A theory for the representation of knowledge

by Franz Guenthner
Hubert Lehmann
Wolfgang Schönfeld

How to represent knowledge is one of the key questions in the construction of expert systems. Its solution depends on a number of factors, the most important of which are how knowledge is to be acquired and how it is to be used. Since we are interested in the use of natural language for communication with computers, we require from a formalism suggested for knowledge representation that it be suitable as a target for the systematic translation from natural language expressions. We want to propose a theory, called *Discourse Representation Theory (DRT)*, which was originally developed by Kamp to analyze natural language discourse, as a means to represent knowledge in an expert system. With Discourse Representation Theory it has been possible to solve certain cases of contextual relations which have puzzled linguists and logicians for a long time. In this paper we give a precise definition of DRT and describe the rules used to translate from natural language to Discourse Representation Structures (DRSs), the central notion of the theory. We show how the notation used in DRT relates to standard predicate logic and define its deductive theory. We also outline ways of implementing DRT and the proof procedures we intend to use.

Introduction

How to represent knowledge is one of the key questions in the construction of expert systems. It is often claimed that once this question has been decided, the remaining tasks are "easy" to do. The solution of this question is dependent on two more problems, namely,

1. How is the knowledge acquired from the expert?
2. How can the knowledge in its formal representation be used to answer user questions?

Basically, there are two ways to solve the first problem:

1. The expert communicates his or her knowledge to the system through interaction with a "knowledge engineer," who encodes it directly in the representation chosen.
2. The expert communicates with the system directly, i.e. through statements in natural language, through a graphic interface, or through a formal language he or she is familiar with.

Of these possibilities, we are interested in the natural language communication of knowledge to an expert system. This choice imposes important restrictions on the kinds of formalisms which can serve to represent knowledge, as will become clear below.

Different schemes for representing knowledge have been proposed, most of which originate from the needs for natural language processing and have been used in prototype systems that process fragments of natural language. Among these schemes are *semantic nets, frames, scripts, production rules, conceptual graphs* [1], so-called knowledge representation languages, such as KL-ONE [2] or KRYPTON [3] and others (cf., e.g., the SIGART newsletter of February 1980 [4] for references).

We propose Discourse Representation Theory (DRT), a theory originally described by Kamp [5] to study the meaning of natural language *discourse*, as a knowledge representation language. Formally, DRT can be regarded as a variant of first-order logic; hence it possesses a well-defined semantics and an equally well-defined deductive theory, which distinguishes it from a number of other proposed knowledge representation languages whose semantics—inasmuch as it cannot be easily translated into that of first-order logic (cf. Hayes [6])—remain unclear. Similarly, as was shown by Brachman and Levesque [7], it is quite uncertain that alternate ways to represent knowledge allow for less complex deductive algorithms than those proposed for first-order logic.

The work reported here is being carried out as a joint project of the IBM Heidelberg Scientific Center and the University of Tübingen. The objectives of this project are to develop a prototype system which is able to analyze natural language discourse, extract the information conveyed by it, and translate it into logical form to make it available for answering questions. The texts considered may describe singular facts, events, and states of affairs, but they may also express general rules. In a first application we want to apply this system to German traffic law. In this application the system is to be used by a lawyer or judge for consultation about a particular case description, about the relevant paragraphs, interpretations of the paragraphs, and controversial issues. Thus the lawyer or judge will be able to get a better understanding of complex cases, and thus to build up his or her lines of argument more effectively. The prototype system we are working on, consequently, processes the German language. For ease of presentation, we only use English examples here, as no German-specific issues are discussed. The User Specialty Languages (USL) System which we are using as a base and which was originally developed for natural language interaction with databases [8–11] processes German, English, French, Italian, and Spanish, such that the English examples given would actually be processed as described.

In this paper we first discuss adequacy conditions for knowledge representation languages. We do this in terms of the problems that have to be dealt with, most of which have been recognized in the Artificial Intelligence literature, as well as in theoretical linguistics, logic, and philosophy of language. We emphasize, in accordance with many other researchers in the field, that knowledge representation requires the well-established methods of logic to be successful, i.e., to pass from an art to a theory. To make logic "work" on a machine, it is certainly necessary to provide deductive *strategies* which have to be formulated at a meta-level and are needed to supplement a purely descriptive formalization of a domain. It is our view that a clean separation of descriptive and strategic knowledge is crucial for a proper understanding of the issues in knowledge representation. We are not very specific about issues of strategic knowledge, although we believe that the deductive mechanism we propose furnishes a good base.

The main part of this paper is devoted to a presentation of Discourse Representation Theory. This presentation is based on several articles by Kamp [5, 12–14] and Partee [15]. We present a linear syntax for Discourse Representation Structures (DRSs) which consolidates the different versions presented in the publications cited. Our treatment of the formal semantics differs somewhat from Kamp's presentation in that partial structures are used instead of models. The semantics of DRSs is inductively defined. We then discuss how concepts expressed in natural language can be mapped to predicates. This discussion is a prerequisite for the presentation of the DRS construction algorithms which follows. It is the task of this algorithm to perform the systematic translation of natural language discourse into the form of a DRS.

Another novel feature in this paper is the presentation of a deductive theory for DRSs. It is based on the tableau calculus, which we feel is particularly well suited for computer applications such as expert systems, since in addition to finding the proof of a goal, counter-examples are found when the proof yields a contradiction. This is much more helpful to a user than the mere information that a goal could or could not be proved.

We end the paper with a discussion of extensions of Discourse Representation Theory, which we did not include before in order not to overload the first presentation with too many details and because some of these extensions need more investigation before they can be adequately formalized and integrated.

As an appendix we present a somewhat extended example from our application domain to illustrate how the theory is put to use.

Adequacy conditions for knowledge representation languages

The question of what are the *adequacy conditions* relative to which proposals for knowledge representation languages are to be evaluated and compared has rarely—if ever—been systematically raised.

On the one hand, there is a plethora of such so-called representation languages and, on the other hand, there is at least the explicit claim that these languages should be used to manipulate information. It is therefore surprising that so little attention has been paid to the *way* we can evaluate specific proposals; what are the minimal requirements that any knowledge representation language should meet, etc. On the contrary, proposals for knowledge representation schemes abound with remarks about notation and implementation, but almost never with comments about *expressive power* and *interpretability* in other languages, not to mention notions like *truth*, *theorems*, or *decidability*/

undecidability and the like. In his position statement in the 1980 SIGART Special Issue on Knowledge Representation [4] Robert Kowalski wrote the following lapidary statement:

"There is only one language suitable for representing information—whether declarative or procedural—and that is first-order predicate logic. There is only one intelligent way to process information—and that is by applying deductive inference methods."

In a way we agree completely with him; obviously some qualifications are in order, but the general tenor of this remark retains its force. There is an interesting analogy to be drawn here between the many proposals in the knowledge representation language field and the almost countless proposals for so-called "inference engines" in the expert systems field. Expert system designers like to speak of separating the "knowledge base" from the "inference engine," and they insist that the latter could be employed more or less "universally." But every expert system comes with its own "inference engine," and each one of them is typically much less powerful, much less transportable, etc., than even *ordinary Prolog!* If there is a representation language and a way of manipulating information that is something like "universal," it is without doubt logic. Instead of pointing out the many applications logical techniques have had in the past, let us simply indicate a few of the adequacy conditions any logical representation language brings with it automatically, so to speak. From logical representation languages we get the following advantages:

- Syntax: a clear notion of what constitutes a well-formed expression at all levels of syntactic complexity.
- Semantics: a clear relation to the structure of the world.
- Inference theories: a variety of equivalent alternatives.
- Notions of correctness: nothing false can be derived or proved.
- Notions of completeness: every truth is provable.
- Expressive capacity: what can and what cannot be expressed in a particular system.
- Distinction between propositional logic, first-order logic, and higher-order logic.
- Other systems such as lambda-calculus.
- Various other meta-logical notions concerning decidability complexity and implementational complexity.

No other framework for knowledge representation really has these properties, and the problems that are discussed in other frameworks (fuzziness, incomplete information, nonmonotonic reasoning, etc.) are not adequately dealt with in any of these either.

The success of logic programming languages like Prolog lies in the fact that they are an optimal compromise between representational expressiveness, deductive tractability, and efficient implementability. This constrasts with full first-order predicate logic, which was not developed for the needs of knowledge representation as it is understood here. It is too general both from a semantic point of view and a representational point of view.

We regard Discourse Representation Theory as a step which retains the advantages of predicate logic but is closer to natural language and to natural reasoning. We should point out though that semantically there are still many open problems (mainly appropriate denotation types and truth conditions for a variety of expressions; cf. McCarthy [16] for a short survey):

- From a reasoning point of view there are some open problems (beginning with the distinction between logical and "commonsense" reasoning: There is at the moment no working system for nonmonotonic reasoning and not even for reasoning in partially defined situations).
- In any event, in order to develop more complicated systems which accommodate more complex denotation types (e.g., events, causality, and the like), it is extremely important to have a clear grasp of the properties of the underlying language.

Basic notions of Discourse Representation Theory

Discourse Representation Theory (DRT) is a theory of meaning for natural languages which integrates both semantic and pragmatic aspects of language within a single framework. It is designed to deal with multisentential discourse, which distinguishes it from most previous approaches to the study of the semantics of natural languages. Aspects such as pronominal reference, tense, and propositional attitudes cannot be successfully dealt with when sentences are only looked at in isolation. The relation that a given sentence has with its preceding discourse is crucial to the understanding of these phenomena.

DRT defines as its central notion the *Discourse Representation Structure (DRS)*, which is a pair $<U, C>$ where U is a set of *reference markers* (the universe) and C a set of *conditions* which are either *atomic* (i.e., of the form $P(u_1, \cdots, u_n)$ or $u_1 = u_2$) or *complex* (i.e., expressing negation, implication, or disjunction). These conditions can be regarded as satisfaction conditions for a model. When K_1, K_2 are DRSs, then $<\neg, K_1>$, $<\rightarrow, K_1, K_2>$, and $<\vee, K_1, K_2>$ are conditions. DRSs can thus be embedded into each other.

We now define a few terms that we need to describe DRSs. The topmost DRS is called the *principal* DRS. A DRS K_1 is *subordinate* to a DRS K_2 if K_1 is embedded in K_2 or if, for some K_3 embedded in K_2, K_1 is subordinate to it. If K_1 is subordinate to K_2, then K_2 is *superordinate* to K_1. A DRS K is called *proper* when every reference marker u which occurs in an atomic condition of K or a DRS K' subordinate to K is contained in the set of reference markers of K or K' or some DRS subordinate to K and superordinate to K'.

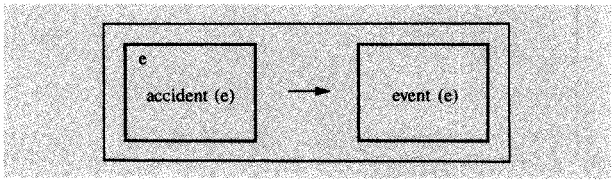

Figure 1

DRS representation of *Each accident is an event.*

The analysis of a discourse proceeds sentence by sentence. First of all the *principal* DRS is generated, and all conditions stemming from a sentence are embedded in it or in a DRS subordinate to the principal DRS. How this is done is defined by the *DRS construction algorithm*, which we describe below after having given a formal description of the syntax and semantics of DRSs.

• *Syntax of DRSs*

In his original paper [5], Kamp introduced a graphic notation for DRSs in which every DRS is represented as a rectangular box. The top part of the box contains an optional list of reference markers, its universe U. The rest of the box contains the conditions C of the DRS. As an example consider the sentence *Each accident is an event*, which is represented as in **Figure 1**.

The principal DRS contains just one condition, the implicational condition used to represent *every*. Note that the sentence *If something is an accident, then it is an event* would have led to the same representation. (A few intermediate steps would have been required, though, to get rid of the copulas and the two pronouns.) While Kamp's notation is very well suited to display the logical structure of a discourse to a human reader, it is not quite as easy to manipulate in a machine. We have therefore developed the following linear notation, which also includes additions to the notation that were introduced in later papers by Kamp and others [5, 12–15]. We use a Backus-Naur Form to characterize the syntax of DRSs:

```
<drs> ::=
        [ <reference-marker-list> : <conditions> ] |
        [ <conditions> ] |
        [ <reference-marker-list> ]
<reference-marker-list> ::=
        <reference-marker> |
        <reference-marker> , <reference-marker-list>
<reference-marker> ::=
        <object-marker> |
        <et-marker>
<object-marker> ::=
        u<number>
<et-marker> ::=
        <event-marker> |
        <time-interval-marker>
<event-marker> ::=
        e<number>
<time-interval-marker> ::=
        t<number> |
            n
<conditions> ::=
        <condition> |
        <condition>.<conditions>
<condition> ::= <atomic-condition> |
        <conditional-condition> |
        <disjunctive-condition> |
        <negative-condition> |
        <event-condition>
<atomic-condition> ::=
        <predicator> ( <argument-list> ) |
        <predicator> |
        <term> = <term> |
        <et-marker> ⊆ <et-marker> |
        <et-marker> ≺ <et-marker> |
        <et-marker> o <et-marker>
<predicator> ::=
        <identifier>
<argument-list> ::=
        <term> |
        <term> , <argument-list>
<term> ::=
        <reference-marker> |
        <number> |
        <functor> ( <argument-list> )
<functor> ::=
        <identifier>
<conditional-condition> ::=
        <drs> → <drs>
<disjunctive-condition> ::=
        <drs> ∨ <drs>
<negative-condition> ::=
        ¬ <drs>
<event-condition> ::=
        <event-marker> : <drs>
```

An identifier may be any string of characters and digits, a number any string of digits.

The example given above in graphic notation now looks like this:

$$[[e{:}accident(e)] \rightarrow [event(e)]]$$

• *Formal semantics of DRSs*

Before we formally define the semantics for DRSs, let us first discuss what is at stake here. The meaning of a DRS in the real world (whatever that means) should be derivable from the meaning of its atomic constituents. We thus have to start

by assigning meanings to atomic DRSs. These meanings should be unique. (It is one of the advantages of formal over natural languages that we *can* achieve that.) A consequence is that any atomic formula (without variables) is either true or false in a given situation (we do not adopt the standpoint of intuitionistic logic here). Moreover, the meanings of the constituents of a DRS should determine the meaning of the latter in a unique way.

We investigate the semantic properties of the language in terms of constructed artificial models, as it is in general impossible to specify the semantics in terms of properties of the real world. (Certain similarities between DRT and situation semantics [17] can easily be seen.) Clearly, these models and the definition of meaning in them should have as many properties of the real world as necessary. The usual approach taken in mathematical logic restricts models to sets of individuals, on which some relations and functions exist. All these are given extensionally; i.e., for any pair (in case of a binary relation) of individuals it is specified whether or not the relation holds. (Possibly this question is recursively undecidable.) This approach is well justified by mathematical practice; e.g., for the set of natural numbers, mathematicians try to find all true theorems. In other words, extensional semantics mirrors the attempt of mathematicians to know all properties of a certain relation, function, etc.

This is rather different from natural language discourse. To process the sentence *Pedro owns a donkey*, human beings need not know the entire extension of the relation *own*. It may well be that, during the discourse, we learn more and more of what *own* means. In order to model such processes of learning by telling, we must be able to express partial knowledge. We do that in such a way that we can still rely, if necessary, on extensional mathematical semantics.

To define the semantics of DRSs, we proceed inductively, as usual. For the sake of readability, the definition is split into several parts. Let us start with that part which corresponds to the language of classical first-order predicate logic.

A *literal* is a DRS of the form α or $\neg\alpha$ where α is an atomic condition. A *ground literal* is a literal without reference markers. A set of ground literals is *consistent* if it contains no complementary pair α, $\neg\alpha$. A *partial structure* is an ordered pair (A, L) where L is a consistent set of ground literals and A a set of terms such that, for any term a occurring in L, $a \in A$. (We assume that we have a complete denotation system for the set of individuals we have in mind.)

Note: We need not specify the nonlogical symbols of the language—they can be read off L. Moreover, we need not require that A be functionally closed.

Let $u1, \cdots, un{:}\alpha$ be a DRS, and $a_1, \cdots, a_n$ a list of terms. By $\alpha(u1/a_1 \cdots un/a_n)$ we denote the result of substituting a_i for ui in α. Clearly, if the reference marker list is empty, i.e., $n = 0$, then α remains unchanged. Note that $\alpha(u1/a_1 \cdots un/a_n)$ is of the same syntactic category as α. Furthermore, note that a term a denoting an event is substituted for all occurrences of an event marker so that they may only occur in the form $a{:}\alpha$.

Definition Let $S = (A, L)$ be a partial structure. For any syntactic construct α containing no free occurrences of reference markers,* the *truth value of* α *in* S [in symbols: $v_S(\alpha)$] is inductively defined as follows:

- For any atomic condition α,
  $v_S(\alpha) = true$ if $\alpha \in L$,
  $v_S(\alpha) = false$ if $\neg\alpha \in L$,
  else $v_S(\alpha) = undef$.
- For any DRSs α, β,
  $v_S(\alpha \vee \beta) = \text{or}(v_S(\alpha), v_S(\beta))$.
  $v_S(\neg\alpha) = \text{not}(v_S(\alpha))$.
- For any condition α and any conditions β, $v_S(\alpha.\beta) = \text{and}(v_S(\alpha), v_S(\beta))$.
- For any reference markers $u1, \cdots, un$ and any conditions α containing at most $u1, \cdots, un$ unbound,
  $v_S([u1, \cdots, un{:}\alpha]) = \max \{v_S(\alpha\ (u1/a_1 \cdots un/a_n)) \mid a_1, \cdots, a_n \in A\}$.
- For any DRS of the form $[u1, \cdots, un{:}\alpha]$ $(n \geq 0)$ and any DRS β,
  $v_S([u1, \cdots, un{:}\alpha] \rightarrow \beta) = \min \{\text{seq}(v_S(\alpha(u1/a_1 \cdots un/a_n)), v_S(\beta(u1/a_1 \cdots un/a_n)))|a_1, \cdots, a_n \in A\}$.
- The case $\alpha = empty$, i.e., $v_S([u1, \cdots, un] \rightarrow \beta)$, is defined appropriately.

Here we assume that the three truth values are ordered: $true < undef < false$. For any truth values x, y let

$\text{or}(x,y) = \max \{x,y\}$,
$\text{and}(x,y) = \min \{x,y\}$,
$\text{not}(\text{true}) = false$, $\text{not}(\text{undef}) = undef$, $\text{not}(\text{false}) = true$,
$\text{seq}(x,y) = \text{or}(\text{not}(x),y)$.

Note that we do not require that the predicators $=$, $\prec$, and $\subseteq$ have any special semantics. Instead, we assume that their properties are specified by *meaning rules* or, possibly, schemata for such rules as done in mathematical logic for the equality $=$. For the case of a conditional condition, note that the reference markers occurring in the list at the top of the premise should be seen as universally quantified and, furthermore, have their scope extended to the whole of the conditional condition.

To handle event markers, we define an operation which reduces DRSs with event markers to those without. Suppose that we know for each predicator P in a DRS α whether it is event-dependent or not. Furthermore, suppose that event markers have to stand in the first argument position (if they

* We assume that the principal DRS has this property and formulate our definition in such a way that it remains true for all induction steps.

occur at all) in any atomic condition. For any term a and any DRS α, we define the reduction $\alpha(/a)$ (we might also call it *full parameterization*) inductively.

If α is atomic with an event-independent predicator, then $\alpha(/a) \equiv \alpha$.
If, for an event-dependent predicator P, $\alpha \equiv P(b, c_1 \cdots, c_n)$, where b is an event term or $\alpha \equiv P(c_1 \cdots, c_n)$, then $\alpha(/a) \equiv P(a, c_1, \cdots, c_n)$.

The other cases are similar.

Definition (continued)

- For any event term a and any DRS α,
  $v_S(a{:}\alpha) = v_S(\alpha(/a))$.

Other kinds of modifiers, such as the time-interval-marker, are handled in the same way.

- *Representation of concepts underlying natural language expressions*

When a body of knowledge is to be formalized, it has to be determined first of all what the relevant concepts are and what kinds of terms or predicates are suitable to represent them. (This problem is hardly ever discussed in the literature on knowledge representation, one exception being Woods [2], who briefly mentions it.) When one deals with the manifestation of such concepts in natural language expressions, certain guidelines are furnished by the conditions of occurrence of such linguistic expressions in larger contexts. Thus one can achieve *modular* formalizations where only those concepts particular to a given domain are specifically defined for it, and all the more general concepts are defined with the appropriate generality. (This is probably achieved at the cost of some efficiency.)

The second step in the formalization is to establish the relations holding among the concepts relevant for the given domain. This is the most difficult part of the formalization process. The statements used to express such relations we call *meaning rules*. The argument brought up by some proponents of knowledge representation languages that meaning rules are inadequate and that they would lead to horrible performance stems from a confusion of representation and implementation. We want to clarify the logical properties of our representation before we explore the most sensible implementation.

In the following two sections, we discuss methods for choosing predicates and then methods for finding meaning rules. We argue that looking at how the world is conceptualized in natural language helps in both of these tasks.

Predicates and their arguments

Verbs, common nouns, and adjectives are often treated as predicates. When it has to be decided how their respective complements are to be represented, as argument places or as some kind of operators, it is hard to find agreement. We restrict ourselves here to a few comments on each of the categories mentioned, since it would be far beyond the scope of this paper to list all the relevant criteria.

For predicates with more than one argument place, one has to define how complements are mapped to argument places. This is conveniently done by defining *roles* as, for instance, in the USL system (cf. Zoeppritz [11]), in some versions of semantic networks, and also in database theory.

Verbs The complements of verbs have been classified in a number of different ways by different linguistic schools. Often used are the traditional categories *intransitive*, *transitive*, and *di-transitive* for English. The corresponding verbs are then represented by 1-place, 2-place, and 3-place predicates, respectively. This leaves open how verbs taking prepositional objects are to be treated (provided it has been determined which prepositional phrases play the role of objects and which the role of adverbials). *Valency theory* has brought some advance in this regard, and a few dictionaries have been compiled (in particular, for German) that specify the valencies of substantial sets of verbs. A disadvantage of valency theory is, however, that the criteria used to establish valencies are not made sufficiently explicit, with the result that conflicting classifications are proposed.

A quite different approach to classifying verb complements which builds upon Fillmore's case grammar has become very popular in the AI community and has led to the introduction of *case frames* or *templates*. When *deep cases* as suggested by Fillmore are used, two problems arise and have to be solved in a systematic way: (1) the mapping of deep to *surface* cases has to be defined for each verb, and (2) criteria have to be defined that allow the classification of verbs according to the deep cases they govern. Problem (1) implies that the same kind of analysis has to be performed which is necessary for valency theory.

Our own approach uses *surface* cases and prepositional objects and is thus close to valency theory in some respects. (A detailed description can be found in Zoeppritz [11].) We are currently extending this classification to include restrictions on adverbials which can modify a given verb. It is part of our expert systems project to classify the verbs in the most common 20 000 words found in juridical texts.

From a semantic point of view it is quite important to classify verbs as *static* verbs and *event* verbs. This is necessary to determine when a new sentence advances the point of reference in a discourse. Compare

e1: the car ran off the road.
e2: it hit a lamp post.

and

e2′: it was fast.

While e2 advances the discourse, e2′ does not. So we could stipulate conditions e1 ≺ e2 (e1 before e2) and e1 o e2′ (e1 overlaps e2′), respectively.

Common nouns The logical form of common nouns is even less agreed upon than that of verbs. Since nouns often occur by themselves (i.e., without a complement), it is tempting to represent them as unary predicates. Where genitive attributes—which are virtually always possible—are encountered, a functional representation is often suggested. This leads to multiple representations for each noun, which we believe is inadequate. In our opinion there is nothing wrong with supplying nouns with valencies similar to verbs (as has been sometimes suggested). This leads to the definition of predicates whose degree corresponds to the valency found for the noun. If certain arguments are not specified in a given sentence, we think it best to produce the desired form by λ-abstraction. (We have not yet included λ-abstraction in the formal description of DRT given above, but the extension is straightforward.)

Genitive attributes express a variety of different relations, not all of which should in fact be represented as argument places of the predicate denoted by the governing noun. Thus *owner of a car* is represented as **owner(x,y).car(y)** (one might even reduce the noun *owner* to the verb *own* if desired), but *wheel of a car* is represented as **wheel(x).car(y).part(x,y)** based on a rule schema

$$[\text{genatt}(\text{wheel}(x), \text{car}(y))] \rightarrow [\text{wheel}(x).\text{car}(x).\text{part}(x, y)]$$

where **genatt** signifies the relationship expressed by the genitive attribute. It is a matter of vocabulary definition to characterize nouns such as *owner* and *part* as *relational*, whereas for other nouns meaning rules have to be defined, such as the one given above. (More on relations expressed by genitive attributes can be found in Wirth [18].)

Adjectives Kamp briefly discusses the semantics of adjectives in [14]. Intersecting adjectives can be treated as unary predicates, as long as they do not take complements. Adjectives such as *dependent on, relative to, eager to, easy to* that do take complements have to be treated in much the same fashion as verbs and nouns. Nonintersecting adjectives can for practical purposes be represented by compound predicates, e.g., for *alleged damage* one would introduce **alleged-damage**. Such a treatment is not completely general, as can be seen from the example: Not just anything can be alleged to be damage, but probably only results of events (i.e., the *genus proximum* of *damage*), such that one could define alleged damage as *the result of an event which has been alleged to be damage.* Thus the rule schema for *alleged N* would be *the genus proximum(N) which has been alleged to be N.* If such rule schemata could be found for all nonintersecting adjectives, then the representation of adjectives as functions on nouns as suggested by Montague could be used: The functions would essentially be the rule schemata sketched here.

Meaning rules

What we discuss here as meaning rules makes up the content of semantic nets, frames, conceptual graphs, and similar schemes for representing knowledge (including thesauri as they are used in information retrieval). There are few relations among concepts that account for the bulk of the information contained in actually worked-out *knowledge bases.* These are

1. Generalization (the famous IS-A of semantic networks and the BT/NT of thesauri).
2. Type restrictions (or *selection restrictions*) on complements.
3. Part/whole.

(For a discussion of IS-A, cf. Brachman [19]; for the generality relation itself, cf. Woods [2].) These are in fact the relations that are most needed to disambiguate sentences and to resolve contextual references; hence it is justified to use specialized representations and algorithms for them (as, e.g., proposed by Schubert, Papalaskaris, and Taugher [20]). These relations, as important as they may be for the processing of language, actually represent very little of the knowledge in a given domain. To find out whether someone has violated a paragraph of the law, one needs a formal representation of that paragraph, and then one can check whether what he or she has done can be subsumed under it. Such formal representations we want to call meaning rules as well, realizing that we might end up calling every universally quantified statement a meaning rule.

The meaning rules we discussed first are very numerous, and it is an interesting question whether they can be somehow automatically or at least semi-automatically derived. Wirth [18] proposed a procedure for the semi-automatic extension of a thesaurus already containing the three relations mentioned above for part of the vocabulary. His program analyzes sentences where for some concepts at least one of these relations is known, and he makes a proposal to the user, which in most instances is sensible and which the user can accept or reject. To make this procedure fully automatic would imply that new relations are *inductively* derived, but this might be feasible, provided one could find an appropriate verification scheme for inductively derived relations.

The second, logically more complex, type of meaning rules appears in many different forms, only some of which are linguistic—most of our knowledge on space, time, events, and actions is never verbalized. When such knowledge is verbalized, however, it takes the form of definitions, theorems, statutes, etc. The problem we raised in the introduction, of the expert imparting his knowledge to a

system in the form of a natural language discourse, is the problem of generating meaning rules through systematic translation of linguistic expressions. We have coded legal knowledge in the form of Discourse Representation Structures, so we know it can be done, but we realize that there are still many problems in the systematic translation, in the proper axiomatization of general knowledge which is prerequisite to understanding domain-specific rules, and in specialized deductive strategies. It is the objective of the research in which we are presently engaged to contribute to a better understanding of these issues.

- *DRS construction algorithm*

It is the task of the DRS construction algorithm to build a DRS from a given discourse $D = <s_1 \cdots s_n>$, where the s_i are the sentences of a natural language discourse. It is not the objective of this paper to spell out all the details (different versions of the algorithm are found in Kamp [5, 12, 14]). We are relatively informal in the description, as in the present context it is mainly important to show that the relationship between natural language expressions and DRSs is indeed systematic.

The DRS construction algorithm includes

1. Syntax analysis of sentences.
2. Rules to translate expressions into reference markers and conditions.
3. Rules to establish contextual references.

To perform syntactic analysis, several approaches have been tried in connection with DRT:

- The User Specialty Languages (USL) System (Lehmann [8, 10], Zoeppritz [11]) uses a parser for general Phrase Structure Grammars. (The parsing system itself is called User Language Generator (ULG) and is described in [21].) The parse trees it generates are translated into so-called Intermediate Structures [similar to F-structures in Lexical Functional Grammar (LFG)]. A program [22] was written to generate DRSs from Intermediate Structures covering the fragment described in Kamp [5].
- Frey and Reyle [23] use Lexical Functional Grammar to analyze a fragment of French and translate the resulting F-structures into DRSs.
- Karttunen uses Unification Grammar (Kay [24]) to generate DRSs from so-called functional structures (again similar to the F-structures of LFG).

Rules for generating reference markers and conditions

Our DRS construction rules differ somewhat from the ones given by Kamp in [5] and [14] in that Kamp assumes a parallel operation of syntax analysis and application of DRS construction rules, whereas we assume that USL Intermediate Structures have been generated before DRS construction rules are applied. Our approach has the advantage that (1) the complexity of syntactic structures is reduced; e.g., passive is eliminated before DRSs are generated, (2) scope problems of negation, quantifiers, and coordination can be dealt with more easily, and (3) forward (kataphoric) references and references involving deductive processes can be resolved in a more straightforward manner. Syntax analysis and generation of Intermediate Structures has been described in Ott and Zoeppritz [9] and in Guenthner and Lehmann [22] for the fragment of Kamp [5] and are not repeated here, although the present fragment is more extended. A few words on Intermediate Structures are necessary, however, to understand how our DRS construction rules operate. An Intermediate Structure is a tree consisting of different types of nodes:

RELATION nodes (R-nodes) consisting of a predicator and a list of ARGUMENT nodes.
ARGUMENT nodes (A-nodes) consisting of a role name and a node of type NOMSTR or VERBSTR.
NOMSTR nodes, which list features of nouns (including quantification and negation) and an R-node or a constant.
VERBSTR nodes, which list features of verbs (including verb negation) and an R-node.

Intermediate Structures can be built up recursively, and they are also recursively processed to generate DRSs. The Intermediate Structure of the sentence *every accident is an event* thus looks like

```
V(
  R(is,
    (A(NOM,N(every,R(accident,nil))),
      A(NOM,N(a,R(event,nil))))))
```

where NOM is the role name indicating *nominative.* Processing always starts with the verb at the top node, but to actually write down the condition expressed by the verb, all its arguments must have been processed, so the construction algorithm goes to the next level of recursion at the next R-node, which in the example would be *accident. accident* is universally quantified, which will cause introduction of a complex condition in the principal DRS (which we assume to have been empty):

$$[[e1{:}accident(e1)] \rightarrow [\gamma]],$$

where γ has to be replaced by the conditions generated by the verb *is* and its second argument. The indefinite NP *an event* leads to the introduction of a new reference marker *e2* in the consequent DRS and the condition *event(e2)*. The processing of *is* picks up the reference markers introduced by its arguments and produces the equation $e1 = e2$, so we get

$$[[e1{:}accident(e1)] \rightarrow [e2{:}event(e2).e1 = e2]].$$

It is the matter of a further processing step which we do not regard as part of DRS construction proper to replace *e2* by *e1* and eliminate the now redundant equation $e1 = e1$.

Individuals, proper names, definite and indefinite descriptions In [5] Kamp treated proper names as individual constants. The occurrence of a proper name in a discourse would introduce a new reference marker in the principal DRS and generate an equation of the form

<represention of proper name> = <reference marker>

in the principal DRS. This approach was felt to be inadequate, as the same proper name may be used to refer to different individuals. Now proper names are represented by unary predicates ["the person named Peter" or $Peter(u_i)$], where u_i is *anchored* to some element $a \in A$ via an equation $u_i = a$. The unary predicate is always added to the principal DRS.

Common noun phrases are treated in the present fragment only if they are singular and used nongenerically. For each common noun phrase, a new reference marker u_i is inserted into the reference marker list of the *current* DRS, and a predicate P representing the governing noun in the NP is appended to the list of conditions in the current DRS. If P is unary, then u_i is inserted in the argument place of P. Otherwise u_i is inserted at the argument place representing the domain of P, and the remaining argument places have to be filled after the (then existing) noun complements have been processed. When the noun phrase is indefinite and no further complements remain to be processed, this is all that has to be done.

Definite singular NPs can be used in a number of different ways. We single out two of these uses: the *anaphoric* and the *unique reference* use. Our strategy is to always assume that the definite NP is used anaphorically and generate an equation of the form $u_i = x$, where x is the reference marker of the antecedent NP. It is the task of that part of the construction algorithm which applies rules of contextual reference to substitute for x the actual reference marker u_j of the antecedent NP. To find a proper referent is in principle a deductive process (which may be short-circuited in many instances). If an appropriate antecedent for u_i cannot be found in the discourse, the unique reference use of the definite NP is assumed, and the equation $u_i = x$ is eliminated.

Relative clauses The intermediate structure of a relative clause has the form

V(
  R(*verb*,
    A(*role, relative pronoun*),
    further arguments))

The reference marker used for the relative pronoun is identical to the one introduced for the domain of the governing noun. The verb and all its other arguments are processed as usual.

Verbs A verb v introduces in the current DRS an event condition of the form

e_i:[$v(u_1 \cdots u_n)$]

together with condition(s) representing the tense information, e.g., $e_i \prec$ n (where n is a distinguished marker indicating *now*) when the verb was in past tense.

Every A noun phrase of the form *every* N introduces a complex condition $K_i \rightarrow K_j$ with a new reference marker u_l in K_i and the condition(s) generated by N. K_j contains the remaining conditions generated by the R-node dominating N.

Conditional clauses The Intermediate Structure for a conditional clause has the general form

V(
  R(COND,
    A(IF,*antecedent*),
    A(THEN,*consequent*)))

Processing of a conditional clause introduces a complex condition of the form

[*antecedent*] $\rightarrow$ [*consequent*]

in the current DRS, where *antecedent* and *consequent* stand for clauses which are processed in the usual way.

Contextual relations

One of the main motivations for Discourse Representation Theory was the aim to provide a better treatment for contextual relations, above all pronominal reference and reference through definite noun phrases. The theory says that reference within a sentence and between different sentences should be treated in a uniform way, which distinguishes it from most previous attempts at pronominalization. The reference of pronouns is governed by five types of criteria:

1. Morphological: gender and number.
2. Syntactic: e.g., disjoint reference.
3. Configurational: accessibility of reference markers.
4. Semantic: e.g., type restrictions.
5. Pragmatic: e.g., topic, mutual knowledge.

These criteria have been discussed in some detail in Guenthner and Lehmann [25] (cf. also the references given there for further work on pronominalization). We do not repeat these criteria here in detail, but we discuss the configurational criterion described by DRT.

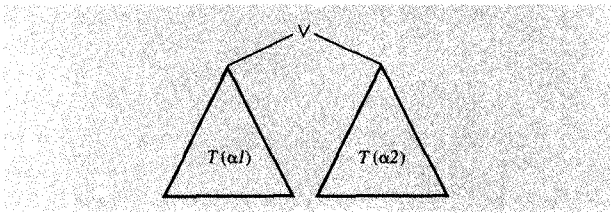

Figure 2

Tableau T(α1 ∨ α2).

Before we define the accessibility relation, let us consider a few examples:

John hit every car. *It* was green.

There is no way that *it* could refer to *car*. Note that in the corresponding DRS the reference marker for *car* is more deeply imbedded than that for *it*,

$[u1,u3,e1{:}John(u1).[u2{:}car(u2)] \rightarrow [e1{:}[hit(u1,u2)]]$
$.green(u3)]$

and this would be the only morphologically acceptable referent. Another example along these lines is

The accident was not observed by anyone. *His* car was parked.

Again, there is no way the pronoun *his* could refer to *anyone* whose reference marker is introduced in a more deeply embedded DRS than the reference marker for *his*:

$[e1, u2, u3{:}accident(e1).$
$\neg[u1{:}observe(u1, e1)].$
$genatt(car(u2),person(u3)).parked(u2)]$

When we now define that a reference marker u is *accessible* for a reference marker v iff both u and v belong to the universe of the same DRS or u belongs to the universe of a DRS which is superordinate to the DRS to whose universe v belongs, we get the desired effect for the examples given, i.e., the reference markers for *car* and for *anyone* become inaccessible.

The morphological and syntactic criteria require that some information be held available that comes from syntax analysis. This is gender, number, and the relative position of an NP in a syntax tree. The last is the most complicated of all and is needed to deal with *disjoint reference*, i.e., sentences such as

John hit him.
He hit John's car.

In both examples there is no way in which the pronoun could refer to *John*. We keep this information along with the list of reference markers.

Deductive theory

Suppose that we have a knowledge base Γ consisting of a collection of DRSs and a *goal* α and we want to know whether α logically follows from Γ. We do not present here a proof system by formalizing this notion of consequence literally. This means that we do not develop a Hilbert-style deductive theory. Instead, we take the tableau approach [26] since it fulfills the needs of computer implementation in a much better way. The central idea is that we should formalize not *consequence* but *construction of counterexamples*. The main advantage is that completeness comes in very naturally: The logical rules can be seen as a means to construct counterexamples. And this is essential for proof search procedures implemented on a machine. We pose a question to the computer since we ourselves do not know the answer. The computer can indicate the answer *no* (which is at least as probable as *yes*) only by giving some hints on counterexamples. This is the reason why we believe that deduction mechanisms based on resolution calculus [27] are not appropriate for processing knowledge.

To be more precise, we say "α follows from Γ" (in symbols: $\Gamma \models \alpha$) if for any partial structure S such that $v_S(\Gamma) = true$ (where $v_S(\Gamma) = \min \{v_S(\gamma) \mid \gamma \in \Gamma\}$) and $v_S(\alpha)$ is defined, we have $v_S(\alpha) = true$. Consequently, $\Gamma \models \alpha$ does not hold if there is a partial structure S such that $v_S(\Gamma) = true$ and $v_S(\alpha) = false$. The latter is equivalent to $v_S(\Gamma \cup \{\neg\alpha\}) = true$ or, in yet another terminology, to *S is a model of* $\Gamma \cup \{\neg\alpha\}$. The idea of *tableau calculus* is to systematically construct models for $\Gamma \cup \{\neg\alpha\}$ in such a way that, if there is none, this shows up after finitely many steps. *Correctness* means that if we conclude within the calculus that there is no model, then there is indeed none. *Completeness*, on the other hand, is just the reverse implication; i.e., if we cannot conclude that there is no model, then there is indeed one.

Completeness is proved in the following way. We define a procedure which performs in some systematic way *all* possible applications of rules for generating a model. Then we inspect its behavior. If all possibilities (cases) lead to a contradiction, then it stops and outputs *"There is no model."* The trace of all steps done so far is a formal proof for the inconsistency of $\Gamma \cup \{\neg\alpha\}$. If it does not stop with this result, then there may be two cases:

1. It stops since no rule is applicable, but some case is not contradictory.
2. It runs on infinitely.

It remains to prove that, in both cases, a model is generated. Clearly, (2) results in an infinite model whereas, in case (1), all models are finite. We would like to stress the fact that this model generation capability is that property of a deductive theory which is of highest importance in computer applications.

• *Propositional DRSs*

Let us start with the case where there are no reference markers in the DRSs. This corresponds just to propositional logic. For the sake of simplicity, we furthermore assume that all DRSs are given in *negation normal form*, i.e., are composed only by the logical connectives ., ∨, ¬ and contain ¬ only immediately before atomic DRSs. To achieve this we have to apply de Morgan's rules. In the general case we have to build just these rules into our deduction mechanism, which is rather easy.

Let α be such a DRS. As explained above, we have to specify a procedure which finds out whether there are partial structures S such that $v_S(\alpha) = true$.

The central idea is to read the definition of semantics backwards. For example, in order to make $v_S(\alpha.\beta) = true$, we have to make $v_S(\alpha) = true$ and $v_S(\beta) = true$. These two subtasks can be pursued one after another (if not in parallel). On the other hand, to satisfy $\alpha \vee \beta$ we have the choice of satisfying α or β. A tree-like organization of the resulting cases (and sub-···-sub-cases if ∨s are nested) is convenient.

Our definition below is slightly different from Smullyan's: We insert into the tableau only the final formulas, i.e., atomic or negated atomic formulas. No intermediate results are stored; they may be seen as "pushed onto a stack." Smullyan, probably not knowing the stack mechanism, keeps all subformulas arising from any rule application in the tableau. This has the advantage that the tableau contains a full trace of the search process. On the other hand, the final results, i.e., the counterexamples, cannot be read off easily.

Definition For any DRS α, the *tableau for* α [symbol $T(\alpha)$] is a tree the nodes of which are labeled by ground literals, by *contr* (for contradiction), or by the connector ∨. Branching nodes are labeled by ∨, whereas *contr* labels leaves ending inconsistent branches as soon as this inconsistency arises. [This means that a *contr*-leaf occurs exactly as (the single) son of the second item of a complementary pair of literals contained in some branch.] $T(\alpha)$ is inductively defined as follows:

- For any atomic condition or negated atomic condition α, $T(\alpha)$ is the one-element tree the node of which is labeled by α.
- For any condition $\alpha 1$ and any conditions $\alpha 2$, $T(\alpha 1 \vee \alpha 2)$ is the tree composed of $T(\alpha 1)$ and $T(\alpha 2)$, joined by a new node labeled by ∨. (See **Figure 2.**)
- For any DRSs $\alpha 1$ and $\alpha 2$, $T(\alpha 1.\alpha 2)$ is constructed in the following way: In $T(\alpha 1)$, append to each leaf not labeled by *contr* a copy of $T(\alpha 2)$, as in **Figure 3**.

$T(\alpha)$ is a *proof* for the unsatisfiability of a DRS α if all its branches are closed by *contr*.

For example, consider the DRS $\alpha = ([\neg[r].\neg[s]] \vee [s])\ .\ ([r.\neg[s]] \vee [\neg[r].s])\ .\ \neg[r]$. Its tableau is shown in **Figure 4**.

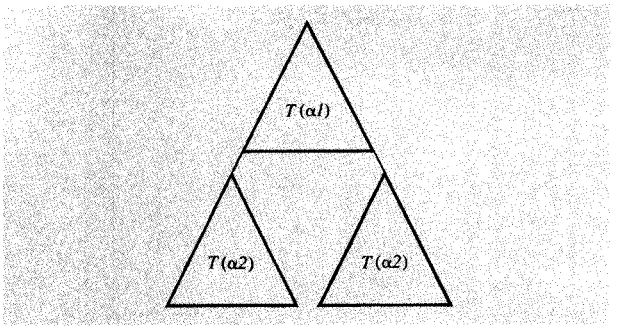

Figure 3

Tableau T (α1.α2).

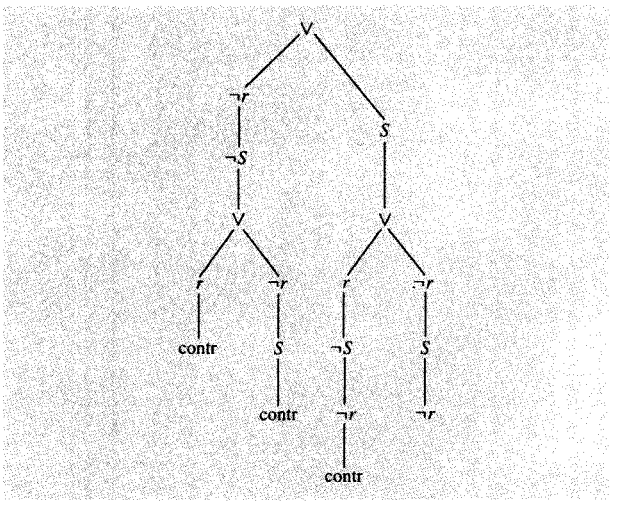

Figure 4

Tableau of DRS $\alpha = ([\neg[r].\neg[s]]) \vee [s]).\ ([r.\neg[s]] \vee [\neg[r].s]).\ \neg[r]$.

Note that it is no proof. From its rightmost branch, we may read the partial structure $S = \{s, \neg r\}$. We have $v_S(\alpha) = true$.

The process of generating a tableau can be better understood by looking at a DRS as a *logic diagram:* We connect atomic and negated atomic conditions by edges in such a way that those conditions are connected paths which must be simultaneously satisfied. Then we have to find a *thread of truth* along which we can go and make all occurring conditions true in order to satisfy the whole DRS. (Compare this with the well-known flow diagram representation of programs: Execution of a program can be visualized by pursuing a *thread of control* through the flow diagram.) The logic diagram for the DRS given above is shown in **Figure 5**. It is not difficult to make this notion

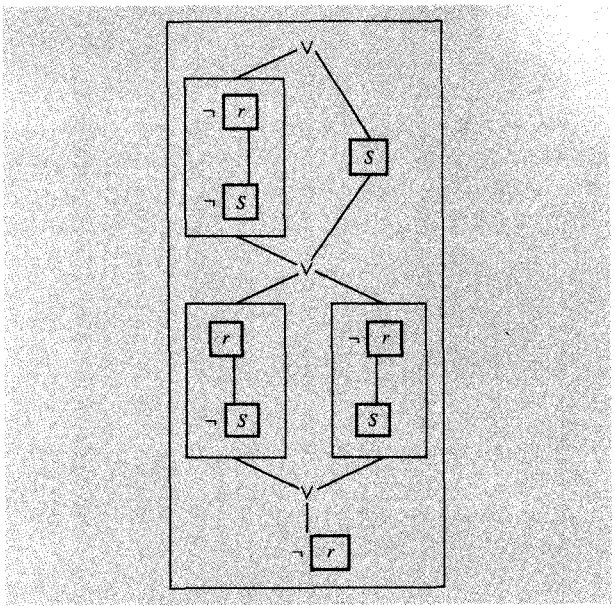

Figure 5

Logic diagram of the DRS in Fig. 4.

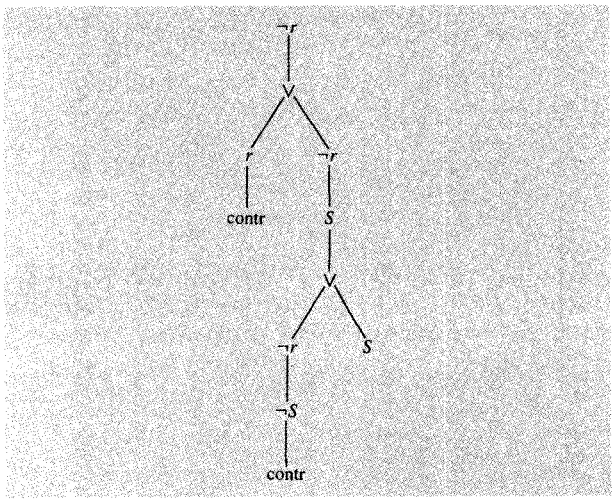

Figure 6

Preferred ordering for Γ.

precise, but we do not work it out. It seems that it is an interesting alternative to the usual and-or graphs since it makes the difference between "and" and "or" visible. Logic diagrams were introduced, in a slightly different way, in [28] but seem to have been known to logicians for a much longer time.

Let us recall the main property of logic diagrams: A (propositional) DRS is satisfiable iff its logic diagram contains at least one consistent path from top to bottom. If, when going along a path, we arrive at a branching node, then the two beginning subpaths correspond to sub-DRSs connected by a ∨. We can choose one of them to be made true. On the other hand, if a subpath is concatenated to another one, then the corresponding sub-DRSs are connected by a period. We have no choice and have to satisfy both. Now note that, for any DRS, its tableau is just the *unfolding* of its logic diagram, with inconsistent branches cut off as soon as the inconsistency arises. Hence, checking whether there is a consistent path through a DRS can be done by checking whether there is a consistent branch in its tableau. Altogether, this means that tableau calculus (for propositional DRSs) is correct and complete.

We now consider the case of a set of propositional DRSs. For example, the DRS given above can be split into the set Γ consisting of the DRSs

$[\neg[r].\neg[s]] \vee [s]$,
$[r.\neg[s]] \vee [\neg[r].s]$,
$\neg[r]$.

Clearly, the *tableau for the set* Γ may be defined as the tableau of a conjunction of all its elements (e.g., the original DRS of this example). But this is not a good choice: When humans perform deductions, they carefully choose the order in which single parts of their knowledge are applied. This reduces complexity. So, for the above example, a much better ordering of Γ exists, as shown in **Figure 6**. Note that this tableau is significantly smaller than the original one.

Let us sketch how to proceed in the general case of a set Γ of DRSs. We start by choosing a $\gamma \in \Gamma$ and generate its tableau (in case we check $\Gamma \cup \{\neg\alpha\}$, we clearly choose $\neg\alpha$). Now suppose that the tableau for Γ has been generated to a certain extent. Choose a branch B which is (so far) consistent and a $\gamma \in \Gamma$ such that no literal on B stems from γ. Generate its tableau and append it to B. Then check the resulting branches for consistency.

The conditions for choosing the next formula guarantee that each DRS is treated exactly once on each consistent branch. This has two consequences: First, no loops can arise. Second, the partial structure determined by a consistent branch satisfies all $\gamma \in \Gamma$ and, hence, is a model of Γ. This means completeness of tableau calculus for sets of DRSs. How to formulate good strategies to generate small tableaux cannot be described here. But it is one of the central problems of knowledge processing.

• *Arbitrary DRSs*

Suppose now that a set Γ of arbitrary DRSs is given. We reduce this to the propositional case. Note that there are two kinds of reference markers: those which are interpreted as existentially quantified (the normal case) and those which

are universally quantified (since they occur in the antecedent of a conditional).

Let us start with the first case, i.e., suppose that a DRS $[u1, \cdots, un{:}\alpha]$ chosen from Γ has to be analyzed. For each $i = 1, \cdots, n$ and each occurrence of ui in α, substitute a new individual constant, say a_i. Then go on as in the propositional case.

Now suppose that we have a DRS $[u1, \cdots, un{:}\alpha] \rightarrow \beta$. Recall that the ui must be interpreted universally. Hence, any term occurring in the tableau constructed so far has to be substituted. More precisely, let $a_1, \cdots, a_n$ be any sequence of terms with the following properties:

- Each a_i occurs somewhere in the tableau constructed so far.
- $a_1, \cdots, a_n$ was not yet substituted in $\alpha \rightarrow \beta$ for $u1, \cdots, un$.

Then, for $i = 1, \cdots, n$, substitute a_i for all occurrences of ui in α and β. Then go on for $\alpha \rightarrow \beta$ as in the propositional case.

In both cases, if the resulting DRS is not propositional, then the above process has to be iterated. Note that substituting all occurring terms for all occurrences (*level saturation*) is very inefficient. There are better strategies, as shown by Bowen [29] for the sequent calculus (which is quite close to tableau calculus). His central idea is to perform only those substitutions which result in at least one complementary pair of literals. Additionally, recall that, even in the pure propositional case, the order in which we choose the next DRS affects the amount of search space.

For illustration, let us give an example (taken from [30]). Let $\Gamma = \{\gamma_i \mid i = 1, 2, 3, 4\}$, goal γ_4, and $\gamma_1 = \forall xPx$, $\gamma_2 = \forall z(\neg Pz \vee (Qz \vee Rz))$, $\gamma_3 = \forall y(\neg Qy \vee Sy)$, $\gamma_4 = \neg(Ra \vee Sfa)$. See **Figure 7**.

In this tableau, we have indicated the origin of any literal. Furthermore, we have eliminated $\vee$-nodes to make the diagram more readable. Note that it is no proof; a model of Γ is given by the branch with leaf Sa.

To sum up, there are two kinds of choices which have to be performed carefully:

- Which DRS $\gamma \in \Gamma$ is to be handled next?
- If the chosen γ is universal, which substitution is to be performed next?

As we have stated previously, it is one of the central problems of knowledge processing to find good strategies to make the choices. This will influence efficiency much more than using other rule sets or even other knowledge representation languages. *The question of tractability is a matter of expressiveness, and not of the particular description language* [7].

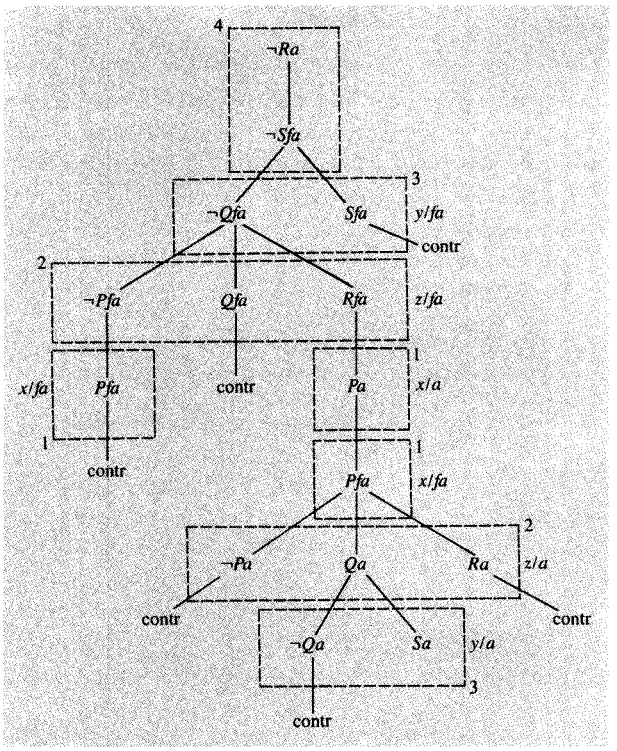

Figure 7

Tableau of $\Gamma = \{\gamma_i, i = 1, 2, 3, 4\}$.

Extensions to the basic theory

One of the novel features of DRT is the uniform representation of old and new knowledge; in fact, the entire process of changing a knowledge base, i.e., of *informing*, can be regarded as consisting in the incorporation of new conditions into an already existing DRS.

- *General treatment of information and linguistic meaning*

In Discourse Representation Theory we speak of the incorporation of the DRS of a discourse into an already established DRS. This process can also be regarded as the extension of an existing DRS. From a model-theoretic point of view, what this means is that the class of models of the latter is narrowed down to a smaller class as a result of the new information brought in by the discourse. More generally, however, we may speak of the "linguistic meaning" of a sentence or discourse as a function on the DRSs which it induces. In particular, the meaning of a DRS is a function which maps DRSs into DRSs. (Various properties of such functions are spelled out in detail in [31].) And from this function we may reconstruct in a systematic way the derivative notions of truth in a model and proposition. This is to be contrasted to the way one attempts to deal with these notions in standard model-theoretic

semantics, where one starts from extensions in models (or possible worlds) and then goes on to reconstruct intensions in terms of these. In DRT we start with linguistic meaning (a level which allows much finer distinctions between meanings than intensions) and reconstruct intensional semantic values as well as extensions from it. Thus it may very well turn out that, for instance, logically equivalent sentences (i.e., intension-equivalent sentences) do not have the same "linguistic meaning." Consider, for instance, this example (due to Barbara Partee):

One of the ten balls is not in the bag. It is under the table.
Nine of the ten balls are in the bag. It is under the table.

As this example makes clear, logically equivalent sentences may behave very differently in discourse, i.e., establish different DRSs, and should thus be considered as having different linguistic meanings.

We speak of extending a given DRS with the DRS of a given discourse. This concept gives rise to a rather novel distinction between two kinds of semantic relations (discussed below). But in addition to the notion of extending a DRS with new conditions, we should also allow a further way of operating on a DRS. Whenever a given DRS is extended on the basis of new incoming information, another process typically takes place, namely, a completion of the new DRS. This completion in general involves certain concepts associated (sometimes only in a loose way) with the properties introduced into the DRS. It is here that a number of well-known problems involving stereotypes (cf. Dahlgren [32]) and the default assumptions should be dealt with. We do not deal with these issues in detail here; it is sufficient to point out that this notion of a completion needs to be distinguished from the more "logical" process of building up a DRS from natural language input.

• *Treatment of ambiguity and presupposition*

From a linguistic point of view the analysis of how DRSs are constructed in a systematic way from natural language input has always faced the problem of how to deal with the ambiguity as well as the presuppositions of natural language utterances. In the past little clarity has been shed on these issues, and in fact these concepts have often been confused with a variety of other notions (ambiguity, for example, with the concept of vagueness, etc.).

Within DRT we are able to distinguish two kinds of semantic relations in a systematic manner: Let us consider an arbitrary DRS K. On the one hand we can characterize the relations between K and the class of its intended models, i.e., the models that represent the situations the discourse from which K results or "talks about." Let us call these relations *truth relations*. Given K we can, for example, ask whether K is true in a particular model M, or we can ask whether K has a model, i.e., whether or not K is *consistent*. Or we can ask *how* (for instance, relative to which criterion of the application of predicates) we can determine whether M is a model of K (it is here that issues of vagueness are to be resolved). Notice that such problems have nothing whatsoever to do with questions of ambiguity or presupposition.

But given a DRS K we can also consider relations of a quite different kind. The most interesting question here is the determination of the possible *extensions* of K given a discourse D. The semantic relations we want to define here concern the background DRS K and the incorporation of the DRS K' of the discourse to be processed in the context K. There are many semantic properties of natural languages which should be regarded as pertaining to relations of this type, which we call *discourse relations*. Two of the most well-known relations that belong here are *ambiguity* and *presupposition*. We say that a sentence S is *ambiguous* if there is a DRS K such that the DRS $K(S)$ associated with S can be incorporated into K in at least two different ways. For instance, let K be the empty DRS; then it is clear that a sentence like

John saw Mary with the telescope.

will give rise to two distinct extensions of K. But in ordinary discourse the background DRS into which the content of a sentence is to be incorporated is, of course, never empty. In fact, it is the presence of the background DRS which governs our choice of words so that ambiguities of the above kind never arise in a problematic way.

In order for a sentence like the above to remain systematically ambiguous, the hearer should be acquainted with but the barest information concerning the individuals involved and also concerning the communicative intention underlying the utterance of the sentence. This is indeed quite uncommon. On the contrary, whenever such utterances occur, the DRS into which their content is to be incorporated contains quite a lot of information which will force the disambiguation right away. For example, the use of the definite article requires an appropriate linking between the discourse referent associated with *the telescope* and a discourse referent in the background DRS. In most cases the latter will be related to either one of John and Mary, and this will resolve the ambiguity in the *resulting representation*. The sentence itself—given the informal definition above—remains ambiguous. How such ambiguities are systematically resolved is in general a quite complicated matter, for not all cases are structurally as simple as the case of the modification in the example above.

The case of presupposition also concerns a similar question: namely, the question into which DRSs K a given DRS K' *can* be incorporated. Informally, we say that with each DRS K we associate a set $\mathbf{K}$ of *presupposed* DRSs, those which K can in principle extend. In many cases (though certainly not always) it is possible to characterize this set via sentences in the language, and we can therefore speak of the

presupposition of a sentence as being another sentence. Among the many interesting consequences of regarding presupposition as a relation between DRSs, we mention only the fact that (i) on this account presupposition is not primarily defined in terms of *truth* (as is done in just about every other account), and (ii) many problematic cases having to do with the "projection" of presuppositions can be dealt with in a straightforward way once we look into a presupposition in the manner discussed here.

• *Contextual reference and complex anaphora*

The fact that in DRT the meaning of a sentence (or discourse) is taken to be a function which maps DRSs into DRSs (and that DRSs are systematically constructed from sentences together with the current background DRS) allows a sophisticated treatment of various phenomena of contextual reference and anaphora. In particular, personal and temporal anaphora can be dealt with a novel way. In a similar vein certain types of contextual reference can now be accommodated in a uniform way, given the fact that we have a representation of the linguistic context as well as of the more general situational context in which utterances occur. So, for instance, the treatment of anaphoric uses of the definite articles boils down to being able to link the discourse referent introduced for the definite noun phrase with another discourse referent from which it follows (in the logical sense) that it has the property expressed by the noun phrase. Since this linking takes place within the DRS and not within the world, we can have uniqueness in the former whereas, of course, no such uniqueness will in general obtain in the world. The notion of presupposition in the sense hinted at above will also play an important role here; for the use of the definite article will be characterized in terms of the extension of DRSs which satisfy the presupposition of the definite phrases.

• *Plural*

It is clear that DRSs can be many-sorted; in addition to "temporal" individuals (e.g., times, events, intervals, etc.) we can also—much as in the case of higher-order logic—introduce reference markers that are *sets*. In this way a very natural treatment of various kinds of plural expressions can be formulated both at the level of the DRS construction algorithm and at the level of the DRS semantics. For instance, simple plural phrases like *John and Mary* would introduce a set discourse reference marker X with the conditions that John and Mary are elements of X. Depending then on the verb phrase following the NP *John and Mary*, the property predicated of X will either hold for X or for the members of X. A similar account can be given for generalized quantifiers like *most girls*; an account along these lines would allow a very straightforward explanation of why most plural noun phrases allow anaphoric chains (e.g., *Most girls arrived late. They . . .*), whereas related singular noun phrases do not (e.g., *every girl arrived late. *She . . .*). Thus both syntactic and semantic properties of plural expressions can be dealt with in a fruitful manner in the DRS framework. In particular, it seems likely that similar restrictions about the nature of the set discourse referent markers as in so-called "weak second-order logic" (where quantification over only finite sets is allowed) can be made here as well. This would constitute a reasonable way of extending first-order logic to deal with set-denoting expressions.

• *Vagueness and nonmonotonic inferences*

In many ways DRT can be considered a more adequate alternative than standard formulations of first-order logic for the representation of linguistically expressed knowledge. It is closer to the form of natural language discourse and allows a more transparent representational format than standard logic. In addition, it is provided with both a precise semantics and deductive theory, which makes it a more attractive candidate than most other systems of knowledge representation. But, as in standard logic, quite a few problems remain which do not have an automatic solution in DRT either and which are of importance for any attempts to use such a system in a computational setting. Among these problems are the many forms of *vagueness* and *nonmonotonic inferences*.

As we said earlier, vagueness concerns the way a representation can be related to the world, i.e., under which conditions predicates may be said to hold for individuals. At the very simple level of adjectives and adverbs, we already run into such problems continually. It comes as no surprise, therefore, that even within model-theoretic semantics (e.g., of the relational database form) no really convincing proposals for representation and truth have been devised, let alone implemented. There are some quite sophisticated proposals in the literature, but they do not lend themselves easily to implementation. Something quite similar could be said of nonmonotonic inference. There are several ways of pursuing issues having to do with nonmonotonic reasoning. We feel that one of the most fruitful approaches is via the notion of "partial information" and various kinds of completions of partial information. DRT, in a sense, has a built-in concept of "partial information," as every DRS is in a sense a partial specification of the way the world is. There is both an underlying logic for this notion of a partial model (which is nevertheless essentially classical logic) and a number of possibilities of how to formulate types of reasonings in such a setting. We hope to concentrate on these matters in our future research.

Conclusions

In this paper we have presented a theory for representing knowledge mediated through natural language. This theory has on the one hand a firm basis in logic, and on the other

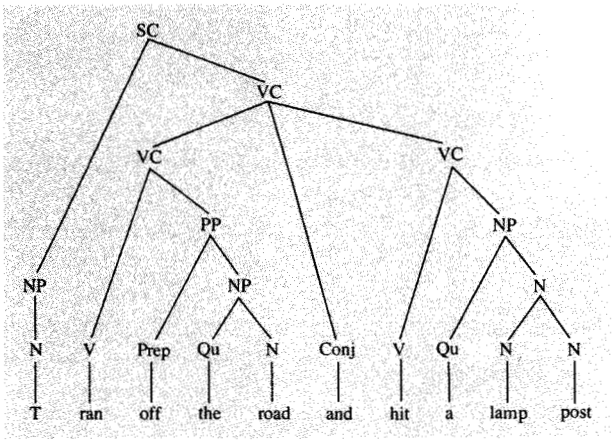

Figure 8

Parse tree of *T ran off the road and hit a lamp post.*

hand it provides for a systematic translation of natural language discourse to logical form. While the fragment of natural language which has been formally described in the theory is still rather limited, we are able to handle with it a variety of phenomena which have puzzled theoretical linguists, logicians, and philosophers of language for a long time. A number of extensions have been worked on that could not be fully discussed in the framework of this paper, but they strengthen our belief in the theoretical fruitfulness of Discourse Representation Theory. At the same time, we as well as other groups are working on computer implementations which also look quite promising. But it will have to be left to some other occasion to discuss these implementations in detail.

Quite a number of problems still wait for an adequate solution, and we mention three areas here:

1. Problems of adequate semantic representation of phenomena, such as causality, ability, etc.
2. Problems of discourse pragmatics, on which depend the proper treatment of contextual references, appropriate system reactions in a dialog with a user, and probably other things which in conventional computer applications are addressed under the label of *user friendliness.*
3. Problems of deductive strategies, which have often been addressed in the Artificial Intelligence literature, but in our view still need much further investigation.

We hope to be able to extend Discourse Representation Theory to deal with many of these problems, at least to the extent required for the expert system project we are engaged in. Discourse Representation Theory in our view provides a very good basis for attacking problems in discourse pragmatics and also for some of the issues in deductive strategies.

Appendix: An example

To show how the analysis and representation of a discourse can be done, we present here an extended example from our application domain, the German traffic law. The example is the description of an accident, where the accused, T, violated his duty to wait for a "sufficiently long" period for someone to record T's identity:

T ran off the road and hit a lamp post.
Damage of 500 DM resulted.
The accident occurred at midnight.
It was not observed by anyone.
T waited for 20 minutes.
He left the scene of accident, and he left his car behind.

For the first sentence of this text we show a parse tree in **Figure 8** as it is produced by the USL system. The category labels used are SC for clause, VC for verb complex, NP for noun phrase, PP for prepositional phrase, N for noun, V for verb, Prep for preposition, Qu for quantifier or determiner, and Conj for conjunction. We have omitted all syntactic features used to control the application of rules (cf. Zoeppritz [11] for more detail on the syntax).

Each node of the parse tree is associated with the name of an interpretation routine. After parsing, these routines are executed to produce the Intermediate Structure corresponding to the sentence:

```
V(
  R(nil,
    A(NOM,N(nil,T),
    K(and,
      V(
        R(run off the road,nil))
      V(
        R(hit,
          A(ACC,N(a,R(lamp post,nil))))))))
```

We use a dummy verb at the top of the Intermediate Structure to be able to distinguish the common subject from other arguments that individually belong to the conjoined verbs. *run off the road* is treated as a phrasal verb (this is not discovered during parsing but while constructing the Intermediate Structure).

From the Intermediate Structures the following DRS is built up (references not yet resolved):

```
[ u1, u2, u3, u4, u5, u7, u8, u9, u10, u11,
e1, e2, e3, e4, e5, e6, e7, e8, e9, t1, t2:
  T(u1).
e1: [ run off the road(u1) ].
  lamp post(u2).
e2: [ hit(u1, u2) ].
  e1 ≺ n.
  e2 ≺ n.
  e1 ≺ e2.
  damage(u3, u4).
  u4 = 500 DM
```

e3: result(u3).
accident(e4).
e3 ≺ n.
e2 ≺ e3.
e5: [occur(e4)].
midnight(t1).
e5 o t1.
e5 ≺ n
¬ [u6: e6: [observe(u6,u5)]].
T(u7).
e7: [wait(u7,t2)].
duration(t2) = 20 min.
e7 ≺ n
scene of accident(u9).
e8: [leave(u8,u9)]
genatt(car(u10),u11).
e8 ≺ n.
e7 ≺ e8.
e9: [leave behind(u9,u10)]
e9 ≺ n]

After the resolution of contextual references, which also requires application of meaning rules, and after elimination of redundant conditions, the following DRS is obtained:

The following meaning rules and rule schemata have been used:

1. [e: [verb] → [event(e)]
2. [e1, e2: e1: [occur(e2)]] → [e1 = e2]
3. [e: accident(e)] → [event(e)]
4. [u1, u2: genatt(car(u1),person(u2))] →
   [car(u1). [owner(u1,u2)] ∨ [driver(u1,u2)]]

One problem still remains: The representation of the second sentence still is not linked to the rest of the discourse. Intuitively, the linking is achieved through the elliptic occurrence of the verb *result*. To fill the gap, one would have to stipulate something like *resulted from this*, in which case the linking would have been explicit. In addition one needs a meaning rule (which has to be part of the definition of *accident* anyway) that says: If there is an accident, then there is damage caused by the accident.

References

1. J. F. Sowa, *Conceptual Structures: Information Processing in Mind and Machine*, Addison-Wesley Publishing Co., Reading, MA, 1984.
2. W. A. Woods, "What's Important About Knowledge Representation?" *IEEE Computer* **16**, 22 (1983).
3. R. J. Brachman, R. E. Fikes, and H. L. Levesque, "KRYPTON: A Functional Approach to Knowledge Representation," *IEEE Computer* **16**, 67 (1983).
4. R. J. Brachman and B. C. Smith, Eds., *SIGART Newsletter: Special Issue on Knowledge Representation*, Volume 70, February 1980.
5. H. Kamp, "A Theory of Truth and Semantic Representation," *Formal Methods in the Study of Language*, MC TRACT 135, J. A. G. Groenendijk, T. M. V. Janssen, and M. B. J. Stokhof, Eds., Amsterdam, 1981, p. 277.
6. P. Hayes, "The Logic of Frames," *Frame Conceptions and Text Understanding*, D. Metzing, Ed., de Gruyter, Berlin, 1979, pp. 46–61.
7. R. J. Brachman and H. J. Levesque, "The Tractability of Subsumption in Frame-Based Description Languages," *Proceedings of the National Conference on Artificial Intelligence*, 1984, pp. 34–37.
8. H. Lehmann, "Interpretation of Natural Language in an Information System," *IBM J. Res. Develop.* **22**, 533 (1978).
9. N. Ott and M. Zoeppritz, "USL—An Experimental Information System Based on Natural Language," *Natural Language Based Computer Systems*, L. Bolc, Ed., Hanser, Munich, 1979, p. 4.
10. H. Lehmann, "A System for Answering Questions in German," paper presented at the 6th International Symposium of the Association for Literary and Linguistic Computing, Cambridge, England, 1980.
11. M. Zoeppritz, *Syntax for German in the User Specialty Languages System*, Niemeyer, Tübingen, 1984.
12. H. Kamp, Evénements, Représentations Discursives et Référence Temporelle" (Events, Discourse Representations, and Temporal Reference), *Langages* **64**, 39–64 (1981).
13. H. Kamp and C. Rohrer, "Tense in Texts," *Meaning, Use, and Interpretation of Language*, R. Bäuerle, C. Schwarze, and A. v. Stechow, Eds., de Gruyter, Berlin, 1983, p. 250.
14. H. Kamp, "SID Without Time or Questions," manuscript, University of Texas, Austin, 1983.
15. B. H. Partee, "Nominal and Temporal Anaphora," *Linguistics and Philosophy* **7**, 243 (1984).
16. J. McCarthy, "Epistemological Problems of Artificial Intelligence" *Proceedings of the Fifth International Conference on Artificial Intelligence*, Cambridge, MA, 1977, pp. 1038–1044.
17. K. J. Barwise and J. R. Perry, *Situations and Attitudes*, Bradford Books, Cambridge, MA, 1983.
18. R. Wirth, *Halbautomatische Erweiterung eines Bestehenden Thesaurus (Semi-Automatic Extension of an Existing Thesaurus)*, diploma thesis, University of Heidelberg, 1984.
19. R. J. Brachman, "What IS-A Is and Isn't: An Analysis of Taxonomic Links in Semantic Networks," *IEEE Computer* **16**, 30 (1983).
20. L. K. Schubert, M. A. Papalaskaris, and J. Taugher, "Determining Type, Part, Color, and Time Relationships," *IEEE Computer* **16**, 53–60 (1983).
21. *User Language Generator: Program Description/Operation Manual*, Order No. SB10-7352, IBM France, Paris, 1981.
22. F. Guenthner and H. Lehmann, "Automatic Construction of Discourse Representation Structures," *Proceedings of the 10th International Conference on Computational Linguistics*, Stanford, CA, 1984, p. 398.
23. W. Frey and U. Reyle, "A Prolog Implementation of Lexical Functional Grammar as a Base for a Natural Language Processing System," *Proceedings of the First Conference and Inaugural Meeting of the European Chapter of the ACL*, Pisa, Italy, 1983, p. 52.
24. M. Kay, "Functional Unification Grammar: A Formalism for Machine Translation," *Proceedings of the 10th International Conference on Computational Linguistics*, Stanford, CA, 1984, pp. 75–78.
25. F. Guenthner and H. Lehmann, "Rules for Pronominalization," *Proceedings of the First Conference and Inaugural Meeting of the European Chapter of the ACL*, Pisa, Italy, 1983, p. 144.
26. R. M. Smullyan, *First-Order Logic*, Springer-Verlag, Berlin, 1968, p. 158.
27. J. A. Robinson, "Machine-Oriented Logic Based on the Resolution Principle," *J. ACM* **12**, No. 1, 23 (1965).
28. W. Bibel, "Matings in Matrices,"*Commun. ACM* **26**, 844 (1983).
29. K. A. Bowen, "Programming with Full First-Order Logic," *Machine Intelligence 10*, J. Hayes, D. Michie, and Y. Pao, Eds., Chichester, 1982, pp. 421–440.
30. G. Wrightson, "Semantic Tableaux, Unification, and Links," *Technical Report CSD-ANZARP-84-001*, University of Wellington, New Zealand, 1984.
31. F. Guenthner, "Zwei Typen von Semantischen Relationen"

(Two Types of Semantic Relations), *Proceedings of a Workshop on Natural Language Processing*, to appear (Bad Neuenahr, 1985).
32. K. Dahlgren, "The Cognitive Structure of Social Categories," to appear in *Cognition*.

Received March 20, 1985; revised August 8, 1985

Franz Guenthner *University of Tübingen, Tübingen, Federal Republic of Germany.* Professor Guenthner has been Professor of Theoretical and Computational Linguistics at the University of Tübingen since 1977. He was an IBM guest scientist at the IBM Germany Scientific Center in Heidelberg from 1982 to 1983. He received his Doctorate in Linguistics in 1973 from the University of Stuttgart, Federal Republic of Germany, and did his postdoctoral dissertation at the University of Stuttgart in 1977. Professor Guenthner was co-editor of the *Handbook of Philosophical Logic* with D. Gabbay and D. Reidel in 1983, of *Studies in Formal Semantics* with C. Rohrer in 1978, of *Meaning and Translation* with M. Guenthner-Reutter in 1978, and of *Natural Language Processing in Prolog* with P. Sabatier in 1985.

Hubert Lehmann *IBM Germany, Scientific Center, Heidelberg, Federal Republic of Germany.* Dr. Lehmann is a staff member at the Heidelberg Scientific Center working on a linguistics and logic-based legal expert system. He is responsible for coordination of the linguistics work and for aspects of semantics. He joined IBM Germany in 1972 in Böblingen, working on automatic language analysis from 1972 to 1973 and the User Specialty Languages system (a natural language access to a relational database) from 1974 until 1982. Dr. Lehmann studied at the University of Stuttgart from 1967 to 1972 and at the State University of New York at Buffalo during 1970/1971 on a Fulbright grant. He received his Dr. phil. in linguistics, mathematics, and philosophy from the University of Stuttgart, Federal Republic of Germany, in 1972. He is a member of the Association for Computational Linguistics (a member of the Advisory Committee of the European Chapter of the ACL from 1982 to 1985), a member of the Association for Literary and Linguistic Computing, and a member of the Gesellschaft für Linguistische Datenverarbeitung, and has been a member of the Advisory Committee of the GLDV since 1981. He is the author of *Linguistische Modellbildung und Methodologie* (1973) and co-editor with P. Hellwig of *Trends Linguistischer Datenverarbeitung* (1985). Dr. Lehmann received an IBM Outstanding Technical Achievement Award in 1984 for his research contribution to the USL System.

Wolfgang Schönfeld *IBM Germany, Scientific Center, Heidelberg, Federal Republic of Germany.* Dr. Schönfeld is a research staff member working on expert systems, especially inference engines and knowledge acquisition. He joined IBM in 1984 after holding the position of Professor of Computer Science at the Stuttgart University. He studied mathematics at the Universities of Marburg and Bonn, Federal Republic of Germany. He received his diploma in mathematics at the University of Bonn in 1970 and his Dr. rer. nat. in mathematics from the University of Stuttgart in 1974. Dr. Schönfeld did his postdoctoral dissertation in computer science at the University of Stuttgart in 1980. He is a member of the Association for Symbolic Logic and the Gesellschaft für Informatik. Dr. Schönfeld was the recipient of an award from the Friends of Stuttgart University.