A FORMALISM FOR RELATING LEXICAL AND PRAGMATIC INFORMATION: ITS RELEVANCE TO RECOGNITION AND GENERATION*

Aravind K. Joshi**
Stanley J. Rosenschein**

I. INTRODUCTION

In this paper we shall report on an initial attempt to relate the representation problem for four areas to each other through the use of a uniform formal structure. The four areas we have been concerned with are:

(1) interpretation of events
(2) initiation of actions
(3) understanding language
(4) using language

Finding such a representation would be extremely useful and very suggestive even though it would not by itself constitute a solution to the whole problem.

Clearly, (1) and (2) are 'pragmatic' in nature and are not limited to natural language processing, while (3) and (4) may be viewed as special cases of (1) and (2) respectively. One of our main goals is to show how both pragmatic and semantic issues may be approached in a formal framework. We have chosen to study the area of 'speech acts' (conversational activities like 'request,' 'command,' 'promise,' ...) as this area is especially rich in interactions among the four areas.

Our goals can be divided into two categories: operational and methodological. On the operational side, we want to implement an actual system which would 'recognize' and 'perform' speech acts and which would use and understand the verbs of 'saying'. The recognition that a particular speech act has occurred is to be on the basis of context and not solely on explicit markers like a performative verb or a question mark. We also want a symmetric system which could generate, in the context of reversed roles, anything it could understand. Initially we would be satisfied that the input and output be in an artificial language which we felt to be adequate to represent the underlying structures of English sentences (1).

On the methodological side, we have two primary desiderata: unformity of representation, and generality in the procedural component. We do not wish to write an intricate procedure for each speech act. We want to represent the speech acts in a structure with useful formal properties. (We settled on the lattice.) We want the 'state of the system' to be a mathematically tractable object as well. The heart of the procedural component is to consist of straightforward (algebraic) operations and relations (LUB, GLB, $\leq$) which could be related to certain cognitive and linguistic phenomena.

A system designed along these lines is being implemented in LISP.

II. RELATED RESEARCH (2)

This work cuts across several areas in linguistics, natural language processing, and artificial intelligence and is related to work done on 'lexical factorization' by certain generative semanticists and others. Here, as there, the attempt was to decompose the meanings of various predicates into combinations of a small group of 'core predicates' or 'primitives'. However, whereas in general the decomposition was allowed to be expressed in any suitable form (trees, dependency networks, ...) we shall decompose into a slightly extended predicate calculus in order to exploit the underlying Boolean algebra and ultimately to construct our derived lattice.

At this point, we should mention two related pieces of work. First, the notion of using lattices for 'recognizing' or 'characterizing' events is an extension of some ideas of Tidhar [T74] (see also [BT75]) who applied the principle to visual recognition of shapes. Also, Smaby's work [S74] on presupposition makes considerable use of lattice constructions, after Dana Scott, in a somewhat related spirit.

III. OVERVIEW OF THE SYSTEM

In Figure 1 we present a block diagram of the system.

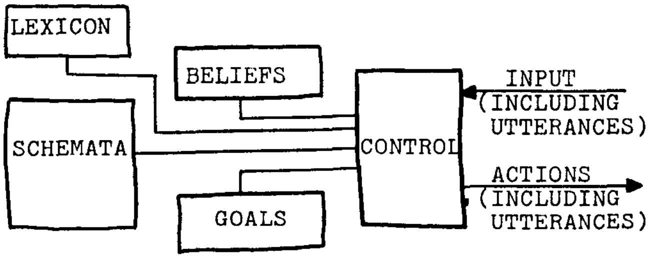


Figure 1

The block which stands for the procedural component is labeled CONTROL; all the rest are data structures. The SCHEMATA block contains the lattice whose points consist of

*This work was partially supported by NSF GRANT SOC 72-05465A01.

**Department of Computer and Information Science, The Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia, 19174

(1) Our representations are compatible with the output of the parser currently being designed and implemented by Ralph Weischedel for the computation of presuppositions and entailments.

(2) A more detailed review of related research will be included in the final version of this paper. Some examples are [B75], [F71], [JM75], [J74], [KP75], [Sch73], [Sc72], [St74], [W72].

the lexical decompositions (definitions) and certain other elements while the LEXICON contains the non-definitional information. The LEXICON and SCHEMATA remain fixed during the course of the conversation. The ´state´ or ´instantaneous description´ of the system is to be found in the BELIEFS and GOALS, which are constantly being updated as the conversation progresses.

In order to avoid confusion, we should point out that in our discussion of the system, ´beliefs´ and ´goals´ are meant as technical terms to be defined entirely by their function in the system. These terms are not to be confused with their corresponding lexical items. We shall have more to say about ´goals´ later, but for now we will concentrate on ´beliefs´.

At any given time, the system has as its ´beliefs´ a set of propositions in a predicate calculus slightly modified primarily to allow for sentence embeddings. This set has the following properties:

(1) closure -- if a proposition is in the belief set, then all its direct consequences (i.e., those following from the definitions of the lexical items) are also in the belief set.

(2) consistency -- the Boolean product of the propositions in the belief set cannot be the element ´false´.

In order to briefly illustrate these restrictions, consider the definition: bachelor (x) $=_{def}$ man (x) & ~ married (x), and the following sets:

(1) {bachelor(John), man(John)}
(2) {bachelor(John), man(John), ~married(John), ~man(John)}
(3) {bachelor(John), man(John), ~married(John)}

Set (1) is not closed; set (2) is not consistent. Set (3) is closed and consistent and is thus a valid belief set.

Note that the direct consequence relation defines a partial order over the propositions. The addition of propositions which are direct consequences of a proposition containing a defined predicate, we call EXPANSION. There is another operation which is something of an inverse of EXPANSION: Given a valid set of beliefs, this operaton augments the belief set with the least summarizing expression(s) having as consequences any two-or-more element subset of the original beliefs. This operation we call SYNTHESIS. For instance, given the set

{man(John), ~married(John)}

the performance of SYNTHESIS would yield the set

{bachelor(John), man(John), ~married(John)}.

In this example, the original set corresponded exactly to the clauses of the definition, but in general this would not be the case; other beliefs might also be entailed by the added proposition(s), and these would also have to be added. (Closure and consistency must still be preserved.)

The next section deals with how such operatons can be defined and what their implications are for a flexible understanding system.

IV. BOOLEAN ALGEBRAS AND LATTICES

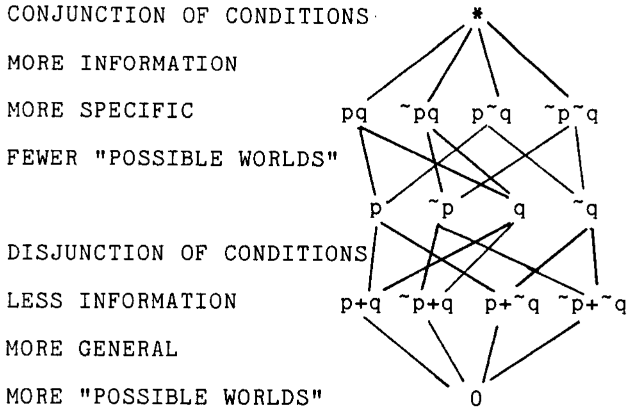
We begin by giving a brief exposition of Boolean algebras as representing information states to be followed by an explanation of how, by constructing a lattice substructure, we can formalize the notion of matching a pattern on incomplete information. The lattice will supply an internal criterion for deciding when there is enough information for a match.

Assume we are given a finite set of (primitive) predicates, each of known degree. Assume further that these predicates are to be applied to a finite set of constants. A predicate of degree n adjoined to n constants is an atomic proposition, and the negation symbol attached to an unnegated atomic proposition also yields an atomic proposition. We can think of all atomic sentences, their conjunctions and disjunctions, together with a ´greatest´ element * and a ´least´ element 0, as forming a Boolean algebra, Bool. In this algebra eery element (except * and 0) is written as a sum-of-products of atomic propositions.

We define the ´less-than-or-equal´ relation ($\leq$) as follows:

(1) $\forall x \in$ Bool, $x \leq *$
(2) $\forall x \in$ Bool, $0 \leq x$
(3) If x is a product term $x_1 x_2 \ldots x_n$ and y is a product term $y_1 y_2, \ldots, y_m$ then $x \leq y$ iff $\forall x_i \exists y_j$ such that x_i is identical to y_i (i.e., the literals of x are a subset of the literals of y).
(4) If s is a sum-of-products term $s_1 + s_2 + \ldots + s_n$ and ti is a sum-of-products term $t_1 + t_2 + \ldots + t_m$ then $s \leq t$ iff $\forall t_i \exists s_j$ such that $s_j \leq t_i$.

Following Dana Scott [Sc72], we identify the meet ($\sqcap$) with disjunction of elements and the join ($\sqcup$) with conjunction. With this convention we get the interpretation that as we go ´upward´ in the structure we get elements containing more information. The maximal element, *, is ´overdetermined´ in the sense that it contains ´too much´ information; it is self-contradictory. Conversely, the lower elements in the structure contain less information, with the minimal element, 0, containing no information at all. These notions are presented graphically in Figure 2.



[p and q are two propositions]

Figure 2

Bool was constructed over fully instantiated propositions, and as such it would not be of direct use in ´pattern matching.´ By adopting certain conventions having to do with variables and their substitutions, we can define T, the Boolean algebra of predicates (or uninstantiated logical forms), which, of course, would become *Bool* if constants were to replace the variables. It is from this structure T, that we construct the lattice of schemata.

The construction of this lattice proceeds as follows. We select from the Boolean algebra T those points which correspond to combinations of conditions which we wish to have serve as ´paradigms´ or ´schemata´. The choice of these points has to do with the empirical question of what clusters, of properties and relations are of cognitive significance, which are EXPANSION´s of lexical items, and so on.

Any arbitrary set of points drawn from T can be completed to a lattice L by adding additional points from T such that for any two points x_1 and x_2 in L, $x_1 \sqcap_T x_2$ will also be in L. While this is the general procedure, we have been working primarily with lattices that have no elements -- other than 0 -- that are strictly less than the elements corresponding to atomic predicates. We write A(x) if x is an element of T and x corresponds to an atomic predicate. We write $\hat{A}(x)$ if there exists a y such that A(y) and $y \le x$. That is, $\hat{A}(x)$ if x is ´at least atomic´.

The $\le$ relation is inherited directly from T, as is the operation $\sqcap$ [3]. However, the operation differs in that, intuitively, one may get out more than was put in. That is, in T if t_1 and t_2 are product terms and $t_1 \sqcup_T t_2 = t$, then for any element t' such that $A(t')$ if $t' \le t$, then either $t' \le t_1$ or $t' \le t_2$. However, in L this is not always the case. For example, in Figure 3, $A \sqcup B = S_1$, and clearly $C \le S_1$ and $D \le S_1$, while C and D are not comparable to either A or B. Thus, while in T we could move our information state ´forward´, in L we can move forward and reasonably extend our information beyond what was strictly given.

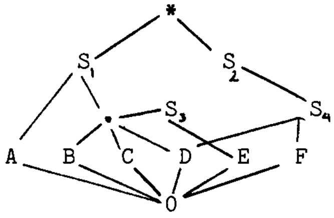


Figure 3

Intuitively speaking, we have absorbed the non-paradigmatic information states to paradigm points; $\sqcup_L$ corresponds to ´jumping to a conclusion´ -- but only to the *least* conclusion which is needed to explain the givens. *The criteria for how much to extend are in the structure itself*.

The actual computation of $x \sqcup_L y$ is not difficult, given that we have $\sqcap$ and $\le$ from T. One method follows from the observation that the least upper bound is the greatest lower bound of all upper bounds and that $x \sqcup_T y \le x \sqcup_L y$. By this method one first computes t, the least upper bound in T. (This is straightforward, as T is a Boolean algebra.) Set r to *. Then for each element x of L for which $t \le x$, set r to $r \sqcap x$. When we exhaust all such x, the value of r will be the least upper bound. Of course, other more efficient methods for computing the l.u.b. also exist.

The mechanism for event interpretation operates in the following manner. The least upper bound is taken of the points in the lattice which, under variable substitution, correspond to the propositions in the belief set and propositions in some input set. Any matched schemata (and their consequences) are added to the belief set. If the least upper bound taken in this way turns out to be *, one of two things has occured. Either the belief set contained a proposition which contradicted an input proposition, (the belief set, one should recall, could never be self-contradictory), or there is no single schema which encompasses all the propositional information. In the former case, a control decision must be made on how to integrate the new material into the belief set. In the latter case, we use the operation ´generalized LUB´, which returns a set of points, each of which is a l.u.b. for a subset of the propositions.

V. LINGUISTIC RELEVANCE

As was noted before, an attempt was made to correlate the schemata with lexical decompositions of English words, especially the verbs of ´saying´. It can be seen that definitional direct consequences (a type of entailment) corresponds precisely to the $\le$ relation. That is, the fact that a sentence

----

[3] In the case of a lattice in which for all elements x, other than 0, A(x) is true, the following modification is necessary: If $x_1 \sqcap_T x_2 = a$, and $\sim\hat{A}(a)$ then $x_1 \sqcap_L x_2 = 0$; otherwise $x_1 \sqcap_L x_2 = x_1 \sqcap_T x_2$.

using the defined predicate bachelor has man as its direct consequence implies that the point in L into which man is mapped is less-than-or-equal-to ($\leq$) the point into which bachelor is mapped. If we label points in the lattice with items from the lexicon, we get structures similar to the one shown in Figure 4. Detailed information about the arguments of each predicate has been left out for the sake of readability.

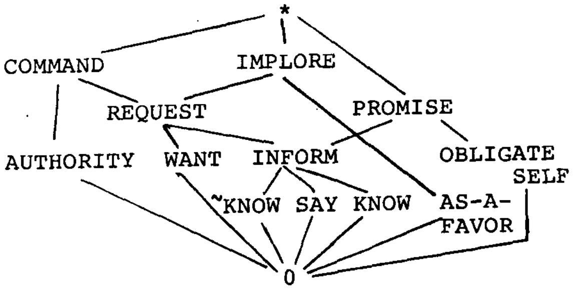


Figure 4.

The reason for embedding lexical items in the lattice is that the l.u.b. operation can be used to choose appropriate words to describe a situation (given as a ´belief set´). That is, we want the act of word selection to be identified with an operation that is naturally suggested by the formal structure. The selection of groups of words is identified with the ´generalized LUB.´

One interesting challenge emanating from this approach was to find a way in which well-known semantic properties of lexical items, such as induced presuppositions, could be integrated into the framework. For this purpose we introduced a new connective, θ, whose behavior is illustrated in Figure 5.

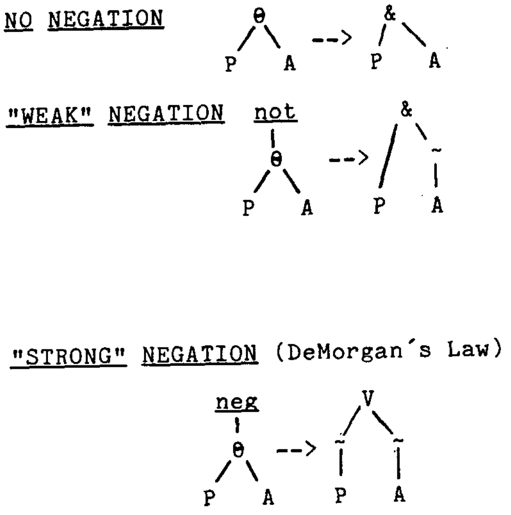


Figure 5

If P is taken to be the presupposition and A the assertion, then the two negation rewritings correspond to the usual understanding of presupposition. However both can be expressed as points in the Boolean algebra. Furthermore, if S is a sentence rewritten as $a \theta b$, then neg(S) $\leq$ not(S) (since $\sim a + \sim b \leq a \;\&\; \sim b$.) Also, if A(a) and A(b) (i.e., if a and b are atomic) then S and not(S) are higher in the lattice than the atomic sentences, but neg(S) is lower.

Recalling that moving upward in the structure is related to ´more specific information,´ some light is cast on the function of presupposition as allowing the general direction of information to be preserved even under negation of a sentence containing a complex predicate. If there were no presuppositional convention, we would move downward in information, since we know only that some component in the complex is false. With presuppositions, however, we know exactly which component is to be negated, so we keep the conjunction of clauses and hence move ´upward.´

VI. THE INITIATON OF EVENTS

Under the appropriate interpretation of the schemata we can represent how goals are set, changed, and accomplished. The essential notion is that a schema can represent a conjunction of pre-conditions, actions, and post-conditions. In this circumstance, if the ´belief set´ and the ´goal set´ satisfy enough pre- and post-conditions respectively for a particular schema to be matched by the l.u.b. operation, then the action may be taken. Of course, in the case of complete information (perfect match) the use of the schemata reduces to conditional expressions and as such is sufficient to represent any sequence of actions -- or to perform any computation. What is more interesting, however, is how the lattice provides a model of ´intelligent´ or ´appropriate´ choice of actions in the case of incomplete information. In this context, too, the ´generalized LUB´ plays a role, namely that of selecting several compatible actions to be performed.

VII. CONCLUSION

We have attempted to show how the lattice operations can be used in a variety of closely related linguistic and artificial intelligence contexts in such a way as to exploit the relationships effectively. What has not been shown here is the control structure which sequences the operations of interpretation and initiation of events (including linguistic events). A theoretically satisfying strategy has not yet been settled upon, though we have been exploring the implications of several candidate strategies. These strategies, together with the formal operations described above, are being implemented in LISP, and preliminary results suggest that such a lattice-structured system is feasible and very promising.

REFERENCES

[B75] Bruce, B. (1975) Belief Systems and Language Understanding, BBN Report No. 2973, AI. Report No. 21, Cambridge, Massachusetts.

[BT75] Bajcsy, R., and Tidhar, A. (1975) 'World Model: The Representation Problem,' unpublished.

[F71] Fillmore, C. (1971) 'Verbs of Judging: An Exercise in Semantic Description.' in Studies in Linguistic Semantics, C. Fillmore and D.T. Langendoen, eds., Holt, Rinehart and Winston, Inc., New York.

[JM75] Johnson-Laird, P. and Miller, G. (1975) 'Verbs of Saying,' unpublished, a section in a forthcoming book on language and perception.

[J74] Joshi, A.<. (1974) 'Factorization of Verbs,' in Semantics and Communication, C. Heidrich, ed., North Holland Publishing Company, Amsterdam.

[KP75] Karttunen, L. and Peters, S. (1975) 'Conventional Implicature in Montague Grammar,' presented at the First Annual Meeting of the Berkeley Linguistic Society, Feb. 15, 1975, Berkeley, California.

[S74] Smaby, R. (1974) 'Consequence, Presuppositions and Coreference,' unpublished.

[Sch73] Schank, R., Goldman, N., Rieger, C., and Riesbeck, C. (1973) 'MARGIE: Memory, Analysis, Response Generation, and Inference on English,' Proceedings of the Third International Joint Conference on Artificial Intelligence.

[Sc72] Scott, D. (1972) Data Types as Lattices, course notes of advanced course on programming languages and data structures, Mathematical Center, Amsterdam.

[St74] Stalnaker, R.C. (1974) 'Acceptance concepts,' unpublished.

[T74] Tidhar, A. (1974) 'Using a Structured World Model in Flexible Recognition of Two-Dimensional Patterns,' Moore School Technical Report No. 75-02. Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia.

[W72] Winograd, Terry (1972) Understanding Natural Language, Academic Press, New York.