READINGS IN KNOWLEDGE REPRESENTATION

QUILLIAN 1967

edited by

Ronald J. Brachman
Schlumberger Palo Alto Research
Currently at
AT&T Bell Laboratories

and

Hector J. Levesque
University of Toronto
and
The Canadian Institute for Advanced Research

MORGAN KAUFMANN PUBLISHERS, INC.
SAN MATEO, CALIFORNIA

6 / M. Ross Quillian
Word Concepts: A Theory and Simulation of Some Basic Semantic Capabilities

M. Ross Quillian's pioneering work on semantic memory models in the mid to late 1960's has greatly influenced almost all subsequent work in Knowledge Representation. Quillian is generally acknowledged to have originated the idea of a *semantic network*, in which dictionary-like definitions are encoded with nodes interconnected with associative links. This relatively difficult-to-find paper is a nice summary of Quillian's 1966 Ph.D. dissertation, in which he produced a simulation program intended to be able to "compare and contrast the meanings of arbitrary pairs of English words." Although the work described in this paper is somewhat simplistic compared to what has since appeared (*e.g.*, compare Quillian's networks to the complexity of KRL [Bobrow and Winograd, Chapter 13]), and was designed primarily as a psychological model of human language behavior, Quillian's memory model was responsible for a large number of concepts that have become fundamental to work in the field, including the type/token distinction and spreading activation search for inference. This work and its influence on representation history is treated in [Brachman, Chapter 10].

Appeared in *Behavioral Science* **12**, 1967, 410–430.

COMPUTERS IN BEHAVIORAL SCIENCE

WORD CONCEPTS: A THEORY AND SIMULATION OF SOME BASIC SEMANTIC CAPABILITIES[1]

by M. Ross Quillian

Bolt, Beranek, and Newman, Cambridge, Massachusetts

In order to discover design principles for a large memory that can enable it to serve as the base of knowledge underlying human-like language behavior, experiments with a model memory are being performed. This model is built up within a computer by "recoding" a body of information from an ordinary dictionary into a complex network of elements and associations interconnecting them. Then, the ability of a program to use the resulting model memory effectively for simulating human performance provides a test of its design. One simulation program, now running, is given the model memory and is required to compare and contrast the meanings of arbitrary pairs of English words. For each pair, the program locates any relevant semantic information within the mod l memory, draws inferences on the basis of this, and thereby discovers various relationships between the meanings of the two words. Finally, it creates English text to express its conclusions. The design principles embodied in the memory model, together with some of the methods used by the program, constitute a theory of how human memory for semantic and other conceptual material may be formatted, organized, and used.

A Memory Model

THE purpose of the research reported here is both to develop a theory of the structure of human long-term memory, and to embody this theory in a computer model such that the machine can utilize it to perform complex, memory-dependent tasks. The concepts of primary concern for storage in the model memory are those generally called the "meaning" of commonplace words, such as "machine," "family," "chair," and so on, it being assumed that such word meaning concepts are held in memory in a manner not fundamentally different from long-term concepts in general. The model employs much of the machinery traditionally used in psychology for representing concepts—conjunctive, disjunctive, and relational sets of attributes, criteriality, and so on, (see for example, Bruner, Goodnow, and Austin, 1956). It also utilizes other representational devices, some of which bear a close resemblance to those used by "transformational" linguistics for representing the "deep structure" of sentences (see Chomsky, 1965). However, the model's general organization is quite distinct from either of these representational systems.

Although the model is proposed as part of a theory of human memory organization, it is not at present intended to handle all the kinds of information that people presumably store in their heads. It is designed to hold only denotative, factual information, and not a person's plans for doing things (see the "schemata" of Piaget, 1950), a person's feelings about words (Osgood, Suci, and

[1] Research contributing to this report was done partly at Carnegie Institute of Technology, partly at the System Development Corporation, and was supported in part by NIH Grant, MH 07722.

A much more extended treatment of the same and related research is given in Quillian (1966). The work is being continued at Bolt, Beranek, and Newman Inc., Cambridge, Mass. This manuscript, at one stage or another, has benefited from critical readings by George W. Baylor, Daryl J. Bem, Walter R. Reitman, and Robert F. Simmons. To all of these the author expresses grateful thanks, as well as to Mrs. Jean Long and Miss Barbara Zimmerman, who have encoded much of the data. Along with almost anyone who attempts to test a psychological model by computer simulation, the author is also deeply indebted to Herbert A. Simon and Allen Newell.

Tannenbaum, 1957) nor his knowledge of the conditional probabilities of word sequences. The theory also deals only with the structure and use of well-developed memory, having little to say as yet about the *acquisition* of stored information. At present the problem of how humans acquire long-term concepts is simply finessed by taking the information to be encoded into the memory model from two already developed sources. These are, first, an ordinary English dictionary and, second, the fund of common knowledge that anyone who encodes this dictionary information into the model must have, that is, his own semantic memory.

The model memory consists, basically, of a mass of nodes, interconnected by different kinds of associative links. Each node may for the moment be thought of as named by an English word, but by far the most important feature of the model is that a node may be related to the meaning of its name word in one of two different ways. The first is directly; that is, its associative links may lead directly into a configuration of other nodes that represents the meaning of its name word. A node that does this is called a type node. In contrast, the second kind of node in the memory refers indirectly to a word concept, by having one special kind of associative link that points to that concept's type node. Such a node is referred to as a token node, or simply token, although this usage implies more than is generally meant by a "token," since, within the model memory, a token is a permanent node. For any one word meaning there can be exactly one and only one type node in the memory, but there will in general be many token nodes scattered throughout the memory, each with a pointer to the single unique type node for the concept. To see the reason for postulating both type and token nodes within the memory, it will be useful to reflect briefly on the way words are defined in an ordinary dictionary.

For defining one word, the dictionary builder always utilizes tokens of other words. However, it is not sufficient for the reader of such a dictionary to consider the meaning of the defined word to be simply an unordered aggregation of the other word concepts used in its definition. The particular configuration of these word concepts is crucial; it both modifies the meanings of the individual word concepts that make up its parts, and, with them, creates a new gestalt which represents the meaning of the word being defined. In our memory model, the configurational meaning of a concept is captured by building up, for each definition, what is best thought of as one plane of token nodes. Each and every token node lies in such a plane, and has both its special associative link pointing "out of the plane" to its type node and other associative links pointing on within the plane to other token nodes comprising the configuration. In short, token nodes make it possible for a word's meaning both to be built up from other word meanings as ingredients, and at the same time to modify and recombine these ingredients into a new configuration. Although the detailed structure of a plane will not be described until later in this paper it will be useful for understanding the model's overall organization to look at Figure 1 at this point.

Figure 1a illustrates the planes of three word concepts, corresponding to three meanings of "plant." The three circled words "plant," "plant 2," and "plant 3," placed at the heads (upper left-hand corners) of the three planes, represent type nodes; every other word shown in the figure's planes represents a token node. The nonterminated arrows from tokens indicate that each has its special pointer leading out of its plane to its type definition, that is, to a type node standing at the head of its own plane somewhere else in the memory. Each of these planes, in turn, is itself entirely made up of tokens, except for the type word which heads it. Figure 1b illustrates one of these planes. Therefore, the overall structure of the complete memory forms an enormous aggregation of planes, each consisting entirely of token nodes except for its "head" node which is always a type node.

Now, what is the full content of a word concept in such a memory? Let us define a full word concept, as distinguished from its plane or "immediate definition," so as to include all the type and token nodes one can get to by starting at the initial type node, or "patriarch," and moving first within its immediate definition plane to all the token

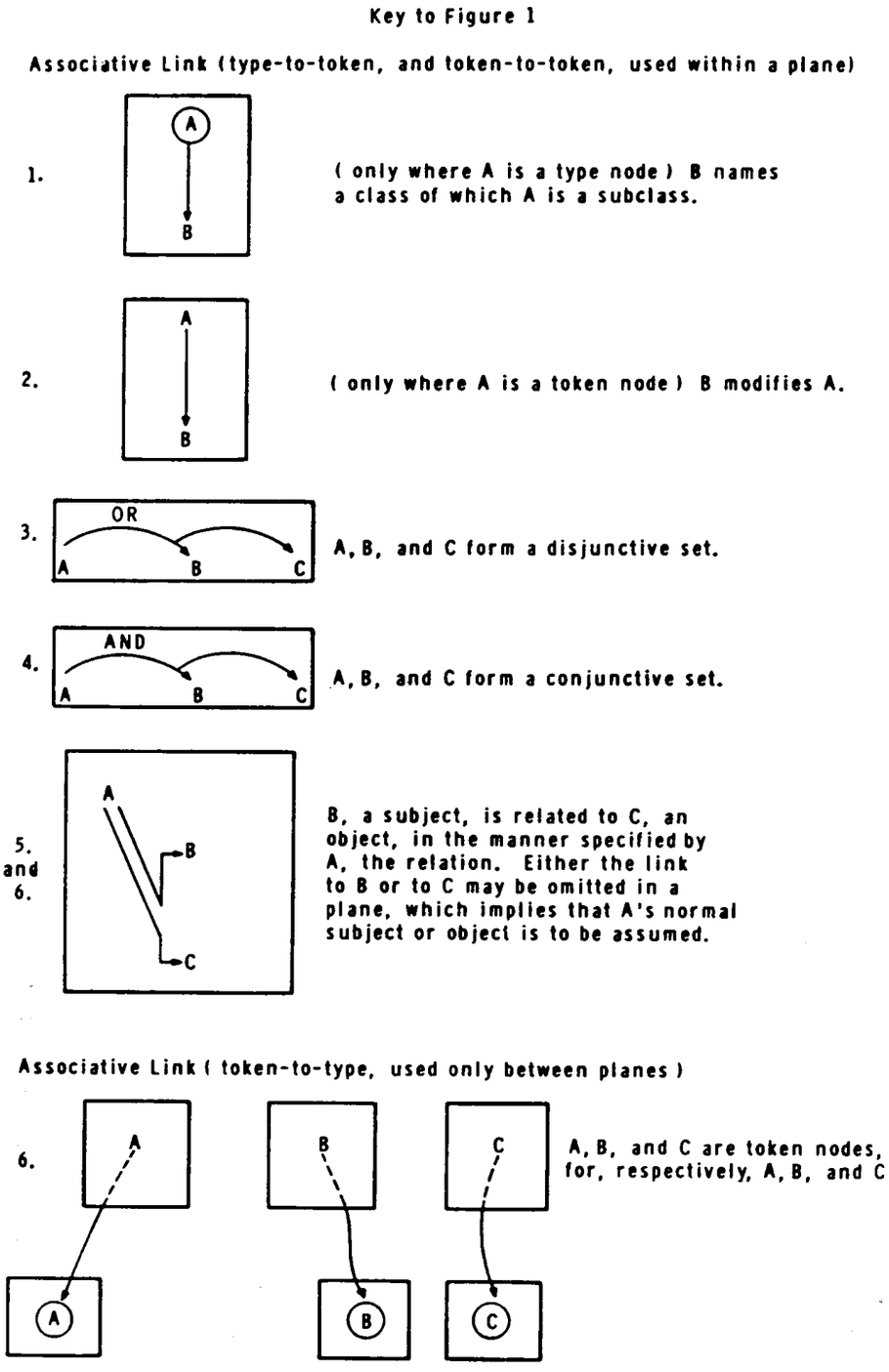

FIG. 1. Sample Planes from the Memory.

nodes found there, then on "through" to the type nodes named by each of these nodes, then on to all the token nodes in each of their immediate definition planes, and so on, until every token and type node that can be reached by this process has been traced through at least once.

Thus one may think of a full concept

analogically as consisting of all the information one would have if he looked up what will be called the "patriarch" word in a dictionary, then looked up every word in each of its definitions, then looked up every word found in each of these, and so on, continually branching outward until every word he could reach by this process had been looked up once. However, since a word meaning includes structure as well as ingredients, one must think of the person doing the looking up as also keeping account of all the relationships in which each word he encountered had been placed by all earlier definitions.

To summarize, *a word's full concept is defined in the model memory to be all the nodes that can be reached by an exhaustive tracing process, originating at its initial, partriarchical type node, together with the total sum of relationships among these nodes specified by within-plane, token-to-token links.*

We now assert that such a memory organization is useful in performing semantic tasks, and constitutes a reasonable description of the general organization of human memory for such material.

To take the latter point immediately, suppose, for example, that a subject were asked to state everything he knows about the concept "machine." Each statement he makes in answer is recorded, and when he decides he is finished, he is asked to elaborate further on each thing he has said. As he does so, these statements in turn are recorded, and upon his "completion" he is asked if he cannot elaborate further on each of these. In this way the subject can clearly be kept talking for several days, if not months, producing a voluminous body of information. This information will start off with the more "compelling" facts about machines, such as that they are usually man-made, involve moving parts, and so on, and will proceed "down" to less and less inclusive facts, such as that typewriters are machines, and then eventually will get to much more remote information about machines, such as the fact that a typewriter has a stop which prevents its carriage from flying off each time it is returned. We are suggesting that this information can all be usefully viewed as part of the subject's concept of "machine." The order in which such a concept tends to be brought forth, from general, inclusive facts to obscure or less and less closely related ones, suggests that the information comprising a word concept in the subject's head is differentially accessible, forming something that may be viewed as a hierarchy beneath the patriarch word. Our memory model's general organization is designed to make a full concept exactly this sort of hierarchically ordered, extensive body of information.

Clearly, a subject could produce hierarchical outputs similar to his output for "machine" for any one of innumerable other word concepts: "war," "family," "government," and so on, so that the overall amount of information he could pull out of his memory in this way seems almost unlimited. The sheer amount of information involved in such concepts argues strongly that both the human subject's memory, and our model of it, contain as little redundancy as possible. In this regard we note that the information a subject can generate as the meaning of "machine" will include all the information he can generate for "typewriter," among other things, and there is no need to restate the information constituting his concept of "typewriter" each time it occurs as part of the concept named by some other word such as "machine, " "office," and so on. In short, a word concept like "machine" seems to be made up, in large part, of a particular ordered arrangement of other word concepts such as "typewriter," "drill press," and so forth.

Again, a large memory structured as we have outlined above capitalizes on this redundancy, by running the pointer from every token node for a word meaning to the same type node. Note that in such a memory any given type node will have many token nodes, located in various other planes, all pointing to it, and its full concept may well contain token nodes pointing back to the type node that heads one of these planes. In other words, there is no restriction to prevent reentries or loops within a full concept, so that all routines that search through or process concepts in the memory must take account of these possibilities. Viewed most abstractly, the model memory forms

PLANT. 1. Living structure which is not an animal, frequently with leaves, getting its food from air, water, earth.
2. Apparatus used for any process in industry.
3. Put (seed, plant, etc.) in earth for growth.

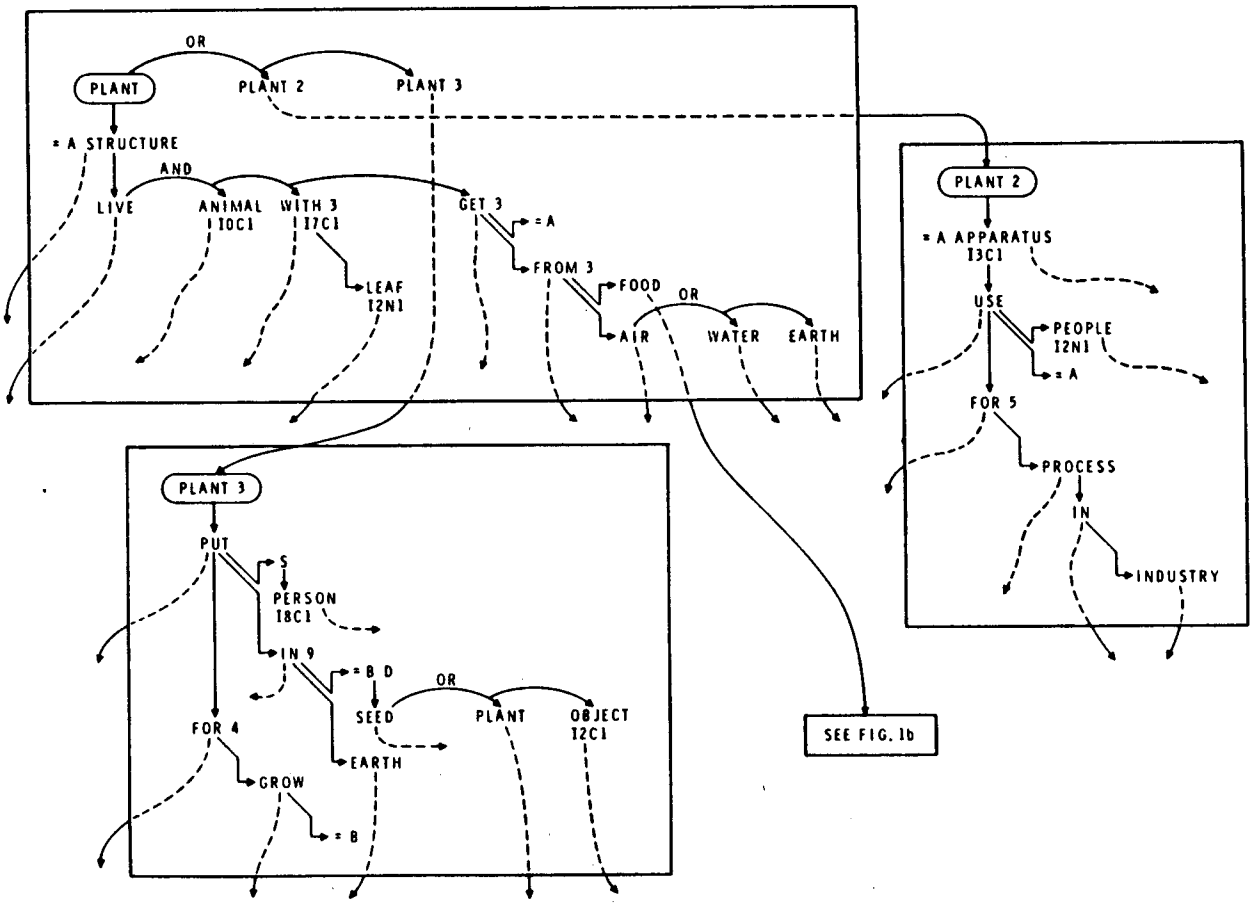

FIG. 1a. Three Planes Representing Three Meanings of "Plant."

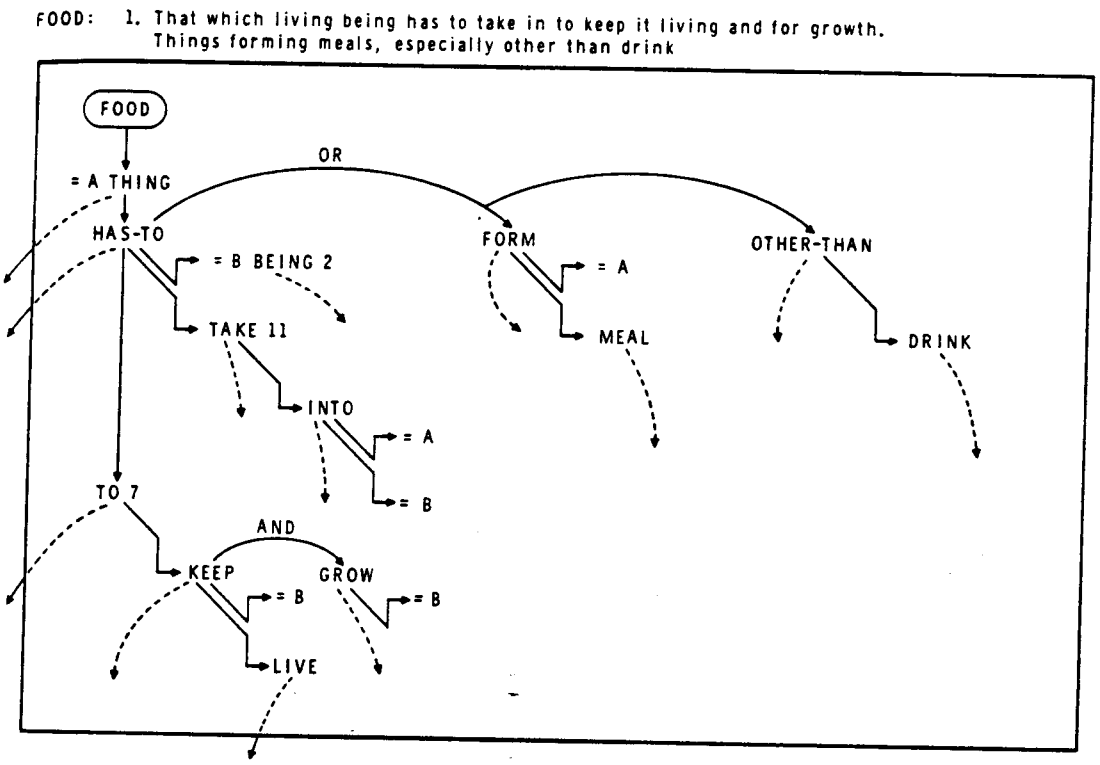

FIG. 1b. The Plane Representing "Food."

simply a large, very complex network of nodes and one-way associations between them. Most importantly, in such a model of semantic memory there is no predetermined hierarchy of superclasses and subclasses; every word is the patriarch of its own separate hierarchy when some search process starts with it. Similarly, every word lies at various places down, within the hierarchies of a great many other word concepts. Moreover, there are no word concepts as such that are "primitive." Everything is simply defined in terms of some ordered configuration of other things in the memory.

Two Constraints on the Memory

Having established the general structure of the model memory as consisting of "planes" each made up of one type and a number of token nodes, it is further necessary to determine the format of the nodes themselves, and the specific varieties of associative links between nodes to be used within a plane.

As to the nature of the nodes themselves, we assume that the relevant units of human conceptual stores are not in fact words, nor are they sentences, nor are they visual pictures; instead they are closer to what we ordinarily call "properties." This assumption is now common in work on concepts (see, for example, Hunt, 1962, or Kelly, 1955), since properties provide a more elemental and, hence, more flexible medium than visual pictures or words, and since either a mental picture or a language concept may be thought of as some bundle of properties (attribute values) and labeled associations among them.

Thus, the nodes of the memory model actually correspond more to properties than to words, although, as will be shown below, this actually need be little different from considering them to be words.

A much more important constraint arises from our assumption that, in order to continue to parallel the properties of human semantic memory, the model must be able

to link nodes together into configurations that are at least as varied and rich as the ideas expressed in natural language. Hence, simply attempting to represent natural-language definitions accurately in the model becomes a very powerful constraint dictating the model's structural properties. Over a considerable period of attempting to encode English text into such network representations, it has always been found necessary to have available several different *kinds* of associative links, rather than the simple undifferentiated associations assumed in most classical psychological studies of word association. At the same time, the model must represent all information in a form that is sufficiently standardized to allow processing by rules that can be specified explicitly, or it will be no more manageable as a theory of memory than is English itself. (See Simmons, 1963, for the most thorough attempt to use English itself as a computer's store of information on which to base the performance of complex tasks.) The representation now used in the memory model therefore lies at a level somewhere between the freedom of English itself and the standardization of, say, symbolic logic. In the memory model, complex configurations of differentiated associations must be built up to adequately represent the meaning inherent in dictionary definitions. These are the structures we have called planes. (It will be seen below that the kinds of links utilized in the model to represent configurational meaning correspond roughly to the "syntactic" interrelations between words of text: verb-to-subject, pronoun-to-referent, word-to-modifiers, phrase-to-other-phrase via some conjunction, and so on.)

While the attempt to get the meaning of English definitions accurately represented as planes of nodes within the memory model constitutes one major constraint on its structure, a second is provided by the attempt to write programs that can do something interesting by using this memory. To some degree these two constraints on the model balance one another: the first urges elaboration and complexity to represent the meaning of definitions accurately, while the second urges that the model be as simple and standardized as possible to make processing feasible.

In selecting a task to perform with a memory model, one thinks first of the ability to understand unfamiliar sentences. It seems reasonable to suppose that people must necessarily understand new sentences by retrieving stored information about the meaning of isolated words and phrases, and then combining and perhaps altering these retrieved word meanings to build up the meanings of sentences. Accordingly, one should be able to take a model of stored semantic knowledge and formulate rules of combination (such as the "projection rules" of Katz and Postal, 1964) that would describe how sentence meanings get built up from stored word meanings. A good many reasonable speculations about such combination rules can in fact be made. (For example, see Cliff, 1959. For a good over-view of these and the empirical work they have produced, see Osgood, 1963.) The bulk of an earlier paper (Quillian, 1963) and of Katz and Postal's recent work consists of such speculations.

It further seems likely that if one could manage to get a small set of basic word meanings adequately encoded and stored in a computer memory, and a workable set of combination rules formalized as a computer program, he could then bootstrap his store of encoded word meanings by having the computer itself "understand" sentences that he had written to constitute the definitions of other single words (Quillian, 1962b). That is, whenever a new, as yet uncoded, word could be defined by a sentence using only words whose meanings had already been encoded, then the representation of this sentence's meaning—which the machine could build by using its previous knowledge together with its combination rules—would be the appropriate representation to add to its memory as the meaning of the new word. Unfortunately, two years of work on this problem led to the conclusion that the task is much too difficult to execute at our present stage of knowledge. The processing that goes on in a person's head when he "understands" a sentence is very large indeed, practically

all of it being done without his conscious knowledge.

As one example, consider the sentence, "After the strike, the president sent him away." One understands this sentence easily, probably without realizing that he has had to look into his stored knowledge of "president" to resolve a multiple meaning of the word "strike." (Consider, for example, the same sentence with the word "umpire" substituted for "president.") Just what subconscious processing is involved in unearthing and using the fact that presidents more typically have something to do with labor strikes than with strikes of the baseball variety is by no means obvious, and a good part of this paper is devoted to stating one way that this can be accomplished, given that it has been decided that "president" is the correct word to attend to. Since sentence understanding involves a great number of such, at present, poorly understood processes, the two language functions that the present program performs are considerably humbler than sentence understanding.

The first of these functions is to compare and contrast two word concepts: given any two words whose meanings are encoded in the model memory, the program must find the more compelling conceptual similarities and contrasts between their meanings. Since, in the usual case, each of the two words to be compared will have several possible meanings, the program is also to specify, for each semantic similarity or contrast it finds, just which meaning of each word is involved. This is one step toward the resolution of semantic ambiguity in text. The second major task of the program is to express all the similarities and contrasts found between the two compared words in terms of understandable, though not necessarily grammatically perfect, sentences.

Although the above tasks are only a part of what apparently is involved in sentence understanding, their performance in a fashion comparable to human performance still calls for a basic degree of semantic horse-sense in which, heretofore, computers have been conspicuously lacking and which, apparently, must be based on an extensive and expressively rich store of conceptual knowledge. Thus, being able to get a computer to perform these tasks indicates to some degree the plausibility of the semantic memory model used.

In briefest form, the program that is presently running is used as follows:[2]

1. The experimenter selects a group of words whose definitions are to provide the total store of information in the memory model during a given series of tests.
2. He looks up each of these words in some ordinary dictionary.
3. He encodes each of the definitions given for each word into a specified "semantic" format, and loads them into the machine with a program that combines them into a single network of token and type nodes and associative links, the machine's model of a human memory.
4. The experimenter is then free to select arbitrarily any pair of words in the store and to ask the program to compare and contrast the meanings of those two words (requiring that its answers be expressed in sentences).
5. He may give some fluent speaker the same pair of words, asking him also to compare and contrast them.
6. He can compare the sentences the program generates to those the human has produced or, more simply, he can just judge for himself whether or not the machine's output is one that might reasonably have been produced by a subject.

If the above procedure reveals any changes the experimenter would like to see in the program's performance, he must then revise either some part of the program,

[2] The major effort at representation of a human-like memory structure in a computer so far has probably been the construction of list processors (see, for example, Newell, 1963). These provide counterparts to "associative links" and to "labeled associative links," plus processes for manipulating data stored in such form. By writing the present model and program in one of these, namely, IPL, it has been possible to begin upon the substantial foundation of design and development existing in that language, and to use "associations" and "labeled associations" freely as building blocks for the model memory.

or some part of the memory structure or content, or all of these, and further test new examples to see if the program now operates in a manner closer to what he desires. Repetitions of this kind of test-correct-retest cycle constitute the essence of the research method; however, it is important to realize that for the purposes of developing a theory of memory, the result of this development process should *not* be thought of as the computer outputs the program will now produce, but rather as what now may or may not have become clear about the characteristics of workable concept-like memories. The more general of these characteristics have been outlined above; unfortunately, the rest amount to more or less fine details of the theoretical memory and of how its provisions should be used to build up configurations corresponding to particular meanings of English text. Therefore, the next section explains these technicalities in some detail; after that, we return to a more general level to describe the actual program's performance.

Details of the Model Memory

As stated above, the relational complexity built up in an English definition is always represented in the memory by a configuration of token nodes linked together to form one "plane." Each token in a plane is linked to its type node (which lies out of the plane) by a kind of association that we show in Figure 1 as a dotted line, while it is related to other token nodes (in the plane) by one or more of the six distinct kinds of associative link listed in the key to Figure 1. In encoding dictionary definitions, these intraplane links are used, respectively, as follows:[3]

1. Dictionary definitions require the use of the subclass-to-superclass pointer whenever they define a word by stating the name of some larger class of which it is a subclass. For example, in the dictionary definition of "plant" shown in Figure 1a, the word's third meaning is said to be a subclass of the class of "putting."

2. Any word or phrase used adjectively or adverbially dictates use of the modification pointer.

3. The multiple meanings of a word, and any phrase such as "air, earth, or water," require the formation of a disjunctive set.

4. Any phrase like "old, red house" or "old house with a red porch" requires that the modifiers of "house" be formed into a conjunctive set.

5-6. Together these two links form the open-ended category, by means of which all the remaining kinds of relationships are encoded. This is necessary because in natural language text almost anything can be considered as a relation, so that there is no way to specify in advance what relationships are to be needed (see Raphael, 1964). This means that a memory model must provide a way to take any two tokens and relate them by any third token, which by virtue of this use becomes a relationship.

It will be recalled that we feel that the nodes of conceptual memory are most usefully considered to represent properties rather than words as such. Representing a

[3] Stated this way, it appears that the semantic model amounts in structure to a kind of parsing system, and that encoding dictionary definitions into it is in part, at least, similar to parsing these definitions.

This is true, and in fact what appears on one plane of the memory model has many points of correspondence with what Chomsky (1965) calls a "deep structure." In particular, the ternary relationships formed by our subject-object links are a lot like what used to be called the structure of "kernel" sentences. However, our use of terms such as "subject," "object" and "modifier" does not always correspond to that of linguistics; and, also, a plane encodes the meaning of a number of sentences, whereas a deep structure is explicitly limited to the representation of what can be represented in a single sentence (Ibid., p. 138ff). Also notice that the correspondence, in as far as it exists, is between one of our planes and one of Chomsky's deep structures, not between a plane and a generative grammar. A generative grammar is an attempt to state explicitly when and how structural information can be related to sentence, whereas the job of a person encoding dictionary definitions into our memory model is simply to get a representation of their structures, that is, to go ahead and use his language-processing abilities, rather than to describe these. Hence our coder does transformations, rather than describing them.

property requires the name of something that is variable, an attribute, plus some value or range of values of that attribute. This feature is achieved in the memory model by the fact that every token is considered to have appended to it a specification of its appropriate amount or intensity in the particular concept being defined. Omitting this specification from a token, which is generally what is done, means that no restriction is placed on the total range of variation in amount of intensity open to the attribute. On the other hand, whenever such specification does appear overtly with a token node, it consists principally of numerical values, stating how the node's total possible range of amount of intensity is restricted. These values allow encoding restrictions to a fineness of nine gradations, that is, they permit nine degrees of "absolute discrimination" to be represented (see Miller, 1956). The exact use and rationale for this kind of specification "tag" has been described elsewhere (Quillian, 1962b, 1963), along with that of the two other tags (representing the "number" and the "criteriality" of a token; see Bruner et al., 1956) that are available in the model. Here it will only be noted that in encoding dictionary definitions all grammatical inflections, along with all words such as "a," "six," "much," "very," "probably," "not," "perhaps," and others of similar meaning vanish, that is do not become nodes themselves but instead dictate that various range-restricting tags be appended to the token nodes of certain other words. Removing all inflections during encoding permits all nodes in the memory model to represent canonical forms of words, which is of importance both in reducing the model's overall size, and in locating conceptual similarities within it (see the following section).

Certain other words besides those mentioned above are also dropped during the encoding process, such as "and," "or," "is," "which," "there," and "that," these being interpreted either directly as relationships that are basic structural aspects of the model or else as directions to the coder about how he is to form the plane structure —as specifications for how the configurations of tokens on a plane are to be structured. Punctuation similarly shows up only in the associative structure of the model.

All pronouns, as well as all words used to refer again to something mentioned previously in the definition, are replaced in the model by explicit references to the earlier nodes. (In Figure 1 such referencing is being done by =A and =B, where some higher token node in the plane has been designated temporarily to be A or B by giving it a prefix of =A or =B. A more recent version of the loading program also allows referring to any token node in any plane, by a sort of "indirect addressing" feature.) This ability to, in essence, reuse tokens repeatedly in a plane, perhaps modifying them slightly each time, is extremely important in making the model correspond to human-like memory. In the course of coding a great many words into the current and earlier network representations, I have come to believe that the greatest difference between dictionary entries and the corresponding semantic concepts that people have in their heads is that, while dictionary makers try hard to specify all the distinctions between separate meanings of a word, they make only a very haphazard effort to indicate what these various meanings have in common conceptually. Although they may not be aware of it, there is a very good reason for this seeming oversight: the best the dictionary maker has available for showing common elements of meaning is an outline-like format, in which meanings with something in common are brought together under the same heading. However, as anyone who has ever reorganized a paper several times will realize, an outline organization is only adequate for one hierarchical grouping, while in fact the common elements existing between various meanings of a word call for a complex cross-classification. That is, the common elements within and between various meanings of a word are many, and any one outline designed to get some of these together under common headings must at the same time necessarily separate other common elements, equally valid from some other point of view. By making the present memory network a general graph,

rather than a tree (the network equivalent of an outline), and by setting up tokens as distinct nodes, it becomes possible to loop as many points as necessary back into any single node, and hence in effect to show any and every common element within and between the meanings of a word. The =A notation causes the network-building program to create such a link.

In all this, it is clear that not only dictionary definitions but also much of the everyday knowledge of the person doing the coding are being tapped and represented in the memory model being built up. For instance, the reader will already have noticed that a numeral is suffixed to the end of some words (a "1" is to be assumed whenever no such numeral appears). This is simply because it is convenient to have each sense of a word named distinctly within the memory, in order to be able to use these in building other configurations. This means that a person building such configurations for input to the model must always decide which possible sense is intended for every token, and use the appropriate suffix.

In attempting to encode dictionary definitions it was found that the memory must provide a mechanism for stating that certain nodes in the immediate definition plane of a type node are variable parameters. A value for one of these parameters will be provided only when the word in whose concept the parameter symbol appears is used in text. Other words within that surrounding text will then form certain parts of the current word's concept; the parameter symbols tell how. To accomplish this, parameter symbols are of three kinds, corresponding to certain ways that other words in text may be related to the word to which the parameter symbols belong. *S* is the parameter symbol whose value is to be any word related to the present word as its subject; *D* is the parameter symbol whose value is to be any word related to the present word as its direct object; and *M* is the parameter symbol whose value is to be any word that the present word directly modifies.

Therefore, to include a parameter symbol in a word's definition plane is to state where within that concept related subjects, objects, and modificands are to be placed, if one or more of these is provided by the text in which the present word is used. For example, when the verb "to comb" is defined by the phrase, "to put a comb through (hair), to get in order," this definition is saying that, when used in text, the verb "to comb" is likely to have an object, which is then to be integrated into its meaning in a certain place, namely, as the object of the node "through." In coding the above definition of "to comb," the object parameter symbol, D, would be used as a sort of "slot" to hold a place for this object until "comb" is actually used in text. It is important not to confuse the sense in which D refers to some object of "comb" and the sense in which there are object links within a plane. D always refers to an object of the word in whose defining plane it appears, while its placement in that plane—indicated by the kind of link from some other token node to it—is another matter. For example, notice in Figure 1a, in the plane for "plant 3," the symbol D (which happens also to have been labeled by = B). This D symbol has been placed as the subject of "in 9," but it is still a D, because it refers to any direct object of the verb "to plant." The symbol D specifies that any such object of "plant" is to be integrated into the meaning of "plant 3" at the place where the D is placed.

A dictionary definition, in addition to stating where within a concept particular sorts of parameter value information are to be "placed," may offer one or more clue words about what such information is likely to be. Thus, in the definition of "to comb" quoted above we are told that its direct object is likely to be "hair."

Clue words play several roles in the memory model, one of which corresponds approximately to the role that transformational linguists ascribe to "selectional restrictions." In other words, the material comprising a full word concept in the memory model can be viewed as consisting of two sorts of information. On the one hand, there is information about the content of the concept itself; on the other there is information about what that con-

cept is likely to combine with when the word is used in text. This latter information is represented by the clue words associated with its parameter symbols. It is significant that this same distinction has been identified in verbal association studies, the associations subjects give to words being divided into paradigmatic (content information) and syntagmatic (parameter clue information) (see, for example, Deese, 1962). Ervin (1961) has shown that the number of content associations, relative to syntagmatic associations, given by young children steadily increases with age.

In the versions of the memory model used in the programs to be described in this paper, clue words have been sought and coded only reluctantly, both they and the parameter symbols having initially been included only because the sort of information comprising them was embarrassingly present in some dictionary definitions. However, it turns out that parameter symbols of some kind play a very crucial role in any such memory, because they make it possible to recognize that two different ways of stating the same thing are in fact synonymous. (See Quillian, 1966.)

As a final point, we note that the model's range readings on tags, together with its ability to form disjunctive sets of attributes, provide it with a ready facility for representing information having a great deal of vagueness. This is essential. It is the very vagueness of the meaning of most language terms that makes them useful—indeed, speech as we know it would be completely impossible if, for instance, one had to specify exactly which machines he made reference to every time he said "machine," and similarly, for every other term whose meaning contains some ambiguity.

To summarize, the memory model, together with the process by which dictionary information is encoded into it, are such that what begins as the English definition of a word seems better viewed after encoding as a complexly structured bundle of attribute values—a full concept, as defined above—whose total content typically extends to an enormous size and complexity throughout the memory. Over all, the memory is a complex network of attribute-value nodes and labeled associations between them. These associations create both within-plane and between-plane ties, with several links emanating from the typical token node, and many links coming into almost every type node.

Performance of the Search Program

It will be recalled that the present program is designed to compare and contrast the meaning of any two word-concepts in the memory store, and then to generate English text to express each of its findings. Notice that this is not the same task as merely using the two words in sentences, a vastly simpler job, for which one need not even consider the semantic concepts associated with the words (Yngve, 1960).

The actual processing system is made up of three separate programs. The first of these transforms input data (definitions which have been encoded as described in the last section) into IPL form and interlaces these to form the total memory model. This program will not be considered further here. The second program compares and contrasts the two given word concepts. It puts out anything found, but in a form expressed in the model memory's own internal language of nodes and links. The third program takes these findings one at a time, and for each generates English text sufficient to express its meaning. Thus, this third program states (in a sort of "me Tarzan, you Jane" style of English) each similarity or contrast of meaning that the second program has found between the two given words.

It is in the operation of the second program, the comparing and contrasting of two concepts, that the interlocking, token-type structure of the overall memory begins to pay off. For, in order to do this job, it is no longer necessary in such a memory to line up some representation of each of the two concepts side by side and try to compare them. Instead, the entire investigation is simply a matter of searching for points in the memory at which the two full concepts intersect (recall how a full concept was defined in the first section above). To see how this is accomplished, recall that the entire memory is a network of nodes

and connecting links. Beginning with the two nodes that the program is given to compare (the two patriarch words), this program works alternately on one full word concept and then the other, moving out node by node along the various tokens and types within each. While it will be convenient to visualize this as creating two slowly expanding spheres of activated nodes around each patriarch, actually there is no geometric significance to the expansion of a concept; the nodes in one concept may be located anywhere in the memory model.

The program simulates the gradual activation of each concept outward through the vast proliferation of associations originating from each patriarch, by moving out along these links, tagging each node encountered with a special two-part tag, the "activation tag." Part of this tag always names the patriarch from which the search began, that is, the name of the concept within which the current node has been reached. Now, the program detects any intersection of meaning between the two concepts simply by asking, every time a node is reached, whether or not it already contains an activation tag naming the other patriarch, that is, showing that this node has previously been reached in the tracing out of the other concept. If there is no such tag, the program next checks to see if there is already an activation tag naming the current patriarch, that is, indicating that this node has been reached previously in tracing out this same concept. If so, the program must take account of this, to inhibit retracing out from the node again and hence repeating its effort, perhaps getting into a loop. Only if neither such tag is found is the node tagged, and further search leading to the nodes it points to considered legitimate.

The second part of each activation tag is the name of the "immediate parent" of the current node,—the node at which the associative link leading directly to it originated. Thus, the "activated" areas of the memory are turned from a one-way network into a two-way network, and, whenever a tag from the opposite patriarch is found, these "immediate parent" parts of activation tags permit the program to trace back "up" from the intersection node to the two patriarchs. This produces two paths, except when the intersection node is one of the

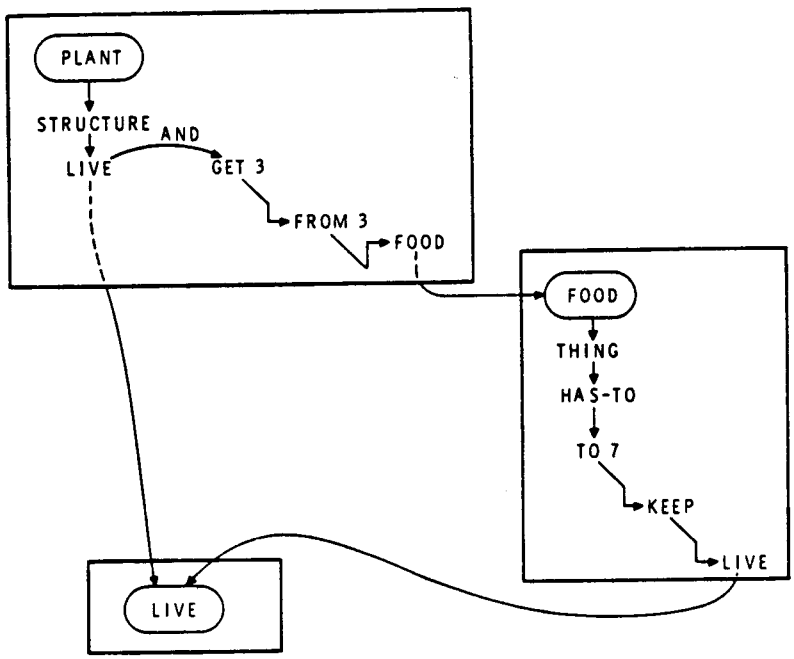

FIG. 2a. Two Paths Direct from Plant to Live.

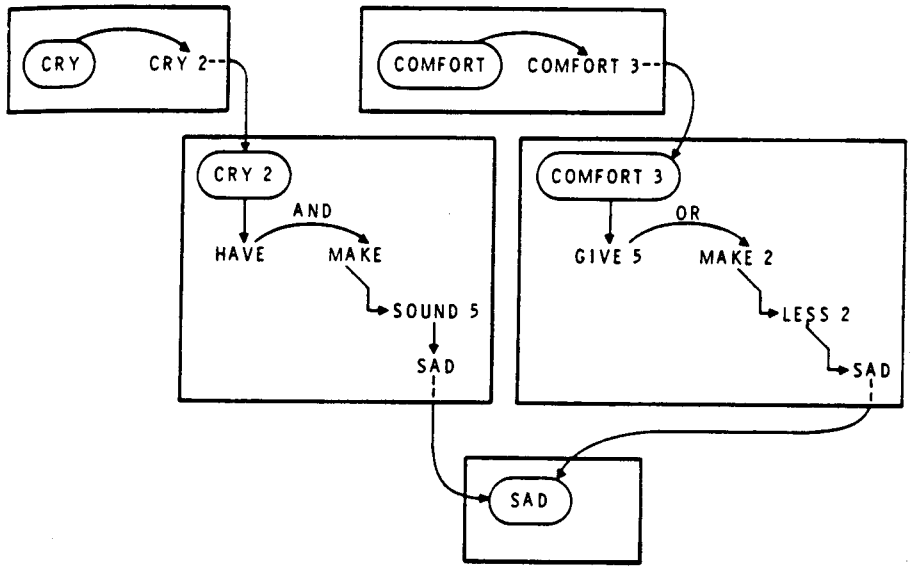

FIG. 2b. A Path from "Cry" and a Path from "Comfort" Which Reach the Same (that is, an intersection) Node.

patriarchs, in which case only a single path is needed, leading from one patriarch directly to the other. Examples of such paths and pairs of paths occur in Figures 2a and 2b, respectively. The paths from a patriarch to an intersection node produced by the second program should not be confused with the "activation" it makes from each patriarch. While this activation is equivalent to an expanding "sphere," a path is only one particular "line" from the center of the sphere to some point within it, one at which it intersects the other full concept's "sphere."

Expanding the two concepts alternately is extremely important; in effect this makes both concepts into searchers for each other, and gives both the maximal number of targets to look for at any given stage of the search.

Performance of the Sentence Generator and Overall Program

The third program, which generates a piece of text to express each path given it by the second program, produces output of the sort illustrated in Table 1. (In this table the paths which the third program has been given to work on are omitted, although the paths for examples 1 and 2 are those of Figure 2.)

The most important point about the sentence producer is that there would seem to be considerable justification for considering it, when taken in conjunction with the first two programs, as an inference maker rather than just a retriever of information. From a relatively small amount of input data, the overall program will indeed derive a very large number of implicit assertions (see calculations below), and make each such assertion explicit in the form of English text. As an example of its most interesting type of "inferential" behavior to date, the reader's attention is directed to the output shown in Table 1 as example 2B. The path that this output expresses is the longer of those shown in Figure 2a. As can be seen from a study of Figure 2a, this kind of performance is made possible by the fact that the memory model closely interconnects related information which has been fed in from a great many different definitions, so that, in order to answer some particular question, the search program can trace out a "plane-hopping" path. While a path lying completely within one plane (except for its

TABLE 1

EXAMPLE OF OUTPUT FROM THE CURRENT PROGRAM
(Paths have been omitted, but see Figure 2)

Example 1. Compare: CRY, COMFORT
A. Intersect: SAD
(1) CRY2 IS AMONG OTHER THINGS TO MAKE A SAD SOUND.[4]
(2) TO COMFORT3 CAN BE TO MAKE2 SOMETHING LESS2 SAD.
(Note that the program has selected particular meanings of "cry" and "comfort" as appropriate for this intersection. The path on which this output is based is shown in Figure 2b.)

Example 2. Compare: PLANT, LIVE
A. 1st Intersect: LIVE
(1) PLANT IS A LIVE STRUCTURE.
B. 2nd Intersect: LIVE
(1) PLANT IS STRUCTURE WHICH GET3 FOOD FROM AIR. THIS FOOD IS THING WHICH BEING2 HAS-TO TAKE INTO ITSELF TO7 KEEP LIVE.
(The paths which these two replies express are shown in Figure 2a.)

Example 3. Compare: PLANT, MAN
A. 1st Intersect: ANIMAL
(1) PLANT IS NOT A ANIMAL STRUCTURE.
(2) MAN IS ANIMAL.
B. 2nd Intersect: PERSON
(1) TO PLANT3 IS FOR A PERSON SOMEONE TO PUT SOMETHING INTO EARTH.
(2) MAN3 IS PERSON.
(Here the program is treating "person" as an adjective modifier of "someone.")

Example 4. Compare: PLANT, INDUSTRY
A. 1st Intersect: INDUSTRY
(1) PLANT2 IS APPARATUS WHICH PERSON USE FOR5 PROCESS IN INDUSTRY.

Example 5. Compare: EARTH, LIVE
A. 1st Intersect: ANIMAL
(1) EARTH IS PLANET OF7 ANIMAL.
(2) TO LIVE IS TO HAVE EXISTENCE AS7 ANIMAL.

Example 6. Compare: FRIEND, COMFORT
A. 1st Intersect: PERSON
(1) FRIEND IS PERSON.
(2) COMFORT CAN BE WORD TO4 PERSON.

Example 7. Compare: FIRE, BURN
A. 1st Intersect: BURN
(1) FIRE IS CONDITION WHICH BURN.
B. 2nd Intersect: FIRE
(1) TO BURN2 CAN BE TO DESTROY2 SOMETHING BY4 FIRE.
C. 3rd Intersect: BURN
(1) FIRE IS A FLAME CONDITION. THIS FLAME CAN BE A GAS TONGUE4. THIS GAS IS GAS WHICH BURN.
(The sentence producer starts a new sentence whenever it needs to say something more about something it has used adjectively.)

[4] "AMONG OTHER THINGS" and "CAN BE" are canned phrases which the program inserts when the next thing it is going to mention is one out of a set of things recorded in its memory. At one point, the program was programmed to insert "AMONG OTHER THINGS" whenever it was about to assert one fact out of such a set. We expected this to make its output have a proper, scientifically cautious ring. However, where it had been saying, (rather clodishly, we felt) "TO CRY IS TO MAKE A SAD SOUND," it now said: "TO CRY, AMONG OTHER THINGS, IS, AMONG OTHER THINGS, TO MAKE, AMONG OTHER THINGS, A, AMONG OTHER THINGS, SAD SOUND." (!) In short, it turns out that if the program is really made to hedge whenever it knows more than it is going to say, one sits around the console all day waiting for it to get around to saying anything. This may not be such a bad simulation of certain individuals, but wasn't what we had had in mind. Thus, the program is now severely restricted as to just when it can hedge. Science marches on.

TABLE 1—*Continued*

Example 8. Compare: BUSINESS, COMFORT
A. 1st Intersect: PERSON
(1) BUSINESS5 IS ACT3 WHICH PERSON DO.
(2) COMFORT2 IS CONDITION3 WHICH PERSON HAVE NEED4.
(The code contains information indicating that "person" should be plural here, but the sentence producer does not yet make use of this information.)
B. 2nd Intersect: PERSON
(1) BUSINESS5 IS ACT3 WHICH PERSON DO.
(2) COMFORT CAN BE WORD TO4 PERSON.

Example 9. Compare: MAN, BUSINESS
A. 1st Intersect: PERSON
(1) MAN3 IS PERSON.
(2) BUSINESS CAN BE ACTIVITY WHICH PERSON MUST DO WORK2.
(Something wrong here. I believe a miscoding in the input data.)
B. 2nd Intersect: GROUP
(1) MAN2 IS MAN AS9 GROUP.
(2) BUSINESS2 IS QUESTION3 FOR ATTENTION OF GROUP.

Example 10. Compare: MAN, LIVE
A. 1st Intersect: ANIMAL
(1) MAN IS ANIMAL.
(2) TO LIVE IS TO HAVE EXISTENCE AS7 ANIMAL.
B. 2nd Intersect: LIVE
(1) MAN IS A LIVE BEING2.

terminal points) amounts only to a representation of some piece of the information put into the memory, a "plane-hopping" path represents an idea that was implied by, but by no means directly expressed in, the data that was input. By analogy, suppose we fed a machine "A is greater than B," and "B is greater than C." If then, in answer to the question "what is A greater than?" the machine responded "B," we would not want to call this an inference, but only a "recall." However, if it went on to say, "A is also greater than C," then we would say that it had made a simple inference. The kind of path that we have been calling "plane-hopping" is exactly the representation of such an inference, since it connects information fed in in one definition with that fed in in another. But the fact that our planes are not simple propositions but rather sizeable configurations, every node of which provides the possibility of branching off to another plane, means that the number of "inferential" paths becomes very large as paths of any appreciable length are considered. Moreover, the possibility that a path may contain fragments from several planes, would seem to indicate clearly that the inferences need not be at all simple, although we do not as yet have actual computer output with which to demonstrate this very conclusively.

Assuming a "complete" semantic memory —one in which every word used in any definition also has a definition encoded—a concept fans out very rapidly from its patriarch. It appears that in such a full model memory the average node would branch to at least three other nodes, considering both its ties to tokens and to its type, if it is itself a token. This means that the average number of paths of, say, up to ten nodes in length emanating from any type of node would be over 88,000, each of which would require at least one unique sentence to express. This is to be compared to 2,046 paths emanating from such a type node if no token-to-type links are available. Another way to look at the potential of a memory store such as the theory specifies is to compute what the present programs could generate if one could get, say, definitions of 850 words encoded and stored in a model memory. There would then be 360,000 word pairs to ask it about. Since at a conservative estimate a memory model this size would provide ten nontrivial semantic connections, and hence sentences or sentence sets, between the average word pair, the present programs would have the capability to generate well over three-and-one-half million short batches of text to express this total conceptual knowledge, ignoring all that information present only

in longer paths. Definitions of 850 words comprise considerably more information than one could model in the core of today's computers, but calculations such as these seem relevant in evaluating the potential of the model as a general theory of long-term conceptual memory.

While a path represents an idea, it is up to the sentence-producing program to get that idea expressed in English. Thus this program must check a path for restriction tags and other features which make it necessary to insert words such as "not" or "among other things" into the sentence generated to express its meaning.

In attempting to express the meaning of a path, this program also deletes, rearranges, and adds words to those given in the path. It works not only with nodes mentioned in the path itself but sometimes looks *around* these nodes in the memory model to retrieve additional information and to check on things it considers saying.

In expressing a complex path such as that of Figure 2a, this text-producing program realizes when the capability of its sentence grammar is being exceeded and starts a new sentence. (See for example, 7.c.1 of Table 1.) Unfortunately, it does this rather often, and a more powerful grammar would clearly be one which, instead of the two sentences shown in Table 1 as example 3.A.1 and 3.A.2, would produce the single sentence: "A plant is not an animal but a man is." Some of the minor improvements of this sentence over the two which the program now produces would not be difficult to program, but the unification of the two paths into one is a bit more complicated. Clearly, this involves something very close to what Chomsky calls transformations.

In summary, although we have not described the operation of the sentence producer in detail, it should be clear that it has little in common with other sentence generation programs, and, in fact, its whole philosophy is contradictory to a good part of the spirit of modern linguistics, inasmuch as this attempts to treat syntactic facts in isolation from semantic ones. Thus other sentence generation programs produce sentences that, in syntax, are grammatical, but which are in meaning either completely random (Yngve, 1960), or random permutations of the "dependency" constraints imposed by an input text (Klein and Simmons, 1963). The program is also designed in complete contradiction to the subordinate place for semantic information that the formulation of Katz and Postal (1964) would seem to imply for a performance model.[5] As a theory, the program implies first that a person has something to say, expressed somehow in his own conceptual terms (which is what a "path" is to the program), and that all his decisions about the syntactic form that a generated sentence is to take are then made in the service of this intention. The sentence producer works entirely in this fashion, figuring out grammatical properties of sentences only as these are needed to solve the problem of expressing a path given to it by the search program.

Thus far, the programs have only been tested on very small model memories, built from no more than 50 or 60 definitions (about 5,000 IPL cells), and on only a few such memories (see Table 2). A small total memory means that most branches of the proliferating search of a concept are always getting cut short by reaching a type node for which no definition has yet been encoded. One of the most surprising findings from running the program has been that even with this relative paucity of overall information, the program almost always succeeds in finding some intersections of meaning. Actually, Table 1 lists only a selected sample of the program's output for each compared pair of words; there are usually five or six pairs of sentences generated for each problem pair given to it, although most of these are only trivial variations of a couple of basic sentences such as those we have selected for Table 1. The larger the model memory, the greater

[5] While on the one hand the transformationalists explicitly deny that their work is a model of performance, at the same time they often seem willing to draw psychological conclusions from their models. (See, for example, Katz and Postal, 1964, pp. 1–2; Chomsky, 1965, pp. 139–141).

TABLE 2

WORDS WITH DEFINITIONS ENCODED FOR USE IN MODEL MEMORIES[6]

instrument	cause	live
insurance	attack	level
invent	argue	lift
interest	business	letter
iron	burn	learn
ice	build	leather
idea	bread	land
friend	behave	kiss
develop	cry	know
event	country	laugh
earth	desire	light
exist	sex	language
drink	plant	law
fire	family	lead
flame	meal	jelly
experience	animal	journey
fact	food	jump
comfort	man	judge
cloth		

NOTE: Space limitations have so far required that definitions of no more than twenty of these words be used to constitute a model memory during a given series of word comparisons. Since this paper was written, almost all of the 850 words of basic English have been encoded, but not yet run in the program.

the number of search branches that remain active, so that the search program becomes able to unearth a great many more semantic connections at a relatively shallow depth beneath any two patriarchs. Ultimately this can only improve the program's performance, although it may also require that more concern be given to directing searches than is so far the case. At present, but for one exception, a search just "progressively proliferates" along all possible branches from the two patriarchs (until it has covered a given number of nodes, such as 400).

The one exception to this blind, "breadth first," search occurs whenever two concepts are found to intersect on a word used prepositionally, such as "for 5" in the concept "plant 2." Instead of treating this as a substantive semantic intersection, the search program merely concentrates an immediate burst of search activity out from the two tokens of the preposition. The reasoning here is simply that, while a match on such a word is not in itself sufficient to be treated as a significant conceptual similarity, it is a good bet to examine immediately the subjects, objects, and modifiers of such prepositions, rather than continue the usual search schedule which normally would not get to these nodes for some time. Unfortunately there is not yet enough evidence available to assess the value of this search heuristic, since its effectiveness, if any, will not show up until the memory model is relatively large.

Discussion

In the current programs, all activation tags are erased after comparison of two-word concepts is completed, but in order to illustrate the generality of the memory model their relevance to two other phenomena will be mentioned. The first of these is the state a person gets into by reading part of a text; namely he gets "in context." In this state he will, for example, be able to decide which of several possible meanings of a new word that he encounters in the text fits the context. To explain this, let us suppose that the person's memory is indeed organized and utilized as is our model memory. Suppose, for example, that the text is about baseball. As the subject has been reading, he has been firing activation "spheres" from the patriarch nodes corresponding to many of the words in the text. The activation tags applied in this manner are not immediately erased, so that they accumulate throughout much of the memory, on nodes such as "batter," "ball," "pitcher," and so on. Now, upon encountering, say, the word "strike," the reader fires one activation sphere from the

[6] It is hoped that the program can be tested on somewhat larger memory models in the not too distant future, although it would also be interesting to attempt to improve the dictionary entries from which data are taken. So far definitions given in Ogden (1942) have been used. This was selected in order to make sure that definitions loop back into one another as often as possible, and because its definitions are short. It now seems there is no worry at all about having enough intersections, and several other dictionaries have been investigated as possible sources of input data. *Funk and Wagnalls* (1959), although not consistent on this point, makes much the most thorough effort to arrange the various meanings of a word into some sort of outline, hence indicating which meanings have something in common.

type node heading its baseball meaning, another from the one heading its labor union meaning. Clearly, intersections will pile up very quickly beneath one of these meanings, and much more slowly beneath the other. The ambiguity is resolved almost instantaneously; if the reader is a human, he would say that one meaning is "in context," the other not. (An experiment demonstrating the use of the model for such automatic meaning resolution is described in Quillian, 1966.)

The considerable effort that has been invested in building automatic parsing programs (see Bobrow, 1963), all assumes that natural language can be dealt with without using any such semantic contextual information, or at least without using it until after all parsing has been accomplished by purely syntactic information. This may be so, but it seems to me more likely that to attack the problem in this way is to pose the wrong, and probably an insoluble, problem. I suspect that successful mechanical language processors must stay much closer to the way humans process language, in which syntax and semantics are surely interwoven at every stage.

Katz and Foder (1963) and Katz and Postal (1964) propose to describe the "structure of a semantic theory," with almost exactly the same machinery Chomsky has used to advantage for describing syntactic structure. However, aside from also assuming that stored semantic information is always to be placed beneath the syntactic information associated with a word, Katz and his coworkers seem to offer us little more about how this information is to be represented than to assert that it can always be arranged into a single tree structure of "markers" and "distinguishers." We have already described (above) why such a single outline type of organization will not suffice to capture the information which people know and use in dealing with words.

As a theory of semantics our memory model has greater flexibility and expressiveness, and correspondingly greater complexity and cumbersomeness to process, than that of formats generally considered, such as set theoretic or symbolic logic frameworks. However, various attempts to translate English into such terminologies have as yet not met with any general success, and in some cases it is clear that there are formidable difficulties in the way (Darlington, 1964). Others have proposed isolating a set of semantic "elements," either by linguistic methods (Lamb, 1964) or psychologically, by analyzing word meanings into Titchnerian sensory units (Quillian, 1961, 1962a). Even if some such reduction is possible, however, its relation to the way people really encode semantic information, and even its usefulness to any sort of empirical semantic investigation, seem unclear. The problem of semantics, it seems to me, lies more in how to represent and arrange information than in how to discover more of it, a problem to which dictionary and encyclopedia compilers have applied themselves with some diligence.

In any case, the model of memory proposed here is both purely semantic and very elaborate, and one may speculate about what changes will be required if it is to serve as a basis for the explanation and simulation of other functions that people perform with the aid of long-term memory. Programming such new tasks is crucial to establish the generality of the memory model. Doing so would further test the current features of the model, and get at some of its properties that are hardly tested at all by the current programs, such as the parameters S, D, and M. As the model is tested by programming such tasks, changes in its details are inevitable. The pertinent question is whether or not a model essentially similar to this one is likely to prove useful for supporting the simulation of other memory-dependent functions, and, of course, for guiding other research on memory functions.

In summary, four key assumptions about word concepts stored in memory have been made: that the information in them is large, differentially accessible, exceedingly rich in expressive power, and yet composed of units that represent properties. The realization of these features in an explicit

model has required a complicated network of associative links, plus a number of other devices. The resulting model has been tested for its ability to allow the meaning of English text to be encoded accurately and, once a model memory has been built up in this way, for its ability to support simulation of two behaviors: the recognition of semantic similarities and contrasts, and the expression of these in sentences. It has been asserted that the memory model and the programs processing it constitute a theory of the general structure and corresponding uses of human memory, although we are aware that a behavioral theory couched in computer terms is sufficiently unfamiliar to make some psychologists feel uneasy.

For those who, like this author, are disposed to consider this model a psychological theory, a great many new problems open up. In addition to exploring the sufficiency of the memory model for other types of behavior, there is the task of discovering how variations in such a structure affect performance, and subsequently of correlating specific features of the model with individual variations of subjects' behavior. Some computer-framed theories have, at least for specific problem-solving behaviors, reached this level of development (see, for example, Newell, Shaw, and Simon, 1962; Simon and Feigenbaum, 1964; or Simon and Kotovsky, 1963). At that point the "information-processing" methodology merges with the main stream of psychological research. For those whose interest is primarily in artificial intelligence per se, let me conclude by suggesting that further advances in reproducing human performance with a computer critically depend on giving such programs memories which can effectively provide them with a "knowledge of the world."

References

Banerji, R. B. A language for the description of concepts. Unpublished dittoed paper, Systems Research Center, Case Institute of Technology, 1964.

Bartlett, F. C. *Remembering, a study in experimental and social psychology.* Cambridge: Cambridge University Press, 1932.

Bobrow, D. G. Syntactic analysis of language by computer—a survey. *Proc. Fall Joint Computer Conference*, 1963, 24, 365–387.

Bruner, J. S., Goodnow, J. J., & Austin, C. A. *A study of thinking.* New York: John Wiley, 1956.

Chomsky, N. *Aspects of the theory of syntax.* Cambridge: M.I.T. Press, 1965.

Cliff, N. Adverbs as multipliers. *Psychol. Rev.*, 1959, 66, 27–44.

Darlington, J. Translating ordinary language into symbolic logic. Memorandum MAC-M-149, Project MAC, Massachusetts Institute of Technology, 1964.

Deese, J. On the structure of associative meaning. *Psychol. Rev.*, 1962, 69, 161–175.

Ervin, S. M. Changes with age in the verbal determinants of word association. *Amer. J. Psychol.*, 1961, 74, 361–372.

Funk and Wagnalls new "standard" dictionary of the English language. New York: Funk and Wagnalls, 1959.

Hunt, E. B. *Concept learning: An information processing problem.* New York: John Wiley, 1962.

Katz, J. J., & Foder, J. A. The structure of a semantic theory. *Language*, 1963, 39, 170–210.

Katz, J. J., & Postal, P. M. An integrated theory of linguistic descriptions. Cambridge: The M.I.T. Press, 1964.

Kelly, G. *The psychology of personal constructs: Volume I.* New York: W. W. Norton, 1955.

Klein, S. Automatic paraphrasing in essay format. SP-1602/001/00, System Development Corporation, Santa Monica, 1964.

Klein, S., & Simmons, R. F. Syntactic dependence and the computer generation of coherent discourse. *Mechanical Translation*, 1963, 7, 50–61.

Kuno, S., & Oettinger, A. Lecture notes for course in language data processing. Harvard Summer School, Harvard University, 1964.

Lamb, S. The sememic approach to structural semantics. In K. A. Romney & R. D'Andrede (Eds.) Transcultural studies in cognition. *Amer. Anthropol.*, 1964, 66, Part 2, 1–74.

Miller, G. A. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychol. Rev.*, 1956, 63, 81–96.

Newell, A. (Ed.) IPL-V programmer's reference manual. Memorandum RM-3739-RC, RAND Corporation, Santa Monica, California, 1963.

Newell, A., Shaw, J. C., & Simon, H. A. The processes of creative thinking. In H. E. Gruber, G. Terrell, & M. Wertheimer (Eds.) *Contemporary approaches to creative thinking.* New York: Atherton Press, 1962, Pp. 63–119.

Ogden, C. K. *The general basic English dictionary.* New York: W. W. Norton, 1942.

Osgood, C. E. On understanding and creating sentences. *Amer. Psychol.*, 1963, 18, 735–751.

Osgood, C. E., Suci, G. J., & Tannenbaum, P. H. *The measurement of meaning.* Urbana: University of Illinois Press, 1957.

Piaget, J. *The psychology of intelligence.* (Tr. M. Cook, & D. E. Berlyne) London: Routledge & Kegan Paul, 1950.

Quillian, R. A design for an understanding machine. Paper presented at a colloquium: Semantic problems in natural language. King's College, Cambridge University, September, 1961.

Quillian, R. A revised design for an understanding machine. *Mechanical Translation*, 1962, 7, 17–29. (a)

Quillian, R. A semantic coding technique for mechanical English paraphrasing. Internal Memorandum of the Mechanical Translation Group, Research Laboratory of Electronics, Massachusetts Institute of Technology, August 1962. (b)

Quillian, R. A notation for representing conceptual information: An application to semantics and mechanical English paraphrasing. SP-1395, System Development Corporation, Santa Monica, 1963.

Quillian, R. *Semantic Memory.* Unpublished doctoral dissertation, Carnegie Institute of Technology, 1966. Also Report AFCRL-66-189, Bolt, Beranek, and Newman, Cambridge, Massachusetts, October, 1966.

Quillian, R., Wortman, P., & Baylor, G. W. The programmable Piaget: Behavior from the standpoint of a radical computerist. Unpublished dittoed paper, Carnegie Institute of Technology, 1965.

Raphael, B. A computer program which "understands." *Proc. AFIPS*, Fall Joint Computer Conference, 1964, 577–589.

Reitman, W. R. *Cognition and thought: An information processing approach.* New York: John Wiley, 1966

Simmons, R. F. Synthetic language behavior. *Data Processing Management*, 1963, 5 (12), 11–18.

Simon, H. A., & Feigenbaum, E. A. An information-processing theory of some effects of similarity, familiarization, and meaningfulness in verbal learning. *J. verb. Learn. verb. Behav.*, 1964, 3, 385–397.

Simon, H. A., & Kotovsky, J. Human acquisition of concepts for sequential patterns. *Psychol. Rev.*, 1963, 70, 534–546.

Yngve, V. H. A model and an hypothesis for language structure. *Proc. Amer. Phil. Soc.*, 1960, 104, 444–466.

(Manuscript received November 28, 1966)

Pictorial form is the possibility that things are related to one another in the same way as the elements of the picture.

LUDWIG WITTGENSTEIN