M = 1$ iff f is an embedding function such that: (i) $g \subseteq f$; (ii) for every condition ξ in Con_K $M \models_f \xi$; (iii) if K has an external anchor A, then $A \subseteq f$.

7. $[f,K]^M = 1$ iff $[f,K]_\Lambda^M = 1$ where Λ is the empty function.[2]

A DRS K is *true* in a model M just in case there is a proper embedding of K in M.

From the standpoint of the theory of information processing, a DRS represents the result of a recipient's processing of a verbal input. The DRS then as it stands captures the information that the recipient has gleaned from the sentence. The correctness definition evaluates that information content. The two components of DR theory have thus distinctly different tasks: anaphora resolution and other phenomena like scope disambiguation that are necessary for discourse understanding must take place at the level of DRS construction; the assignment of truth conditions is provided by the correctness definition. We want to emphasize that the mapping characterizing the correctness definition is not something that the recipient in general has access to or can construct. For a knowledge of such a mapping involves at least on occasion the knowledge of what are the denotations of directly referential expressions (since these provide the constraints that we have called external anchors on the embedding functions); and as almost two decades of philosophical argument have made plain, the recipient of a discourse containing directly referential expressions need not and generally does not know in any interesting sense what the denotations of those expressions are in order for the discourse to be comprehensible to him. In general, we know that we must distinguish between the truth conditional content of an expression and the content of the expression that is available to the interpreter. This is a thesis that follows directly from accepting the basic principles of the theory of direct reference, which DR theory at least to some extent incorporates. So insofar as we wish to model or to mimick the way humans process verbal inputs, then we must limit ourselves in implementations of natural language understanding to the information provided by the DRS construction algorithm.

In effect this is a useful constraint, for it forces the DR theoretician to make a principled division of labor between these two components of the theory. It also allows us to provide a much more clean cut approach to the implementation of semantic theories than is possible with theories that do not invite this sort of division. For we are able to separate tasks of discourse understanding, which, one feels, ought to be tasks that can be accomplished by means of effective algorithms, from the evaluation of a discourse at a set of indices-- which in general has no effective solution. This may be only a small and rather obvious, philosophical point, but it is one which is worth keeping in mind when one is trying to come upn with tractable analyses of various aspects of natural language understanding.

The Actual Implementation : BUILDRS

The implementation we have constructed has three distinct modules-- an LFG parser, a DRS constructor, and finally an anaphoric resolution component. The inputs to BUILDRS are multisentential discourses, and the outputs are scope disambiguated semantic structures, i.e., DRSs, with reference markers introduced by anaphoric pronouns identitifed with the appropriate reference marker introduced by some previously processed NP.

The LFG parser of the kind used in BUILDRS has already been discussed in detail by Frey and Reyle (1983), Frey (1985) and Reyle (1985). We have agumented their LFG parser to handle compound sentences (those containing conjunctions like 'and', 'or' or 'if. . ., then'). The parser provides a separate f-structure for each of the sentential components of a compound sentence. These then serve as arguments to connectives. We also allow for the possibility of several scope assignments, when more than one operator is involved. We have also augmented the parser by attaching to each value of the specifier in an f-structure a unique identifier; in view of the fact that the LFG parser is a front end to our DRS maker, the identifiers we have chosen are new reference markers (i.e., reference markers that have not already appeared in previously processed discourse).

Once the parser has finished its task, the DRS constructor begins its work. The overall structure of the constructor is similar to that of the informal algorithm discussed earlier and also is based on the work of Frey (1985). We first translate the semantically relevant entries in SUBJ into DRS conditions and partial DRS structures, then those in OBJ and finally PRED of the f-structure as a whole (the main verb) is translated into a DRS condition. The translations for the lexical items in an f-structure are stored in a separate database and can be easily augmented as the grammar and the lexicon grow. The translations of the determiners or specifiers in an f-structure yield structures that we shall call *partial DRSs*. In the fragment treated by BUILDRS[3], we deal only with singular determiners; 'a' (or its equivalents 'some' and the like) and 'every' introduce different partial DRSs. They have the form of a triplet. In the case of 'every' it is <**x**, λPλQ, ([] ⇒(([**x**],[P]), ([],[Q]))>, where **x** is the reference marker introduced by the specifier in the f-structure, P and Q are variables for sets of properties (i.e. sets of conditions), and the structure ([] ⇒([**x**],[P]), ([],[Q])) is the schematic form of a DRS. In the case of 'a' it is <**x**, λPλQ, ([**x**],[P, Q])>. The translations of a common noun phrase are also triplets, but they do not contain abstraction over properties or property sets. Instead, they supply the properties that are to be filled in for P and Q. So, for instance, the translation for the predicate 'man' is the triple <λx, 0, [man(x)]>. Again following Frey (1985), we shall call such triples **predicative DRSs.** We have a special translation for the main predicate of the sentence's f-structure; it is an ordered triple of the form <0, 0, β(X, Y)>. X and Y are to be filled in eventually by the reference markers introduced by the specifiers in SUBJ and OBJ respectively. The translations of the entries in the f-structures are then combined together by means of a process called **conversion**. Conversion is like the application of a λ-abstract to an argument. Following the general path of the construction algorithm, we begin by introducing a partial DRS with the specifier of SUBJ. Suppose for example that SUBJ contains 'every' as a specifier. Its translation is the partial DRS (i) <**x**, λPλQ, ([] ⇒(([**x**],[P]), ([], [Q]))>. The common noun phrase in SUBJ (the head noun + any modifiers) yields a predicative DRS, which will contain a complex property if the common noun phrase is itself complex-- i.e. contains modifiers like possessives or relative clauses. So in general it will be of the form (ii)<λx, 0, (U_{CN}, CON_{CN}[x])>, where U_{CN} is the set of reference markers and CON_{CN}[x] is the set of conditions derived from the common noun phrase and where 'CON_{CN}[x]' denotes the fact that at least one condition in CON_{CN} contains 'x' as an argument. Converting (i) with (ii) yields the partial DRS <**x**, λQ, ([] ⇒(([**x** ∪ U_{CN}],[CON_{CN}[x/**x**]), ([], [Q]))>. '[x/**x**]' denotes the replacement of every occurrence of 'x' in the conditions in CON_{CN} with the reference marker **x**.

The process of conversion turns to OBJ and processes it in the same way. This yields another partial DRS that, like the one above, contains a property abstract,e.g., <**y**, λQ, φ>. The first strategy is to construct a DRS exhibiting the left-right scope of quantifiers as they occur in the discourse. So in this case, we first combine the partial DRS from OBJ with the translation of the main predicate, which is of the form <0, 0, β(X, Y)>. The structure <**y**, λQ, φ> combines with <0, 0, β(X, Y)> to yield a *complete* DRS of the form <0, 0, φ'(X)>. φ'(X) is the result of the conversion of the partial DRS φ with the predicative DRS <0, 0, β(X,**y**)>. Finally, conversion is applied again using the partial DRS derived from SUBJ, yielding another complete DRS and the desired result. To get alternative scope assignments for quantifiers, the program backtracks and tries to do conversion in a different way.

To take an example, let us see how the DRS constructor would handle the sentence, 'every man loves a woman.' The output of the parser yields the structure:

(10)

$$\begin{bmatrix} \text{SUBJ} & = & \begin{bmatrix} \text{SPEC} = \text{'every'} \\ \text{GEN} = \text{masc} \\ \text{PRED} = \text{'man'} \\ \text{NUM} = \text{sg} \end{bmatrix} \\ \text{OBJ} & = & \begin{bmatrix} \text{SPEC} = \text{'a'} \\ \text{GEN} = \text{fem} \\ \text{NUM} = \text{sg} \\ \text{PRED} = \text{'woman'} \end{bmatrix} \\ \text{PRED} & = & \text{'love <(SUBJ)(OBJ)>'} \end{bmatrix}$$

The DRS constructor yields the partial DRS (i) <u_1, λQ, ([] ⇒ (([u_1],[man(u_1)]), ([],[Q]))> for SUBJ and the partial DRS (ii) <u_2, λQ, ([u_2], [woman(u_2), Q])> for OBJ. The translation of the main PRED of the whole f-structure is: (iii) loves(X,Y). Conversion of (ii) with (iii) yields the complete DRS (iv): <0, 0, [u_2] [woman(u_2), loves(X, u_2)]>, and the conversion of (i) with (iv) yields < 0, 0, ([] ⇒(([u_1], [man(u_1)]), ([u_2], [woman(u_2), loves(u_1, u_2)]))))>, which is the desired result. To get the alternative scope reading, the successive conversion of (iii) with (i) and then with (ii) yields the DRS: <0, 0, ([u_2],[woman(u_2), ⇒ (([u_1],[man(u_1)]), ([],[loves(u_1, u_2)])])>.

This is the basic part of the DRS constructor. The two embellishments we have made to this basic part concern the treatment of relatives and possessives. The treatments of possessives and relatives are quite similar, so we shall describe here just one-- the treatment of relative clauses. The syntactic treatment of an NP containing a relative clause of the form 'DET N that S' yields following kind of f-structure:[4]

(11)

$$\begin{bmatrix} \text{SPEC} = \text{Det-}\mathbf{x} \\ \text{PRED} = \text{N} \\ \text{RADJ} = \begin{bmatrix} \text{SUBJ} = \text{pro-}\mathbf{x} \\ \text{OBJ} = \varphi \\ \text{PRED} = \beta \end{bmatrix} \end{bmatrix}$$

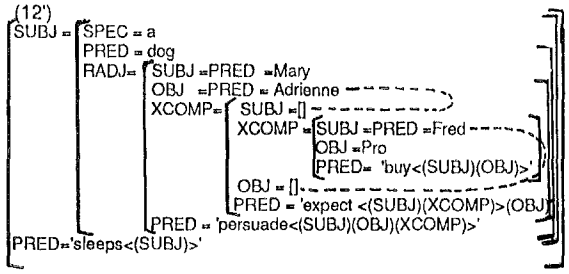
Note that pro carries an identifier identical to that of the specifier of the head.

Suppose that the translation of the specifier introduces a new reference marker **x**. Then the translation of 'pro' yields the following partial DRS: (i) <**x**, λP, ([],[P])>. Since that is all that is in the SUBJ position of RADJ, we get (i) as the partial DRS associated with SUBJ. Now we use conversion to construct the partial DRS for φ (call this (ii)). To get the default scope

assignment we convert (ii) with the translation for β and then convert that result again with (i). The result is a *quasi-complete DRS* of the form (iv) <0, 0, (U_{RADJ}, Con_{RADJ})>, where U_{RADJ} is the set of reference markers and Con_{RADJ} is the set of conditions derived from the processing of RADJ. Quasi-complete DRSs are distinguished from complete DRSs, in that they are derived from f-structures containing 'pro'. To use (iv) in building up the partial DRS for the main NP, the program converts the structure for RADJ (iv) into a λ-abstract term-- viz. (v) <0, λP, (U_{RADJ}, [Con_{RADJ}, P])>. We are now able to combine (v) with the translation of PRED to get the translation of a complex common noun phrase-- viz. (vi) <λx, 0, (U_{RADJ}, [Con_{RADJ}, N(x)])>. Now this predicative DRS may be combined with the partial DRS produced by the specifier in the usual way to yield the partial DRS for the complete NP.

Processing relative clauses in this way allows us to handle long distance dependencies. Consider the sentence,

(12) A dog that Mary persuaded Adrienne to expect Fred to buy sleeps.

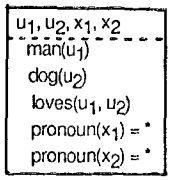


The DRS constructor begins with the specifier of the subj f-structure in the usual fashion and then begins to construct the common noun phrase from the inside out. So then we begin with the innermost XCOMP. Suppose that the specifier introduces a reference marker **x**. How, we have to ask, is the translation of an identification between two f-structures to be done here? We suppose that such an identification is very similar to pro and that it should be translated in exactly the same way. So the translation and conversion of the innermost XCOMP yields the quasi-complete DRS <0, 0 ([f], [buy(f,**x**)])>. Now, we put this together with the translations of the next innermost SUBJ and OBJ and the translation of the predicate 'expect' by conversion to get the following quasi-complete DRS <0, 0, ([a, p], [expect(a,p), p: ([f], [buy(f,**x**)])])>. In a similar fashion we handle the construction of the entire RADJ; translation and conversion yield once again the quasi-DRS <0, 0, ([m, a, q], [persuade(m, a, q) q: ([p], [expect(a,p), p: ([f], [buy(f,**x**)])])])>. Now we convert the result into a λ-abstract since we are done with RADJ and we can now form the complex noun phrase translation in the familiar fashion.

Anaphora Resolution

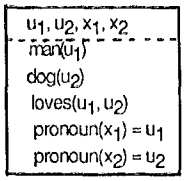
The pronoun resolver is the last module in the program and perhaps the most complex. It operates on a complete DRS as an input and searches for conditions of the form 'pronoun(**x**) = * '. The reason we need to operate on complete DRSs is that the constraints on anaphoric relations imposed by the accessibility relation require that the logical structure of the sentence be determined. The scope of various quantifiers and truth functional operators, however, is only fully determined at the level of complete DRSs. Once a DRS for a discourse D has been constructed, we go back and examine it-- looking for pronouns. We also construct a database in which all the reference markers introduced in the DRS for D are stored in a tree structure, the transitive closure of which defines the accessibility relation on the universe of the DRS for D. Associated with each reference marker in this database are the gender and number of the NP that introduced it. When we come to a reference marker introduced by a pronoun, it will occur on a certain node n_m in the tree. The reference markers accessible to it are all those on nodes n_j such that there is a path from the root to n_m passing through each n_j. At this point the program searches back to find the first available reference marker whose associated gender and number agrees with the gender and number of the pronoun, and which satisfies certain other constraints that the pronoun might have. One constraint is that a reference marker **x** in a DRS K introduced by a non-reflexive pronoun cannot be identified with a reference marker **y**, if K already contains a condition of the form $\varphi(\mathbf{x}, \mathbf{y}, \mathbf{z_1}, \ldots, \mathbf{z_n})$. The opposite is true for a reference marker introduced by a reflexive pronoun. If these constraints are met, the program then replaces the condition 'pronoun(**x**) = *' with a condition of the form **x** = **y**, where **y** is the reference marker that was found to match **x**. Let us take a look at a typical example.

(13) A man loves a dog. He feeds it.



The pronoun resolver now takes over and produces a tree structure of available dicourse referents as it goes through the CON list of the DRS. In this case we have a simple tree of the form u_1, <sing, masc>--- u_2 , <sing, neut>--- x_1, <sing, masc> --- x_2, <sing, neut>. As it is constructing such a tree, it also looks for reference markers introduced by pronouns. The condition 'pronoun(x_1) = *' tells it that x_1 is such a reference marker. At this point, it now searches back for the reference marker in the database with the appropriate number and gender. After rewriting the conditions, the pronoun resolver prints out the DRS:

(13')



There are, however, several problems with the resolver as described so far. One has to do with the anaphoric behavior of definites. Many English speakers find (14) acceptable but (15) bad.

(14) If Mary likes every one who likes John$_i$, then she likes him$_i$.

(15) *If Mary likes every one who likes someone$_i$ then she likes him$_i$.

Definites like proper names seem to be available for anaphoric linkage despite the presence of logical barriers to anaphora. But given our description of the pronoun resolver, we have not said anything that would distinguish the anaphoric behavior of definites as opposed to indefinites. We will follow the suggestion of Kamp (1983) and treat definites as having wide scope over the logical barriers to anaphora available in the present fragment.[5] Thus, while the program constructs the accessibility tree of reference markers, it places each reference marker introduced by a proper name or other definite at the root of the tree as well as in its normal position. This creates a certain amount of duplication abut allows us to get the right reading for (14) while still getting the preferred "most recent NP" readings first for the majority of anaphoric discourses. Thus, our program succeeds in finding the right DRS for (14) but predicts (15) to be bad. It also predicts that a sentence like (16) will fail, which seems marginal to at least some English speakers.

(16) ? If John likes her$_i$, Fred likes Mary$_i$.

There are still more complexities, however, to the pronoun resolver. Though presently not implemented, we see a need to distinguish the resolution strategy for pronouns that occur in subordinate clauses from the standard one. The need to do this is made evident by the apparent acceptability of Bach-Peters type sentences with indefinites like the one in (17):

(17) A man$_i$who hardly knows her$_j$ loves a woman$_j$ who scorns him$_i$.

The processing story that must be told to make (17) acceptable is quite complex. For instance, it cannot be that relative and other subordinate clauses are simply not processed until the main DRS is already completed. Though this processing strategy would in conjunction with our pronoun resolver predict (17) to be good, it would also predict the marginal (18) as equally acceptable:

(18) ? A man who hardly knows her$_j$ loves Mary$_j$.

Pronouns in subordinate clauses seem to pose an additional complication for the pronoun resolver. We believe pronouns in subordinate clauses should be handled slightly differently than those occurring in main clauses. Like other pronouns, when the resolver finds a pronoun in a subordinate clause α it attempts to find the appropriate antecedent. But if it fails, it leaves the pronoun unresolved until it has processed the rest of the DRS. If it finds a condition later on that is linked to the NP in which α occurs, then it will try again to find antecedents for all the unresolved pronouns in α. This sort of strategy would make (17) acceptable but not (18), which is what is desired. We hope to incorporate this into the pronoun resolver soon.

Conclusion

The pronoun resolver is an attempt to take the resolution of pronouns as far as is possible on a purely grammatical level. We realize that there are some cases where resolution depends on world knowledge and guesses about the speaker's intentions and the like. But using world knowledge is often computationally expensive. Our goal was to develop a program that would minimize appeals to world knowledge in building a semantic representation of the content of a discourse and would instead use to maximum advantage the information that can be gleaned from the words themselves.

References

Asher, N. (1982) "Truth Conditions and Semantic Competence: Toward a Theory of Linguistic Understanding," Ph.D. dissertation, Yale University.

Asher, N. (forthcoming) "Belief in Discourse Representation Theory," Journal of Philosophical Logic.

Chierchia & M. Rooth (1984) "Configurational Notions in Discourse Representation Theory", in C. Jones & P. Sells, eds. Proceedings of N.E.L.S 14, Amherst GSLA (1984) pp. 49-63.

Fenstad, J.E., P.K. Halvorsen, T. Langholm and J. van Benthem (1985) "Equations, Schemata and Situations: A framework for linguistic semantics", CSLI Report, Stanford University.

Frey, W. (1985) "Syntax and Semantics of Some Noun Phrases" , in Laubsch (ed):Proceedings of the German Workshop on Artificial Intelligence 1984.

Frey, W. & H. Kamp (1985) "Distributive and Collective Plurals", Talk Presented at University of Texas Conference on Syntax and Semantics, March 1985.

Frey, W., and U. Reyle, (1983) "A PROLOG Implementation of Lexical Functional Grammar as a Base for a Natural Language Processing System." in Proceedings of the First Meeting of the ACL. Europe.

Halvorsen, P. (1983) "Semantics for Lexical Functional Grammar" in Linguistic Inquiry Vol. 14, Num. 4. MIT Press.

Heim, I. (1982) The semantics of Definite and Indefinite Noun Phrases," Ph.D. dissertation, Univ. of Mass.

Johnson M. & E. Klein (1985) "A Declarative Form of DR Theory," talk presented at Summer Association of Symbolic Logic & Computational Logic Meeting, Stanford CA, 1985.

Kamp, H (1985) "Context, Thought, and Communication," in Proceedings of the Aristotelian Society 85, 239-261.

Kamp, H. (1981) "A Theory of Truth and Semantic Representation", in Groenendijk, J., Janssen, Th. & Stokhof, M., eds., *Formal Methods in the Study of Language* , Mathematisch Centrum Tracts, Amsterdam.

Reyle, U. (1985) "Grammatical Functions, Quantification and Discourse Referents." in Proceedings of the Ninth International Joint Conference on Artificial Intelligence. 829-831.

NOTES

[1]We should note that there are other ways of defining accessibility. One might be tempted to do it wholly in terms of the possibility of assigning the DRS containing the identification coherent truth conditions. Chierchia and Rooth (1984) investigate this possibility. We do not find their approach computationally useful, however, insofar as checking for truth conditional consistency is any more expensive than following the accessibility constraints on DRSs.

[2]The truth conditions for attitude reports are too complex to give here. For details see Asher (forthcoming), Kamp (1985).

[3]We have plans to expand the constructor to handle plurals as well.

[4]The other alternative of course is that the pro occur in the object position. Our algorithm handles that similarly.

[5]The question is more delicate within the context of propositional attitude verbs. There it seems not all definites function in the same way. Some permit anaphoric linkage across propositional attitude contexts and some do not.