Taxonomic Lattice Structures for Situation Recognition

William A. Woods
Bolt Beranek and Newman Inc.
50 Moulton Street
Cambridge, MA 02138

1. The Role of a Knowledge Network for an Intelligent Machine

The kinds of intelligent computer assistants that we would like to be able to construct are very much like intelligent organisms in their own right. Imagine for a moment an intelligent organism trying to get along in the world (find enough food, stay out of trouble, satisfy basic needs, etc.). The most valuable service played by an internal knowledge base for such an organism is to repeatedly answer questions like "what's going on out there?", "can it harm me?", "how can I avoid/placate it?", "Is it good to eat?", "Is there any special thing I should do about it?", etc. To support this kind of activity, a substantial part of the knowledge base must be organized as a recognition device for classifying and identifying situations in the world. The major purpose of this situation recognition is to locate internal procedures which are applicable (appropriate, permitted, mandatory, etc.) to the current situation.

In constructing an intelligent computer assistant, the roles of knowledge are very similar. The basic goals of food getting and danger avoidance are replaced by goals of doing what the user wants and avoiding things that the machine has been instructed to avoid. However, the fundamental problem of analyzing a situation (one established either linguistically or physically or by some combination of the two) in order to determine whether it is one for which there are procedures to be executed, or one which was to be avoided (or one which might lead to one that is to be avoided), etc. is basically the same. For example, one might want to instruct such a system to remind the user in advance of any upcoming scheduled meetings, to inform him if he tries to assign a resource that has already been committed, to always print out messages in reverse chronological order (when requested), to assume that "the first" refers to the first day of the upcoming month in a future scheduling context and the first day of the current month in a past context, etc.

The principal role of the knowledge network for such a system is essentially to serve as a "coat rack" upon which to hang various pieces of advice for the system to execute. Thus the notion of procedural attachment becomes not just an efficiency technique, but the main purpose for the existence of the network. This does not necessarily imply, however, that the procedures involved consist of low-level machine code. They may instead, and probably usually will, be high level specifications of things to be done or goals to be achieved. The principal structure that organizes all of these procedures is a conceptual taxonomy of situations about which the machine knows something.

To support the above uses of knowledge, an important characteristic required of an efficient knowledge representation seems to be a mechanism of inheritance that will permit information to be stored in its most general form and yet still be triggered by any more specific situation or instance to which it applies. Moreover, the nodes in the network (or at least a major class of nodes) should be interpretable as situation descriptions. One of the most fundamental kinds of information to be stored in the knowledge base will be rules of the form "if <situation description> is satisfied then do <action description>", or "if <situation description> then expect <situation description>". Situation descriptions are in general characterizations of classes of situations that the machine could be in. They are not complete descriptions of world states, but only partial descriptions that apply to classes of world states. (The machine should never be assumed or required to have a complete description of a world state if it is to deal with the real world.) A situation in this partial sense is defined by the results of certain measurements, computations, or recognition procedures applied to the system's input. Examples of situations might be "You have a goal to achieve which is an example of situation Y" "You are perceiving an object of class Z", "The user has asked you to perform a task of type W", etc.

More specific situations might be: "trying to schedule a meeting for three people, two of which have busy schedules", "about to print a message from a user to himself", "about to refer to a date in a recent previous year in a context where precision but conciseness is required".

The major references to this conceptual taxonomy by the intelligent machine will be attempts to identify and activate those situation descriptions that apply to its current situation or some hypothesized situation in order to consider any advice that may be stored there. Note that "considering advice of type X" is itself an example of a situation, so that this process can easily become recursive and potentially unmanageable without appropriate care.

Conceptually, one might think of the process of activating all of the descriptions that are satisfied by the current situation as one of taking a description of the current situation and matching it against descriptions stored in the system. However, there are in general many different ways in which the current situation might be described, and it is not clear how one should construct such a description.

Moreover, until it is so recognized, a situation consists of a collection of unrelated events and conditions. The process of recognizing the elements currently being perceived as an instance of a situation about which some information is known consists of discovering that those elements can be interpreted as filling roles in a situation description known to the system. In fact, the process of creating a description of the current situation is very much like the process of parsing a sentence, and inherently uses the knowledge structure of the system like a parser uses a grammar in order to construct the appropriate description. Consequently, by the time a description of the situation has been constructed, it has already been effectively matched against the descriptions in the knowledge base.

2. Parsing Situations

As suggested above, the process of recognizing that a current situation is an instance of an internal situation description is similar to the process of parsing a sentence; although considerably more difficult due to a more open ended set of possible relationships among the "constituents" of a situation. That is, whereas the principal relationship between constituents in sentences is merely adjacency in the input string, the relationships among constituents of a situation may be arbitrary (e.g. events preceding one another in time, people, places, or physical objects in various spatial relationships with each other, objects in physical or legal possession of people, people in relationships of authority to other people, etc.) However, the basic characteristic of parsers, that the objects recognized are characterized as structured objects assembled out of recognizable parts according to known rules of assembly, is shared by this task of situation recognition.

Note that it is not sufficient merely to characterize a situation as a member of one of a finite number of known classes. That is, where it is not sufficient for a parser to simply say that its input is an example of a declarative sentence (one wants to be able to ask what the subject is, what the verb is, whether the sentence has past, present or future tense, etc.), in a similar way it is insufficient to merely say that an input situation is an example of someone doing something. One must generate a detailed description of who is doing what to whom, etc.

It is also not sufficient to characterize a situation as a single instance of an existing concept with values filled in for empty slots. In general, a situation description must be a composite structured object, various subparts of which will be instances of other concepts assembled together in ways that are formally permitted, in much the same way that the description of a sentence is put together from instances of noun phrases, clauses, and prepositional phrases. The specific instance built up must keep track of which constituents of the specific situation fill which roles of the concepts being recognized. Moreover, it cannot do so by simply filling in the slots of those general concepts, since a general concept may have multiple instantiations in many situations. Rather, new structures representing instances of those concepts must be constructed and pairings of constituent roles from the concept and role fillers from the current situation must be associated with each new instance.

3. The Process of Situation Recognition

The process of situation recognition consists of detecting that a set of participants of certain kinds stand in some specified relationship to each other. In general, when some set of participants is present at the sensory interface of the system (immediate input plus past memory), the task of determining whether there is some situation description in memory that will account for the relationships of those inputs is not trivial. If the total number of situation descriptions in the system is sufficiently small, all of them can be individually tested against the input to see if any are satisfied. If the

number of such descriptions is sufficiently large, however, this is not feasible.

Alternatively, if there is some particular participant that by virtue of its type strongly suggests what situation descriptions it might participate in, then an index from this participant might select a more manageable set of situation descriptions to test. Even in this case, however, the number of situations in which the constituent could participate may still be too large to test efficiently. In the most difficult situation, no single participant in the input is sufficiently suggestive by itself to constrain the set of possible patterns to a reasonable number. However, it may still be that the coincidence of several constituents and relationships may suffice, providing that the coincidence can be detected. It is this problem of coincidence detection that I believe to be crucial to solving the general situation recognition problem.

As an example, consider the following fragment of a protocol of a commander giving commands to an intelligent display system:

Cdr: Show me a display of the eastern Mediterranean.
[computer produces display]

Cdr: Focus in more on Israel and Jordan.
[computer does so]

Cdr: Not that much; I want to be able to see Port Said and the Island of Cyprus.

In the first clause of the third command of this discourse, (i.e. "not that much"), there is no single word that is strongly suggestive of the interpretation of the sentence. Moreover, there is nothing explicit to suggest the relationship of this clause to the one that follows the semicolon. The latter, if interpreted in isolation, would merely be a request for a display, or perhaps a succession of two displays, while in the context given, it is a request to modify a previous display.

There are two methods that I believe may be sufficient, either individually or in combination, to model coincidence detection. One is the use of factored knowledge structures that merge common parts of alternative hypotheses. The other involves the use of a markable classification structure in which the individual recognition predicates triggered by the ongoing discourse will leave traces of their having fired, so that coincidences of such traces can be efficiently detected. I have been investigating a structure which I call a "taxonomic lattice", that combines some features of both methods.

3.1 Factored Knowledge Structures

Given a knowledge-based system with large numbers of situation-action rules, where it is infeasible to find the rules that match a given situation by systematically considering each rule, one needs to have some way of reducing the computational load. As mentioned before, one approach is to index the rules according to some salient feature that will be easily detectable in the input situation and can then be used to find a much more limited set of rules to apply. This has been done in many systems, including the LUNAR system for natural language question answering [Woods, 1973, 1977]. In that system, rules for interpreting the meanings of sentences were indexed according to the verb of the sentence and rules for interpreting noun phrases were indexed by the head noun. Although this approach reduces the number of rules that need to be considered, it has several limitations still. The first is that there may be some values of the index key for which there are still a large number of rules to consider. In the case of the LUNAR system, for example, the verb "be" had a large number of rules to account for different senses of the word. Another is that there can be certain constructions for which there is no single easily detected feature that is strongly constraining as to possible meaning. In this case, there is no useful index key that can be used to select a sufficiently constrained set of rules to try.

Another limitation of this indexing approach as the range of language becomes more fluent is that in certain elliptical sentences, the constraining key may be ellipsed, and although one can have the rules indexed by other keys as well, the remaining ones may not sufficiently constrain the set of rules that need to be considered. Finally, even when the set of rules has been constrained to a relatively small set, there is frequently a good deal of sharing of common tests among different rules, and considering each rule independently results in repeating these tests separately for each rule.

One approach to solving all of the above problems is to use what I have been calling a "factored knowledge structure" for the recognition process. In such a structure, the common parts of different rules are merged so that the process of testing them is done only once. With such structures, one can effectively test all of the rules in a very large set, and do so efficiently, but never consider any single rule individually. At each point in a factored knowledge structure, a test is made and some information gained about the input. The result of this test determines the next test to be made. As each test is made and additional information accumulated, the set of

possible rules that could be satisfied by the input, given the values of the tests so far made, is gradually narrowed until eventually only rules that actually match the input remain. Until the end of this decision structure is reached, however, none of these rules is actually considered explicitly. This principle of factoring together common parts of different patterns to facilitate shared processing is the basic technique that makes ATN grammars [Woods, 1970] more efficient in some sense than ordinary phrase structure grammars. It has also been used by the lexical retrieval component of the BBN speech understanding system [Woods et al., 1976; Wolf and Woods, 1977] and accounts for the efficiency of the finite state grammar approach of the CMU Harpy system [Lowerre, 1976]. A recent innovative use of this principle appears in Rieger's "trigger trees" for organizing spontaneous computations [Rieger, 1977].

Whether factored together or not, the task of accessing rules is not a simple one. One problem is that rules don't match the input letter-for-letter: rather, they have variables in them with various restrictions on what they can match. For example a rule might say that whenever an access is made to a classified file, then a record of the person making the request should be made. The description, "an access to a classified file" needs to be matched against the user's request (or some subpart of it) and in that match, the description "a classified file" will be matched against some specific file name. In this kind of situation, there is no natural ordering of the rules, analogous to the alphabetical ordering of words, that will help in finding the rules that are satisfied by the given situation. Nor is a structure as simple as the dictionary tree above adequate for this case.

Another problem is that a given situation may be matched by several rules simultaneously with differing degrees of generality. For example, there may be a rule that says "whenever access is made to a top secret file (more specific than classified), then check the need-to-know status of the user for that information and block access if not satisfied". In the case of a request to a top secret file, both of the above rules must be found, while in the case of an ordinary classified file, only the first should. The actual input, however, will not explicitly mention either "top-secret" or "classified", but will merely be some file name that has many attributes and properties, among which the attribute "classified" is not particularly salient.

3.2 Markable Classification Structures

Another technique that holds promise for situation recognition is the use of a markable classification structure in which coincidences of relatively non-salient events can be detected. The keystone of this approach is a technique that Quillian proposed for modeling certain aspects of human associative memory [Quillian, 1966, 1968]. Quillian's technique of "semantic intersection" consisted of propagating traces of "activation" through a semantic network structure so that connection paths relating arbitrary concepts could be detected. For example, his system was able to connect concepts such as "plant" and "nourishment" by discovering the "chain" equivalent to "plants draw nourishment from the soil". If the appropriate information were in the network, this technique would also find chains of indirect connections such as "Plants can be food for people" and "People draw nourishment from food." The method was capable of finding paths of arbitrary length.

The problem of finding connections between concepts in a knowledge network is like the problem of finding a path through a maze from a source node to some goal node. At the lowest level, it requires a trial and error search in a space that can be large and potentially combinatoric. That is, if one element of the input could be connected to k different concepts, each of which would in turn be connected to k others, and so on, until finally a concept that connected to the goal was discovered, then the space in which one would have to search to find a path of length n would contain k^n paths. However, if one started from both ends (assuming a branching factor of k also in the reverse direction), one could find all the paths of length n/2 from either end in only $2.k^{n/2}$.

If one then had an efficient way to determine whether any of the paths from the source node connected with any of the paths from the goal node, such search from both ends would have a considerable savings. This can be done quite efficiently if the algorithm is capable of putting marks in the structure of the maze itself (or some structure isomorphic to it), so that it can tell when reaching a given node whether a path from the source or the goal has already reached that node. However, without such ability to mark the nodes of the maze, the process of testing whether a given path from the source can hook up with a path from the goal would involve a search through all the paths from the goal individually, and a search down each such path to see if the node at the end of the source path occurred anywhere on that path. If this were necessary, then all of the advantage of searching from both ends would be lost.

The use of the graph structure itself to hold marks is thus critical to gaining advantage from this algorithm. Essentially, the nodes of the graph serve as rendezvous points where paths that are compatible can meet each other. The coincidence of a path from the source meeting a path from the goal at some node guarantees the discovery of a complete path without any path requiring more than a simple test at the corresponding node in the graph as each link is added to the path.

What is needed for situation recognition in a generalization of Quillian's semantic intersection technique in which the source and goal nodes are replaced by a potentially large number of concept nodes, some of which are stimulated by immediate input, and some of which are remembering recent activation in the past. Moreover, what is significant is not just simple paths between two nodes, but the confluence of marks from multiple sources in predetermined patterns. Moreover, unlike Quillian, who considered all connections identically in searching for paths, we will consider marker passing strategies in which marks can be passed selectively along certain links. Recently, Fahlman [1977] has presented some interesting formal machine specifications of Quillian-type spreading activation processes which have this characteristic.

4. The Structure of Concepts

In building up internal descriptions of situations, one needs to make use of concepts of objects, substances, times, places, events, conditions, predicates, functions, individuals, etc. Each such internal concept will itself have a structure and can be represented as a configuration of attributes or parts, satisfying certain restrictions and standing in specified relationships to each other. Brachman [1978] has developed a set of epistemologically explicit conventions for representing such concepts in a "Structured Inheritance Network", in which interrelationships of various parts of concepts to each other and to more general and more specific concepts are explicitly represented. The essential characteristic of these networks is their ability to represent descriptions of structured objects of various degrees of generality with explicit representation of the inheritance relationships between corresponding constituents of those structures. A concept node in Brachman's formulation consists of a set of dattrs (a generalization of the notions of attribute, part, constituent, feature, etc.) and a set of structural relationships among them. Some of these dattrs are represented directly at a given node, and others are inherited indirectly from other nodes in the network to which they are related.

Let us assume that each concept that the system understands is represented as a node in one of these structured inheritance networks. The network, as a whole, then serves as a conceptual taxonomy of all possible "entities" that the system can perceive or understand. Each node in this taxonomy can be thought of as a micro schema for the recognition of instances of that concept. Each has a set of dattrs with individual restrictions and a set of structural conditions that relate the dattrs to one another. These restrictions and structural conditions may themselves be defined in terms of other concepts defined by other micro schemata, and so on until a level of primitively defined, directly perceivable concepts is reached.

Each concept in the taxonomy can be thought of as having a level of abstractness defined as the maximum depth of nesting of its constituent structure. Instances of primitively defined concepts have level 0, constellations of those concepts have level 1, a concept having level 1 and lower concepts as dattrs has level 2, and so on. If a taxonomy contained only level 0 and level 1 concepts, then the situation recognition problem would be greatly simplified, since one never needs to recognize portions of the input as entities that participate as constituents of larger entities. The general problem, however, requires us to do exactly that. More seriously, the general case requires us to recognize a concept some of whose dattrs may have restrictions defined in terms of the concept itself. This is true, for example, for the concept of noun phrase in a taxonomy of syntactic constructions. Such recursively defined concepts have no maximum level of abstractness, although any given instance will only involve a finite number of levels of recursion. This potential for recursive definition must be kept in mind when formulating algorithms for situation recognition.

5. The Need for Inheritance Structures

As a result of having different levels of abstraction in one's taxonomy, an input situation will often satisfy several situation descriptions simultaneously, no one of which will account for all of the input nor supplant the relevance of the others. For example, adding a ship to a display is simultaneously an example of changing a display and of displaying a ship. Advice for both activities must be considered. Moreover, a single description may have several different instantiations in the current situation, with situation descriptions becoming arbitrarily complex

by the addition of various qualifiers, by the conjunction and disjunction of descriptions, etc. For example, one might want to store advice associated with the situation [wanting to display a large ship at a location on the screen that is within one unit distance from either the top, bottom, or side of the screen when the scale of the display is greater than 1:1000]. Finally, situation descriptions may subsume other descriptions at lower levels of detail, and advice from both may be relevant and may either supplement or contradict each other. For example, displaying an aircraft carrier is a special case of displaying a ship, and there may be specific advice associated with displaying carriers as well as more general advice for displaying any ship. Thus, conventions will be required to determine which advice takes precedence over the other if conflicts arise.

The organization of large numbers of such situation descriptions of varying degrees of generality so that all descriptions more general or more specific than a given one can efficiently be found is one thing we require of an intelligent computer assistant. In order to build and maintain such a structure, it is important to store each rule at the appropriate level of generality, relying on a mechanism whereby more specific situations automatically inherit information from more general ones. That is, when one wants to create a situation description that is more specific than a given one in some dimension, one does not want to have to copy all of the attributes of the general situation, but only those that are changed. Aside from conserving memory storage, avoiding such copying also facilitates updating and maintaining the consistency of the data base by avoiding the creation of duplicate copies of information that then may need to be independently modified and could accidentally be modified inconsistently.

For example, one may want to store advice about displaying geographical features, about displaying such features that cover an area, about displaying bodies of water, about displaying lakes, etc. Thus, information about finding the area covered by a feature would be stored at the level of dealing with such area-covering features. information about displaying water in a certain color would be stored at the level of displaying bodies of water, and information about having inlets and outlets would be stored at the level of lakes. In any specific situation that the system finds itself, many such concepts at different levels of generality will be satisfied, and the advice associated with all of them becomes applicable. That is, any more specific concept, including that of the current situation, inherits a great deal of information that is explicitly stored at higher levels in the taxonomy.

In the case of the situation descriptions that we are dealing with, even the specification of what dattrs a given concept possesses is stored at the most general level and inherited by more specific concepts. Thus, for example, the descriptions of attribute dattrs for color and weight are stored for a general concept of physical object. These dattrs are then inherited by any more specific concepts of physical objects, such as planes, ships, desks, and pencils.

6. The Taxonomic Lattice

I believe that a general solution to the situation recognition problem can be obtained by the use of a classification structure in which traces of individual elements of complex concepts can intersect to facilitate the discovery of coincidences and connections that may not be strongly inferable from constraining expectations. The structure that I propose to use is a version of Brachman's structured inheritance networks, in which descriptions of all potentially relevant situations are stored with explicit indications of general subsumption of one situation by another, and explicit indications of the inheritance of dattrs and of advice by one concept from another. This structure, which I have called a taxonomic lattice, is characterized by a multitude of situation descriptions at different levels of generality.

We say that a situation description S1 subsumes a description S2 if any situation satisfying S2 will also satisfy S1. In this case, S1 is a more general description than S2, and is placed higher in the taxonomy. For example, [displaying a portion of country] is a more specific situation than [displaying a geographical area], which is in turn more specific than [displaying a displayable entity]. All of these are subsumed by a general concept [purposive activity], which in turn is more specific than [activity]. Moreover, a given description can subsume many incomparable descriptions and can itself be subsumed by many incomparable descriptions. For example, an instance of [displaying a geographical area] is also an instance of [accessing a geographical area], [displaying information], and [using the display], and may possibly also be an instance of [responding to a user command].

The space of possible situation descriptions forms a lattice under the relation of subsumption. At the top of the lattice is a single, most general situation we will call T, which is always satisfied and can be thought of as the disjunction of all possible situations. Anything that is universally true can be stored here. Conversely, at the bottom of the lattice is a situation that is never

satisfied, which we call NIL. It can be thought of as the conjunction of all possible (including inconsistent) situations. Assertions of negative existence can be stored here.

At the "middle" level of the lattice are a set of primitive perceptible predicates -- descriptions whose truth in the world are directly measurable by the "sense organs" of the system. All classes above this level are constructed by some form of generalization operation, and all classes below are formed by some form of specialization. At some point sufficiently low in the lattice, one can begin to form inconsistent descriptions by the conjunction of incompatible concepts, the imposition of impossible restrictions, etc. There is nothing to prevent such concepts from being formed; indeed, it is necessary in order for the organism to contemplate, store, and remember their inconsistency.

There are a number of specific relationships that can cause one situation description to subsume another. A given situation description can be made more general by relaxing a condition on a dattr, by eliminating the requirement for a dattr, by relaxing the constraints of its structural description, or by explicitly disjoining it (or'ing it) with another description. A given description can be made more specific by tightening the conditions on a dattr, by adding a dattr, by tightening the constraints of its structural description, or by explicitly conjoining (and'ing) it with another description. These operations applied to any finite set of situation descriptions induce a lattice structure of possible situation descriptions that can be formed by combinations of the elements of the initial set. We refer to this structure as the virtual lattice induced by a given set of situation descriptions. Note that only a finite portion of this lattice need be stored with explicit connections from more specific to more general concepts. By processing this explicit lattice, one can test any given description for membership in the virtual lattice and assimilate any new situation description into the explicit lattice in the appropriate place corresponding to its position in the virtual lattice.

In operation, any situation description about which information is explicitly stored will be entered into the explicit lattice. Any situation that the machine can understand is in some sense already in the virtual lattice and needs only be "looked up" in it. One task we have set for ourselves to develop efficient algorithms to tell whether a given situation can be understood in terms of the concepts of the lattice and if so, to construct its corresponding description and explicitly record its relations to other concepts in the explicit lattice.

7. An Example

As an example of the situation recognition process using marker propagation in a taxonomic lattice, let us consider a simple case of interpreting the intent of a simple English sentence. The example chosen is not complex enough to require all of the machinery discussed, but is presented here to illustrate the mechanism. The major features of the situation recognition mechanism only become critical in interpreting commands that require several sentences to build up, or which depend on the current context in complex ways, but such situations are difficult to illustrate.

For our example, suppose that the system contained a concept for requests to display a geographical region, and the user's input request were "Show me the eastern end of the Mediterranean." The concept [request] contains dattrs for the requestor, the requestee, a description of the state that the requestor desires, a form of request (demand, order, polite request, expression of preference, etc.), and perhaps others, Requests can take many forms. Assume that we have stored in the system a rule that says "Any sentence of the form: 'show me NP' is a request to display that NP." This rule could be stored in the lattice as a piece of advice associated with the concept "A sentence of the form: 'show me NP'," in such a way that when a sentence of the indicated form was found, an instance of a display request would be created. At that point, this resulting display request would be placed in the lattice in such a way that all more general concepts of which it is an instance would be activated, and in particular, the concept of a request to display a geographical region would be activated.

The parsing of the original sentence can either be done by an ATN grammar, or by a version of the taxonomic lattice itself (one that characterizes a taxonomy of sentence types). Let us assume here that it is done by an ATN grammar that is closely coupled to a taxonomic lattice, with the ATN representing the syntactic information about sentence form and the taxonomic lattice representing general semantic information. As the ATN grammar picks up constituents of the sentence, it reaches states where it makes hypotheses about the syntactic roles that those constituents play in the sentence (e.g. "this is the subject", "this is the verb", etc.). Such hypotheses are then entered into the lattice, where they begin to activate the recognition conditions of concepts in the network. For example, in the taxonomic lattice there is a concept of an imperative sentence whose subject is the system, whose verb is "show", whose indirect object is the user and whose direct object is a displayable object.

As the parsing proceeds, the ATN will make assertions about the sentence it is building up, and it will not only be building up syntactic representations of constituents of the sentence, but will also be building up representations of possible meanings of those constituents. In particular, it will be building up a list of those concepts in the lattice of which the current constituent may be a restriction or instance and a list of the dattr-value pairings that have been found so far. If a parse path succeeds (i.e. reaches a POP arc), then a node in the taxonomic lattice corresponding to that hypothesis will be found or constructed. This node will have links to more general and more specific concepts, and will have its constituents linked to appropriate dattrs of those concepts. At the point when this concept node is found/constructed, a process of activation spreading will be launched in the lattice to find any advice that may be inherited by that concept. This process will also leave "footprints" in the lattice that will facilitate the detection of concepts of which the current one may itself be a dattr (or part of a structural condition).

In the example above, when the parser has parsed the initial portion of the sentence "show me", it has built up in its internal registers the information corresponding to the hypothesis that the sentence is an imperative, with subject "you" and indirect object "me". Moreover, it knows that (in input sentences) "you" refers to the system itself, while "me" refers to the speaker. It also knows that the main verb is the verb "show". Let us suppose that at this point, the parser decides to activate the corresponding taxonomic lattice nodes for the concepts [the system], [the user], and [the verb show] (possibly with pointers to the syntactic hypothesis being constructed and/or the labels SUBJECT, OBJECT, VERB, respectively). Ignoring for now whatever information or advice may be found associated with these concepts or their generalizations, the footprints that they leave in the network will intersect at a node [display request] which has dattrs for requestor, requestee, form of request, and requested thing. They also intersect at other concepts such as [imperative sentence], [active sentence], [action], and a more specific kind of display request [region display request], whose requested thing is a geographical region. This latter concept was created and inserted into the lattice precisely to hold advice about how to display geographical regions, and to serve as a monitor for the occurrence of such situations. Fig. 1 is a fragment of a taxonomic lattice showing the concepts of interest. (For details of the notation, see Brachman [1978], Woods and Brachman [1978].)

When the final noun phrase has been parsed and given an interpretation, the footprints that its activation leaves in the network will awaken the [region display request] node, which will then be fully satisfied, and the parser will create a corresponding instance node, with appropriate bindings for its dattrs. In processing the noun phrase, the parser will discover the adjective "eastern" and the noun "Mediterranean" and will activate the corresponding nodes in the taxonomic lattice. The concept [east] is an instance of [direction], which, among other things, is the restriction for a dattr of a concept [directionally determined subregion] that defines the meaning of such concepts as "north eastern Idaho". Another dattr of this same concept has the restriction [geographical region], which is on the superc chain from Mediterranean. Hence, footprints from "eastern" and "Mediterranean" will intersect at the concept [directionally determined subregion], causing an instance of that concept to be constructed as a possible meaning of the noun phrase. The [directionally determined subregion] concept itself has a superc connection to [geographical region], which happens to be the restriction for the "requested thing" dattr of the concept [region display request] which has already received marks for its other dattrs. Thus, the intersection of footprints from the various constituents of the sentence at this concept node has served to select this node out of all the other nodes in the network. Since the more general concept [display request] is on a superc chain from [region display request], it will also be activated, and advice from both places will be considered.

8. Conclusion

In situation recognition, the nodes of a taxonomic lattice structure serve as rendezvous points where footprints from various constituent elements of a concept can meet. This facilitates the detection of coincidences of related events, which in many cases will not be suggestive in isolation. The implementation of the kinds of operations described above involves a system of marker passing conventions for propagating the various "footprints" around the network, detecting coincidences, creating instance nodes, and propagating further markers when coincidences are found. A major portion of our current research involves the discovery of effective conventions for such marker passing operations. Other issues include working out conventions for how far markers should propagate (amounting to decisions as to where to rendezvous), deciding how much information a mark carries with it and to what extent marks are inherited, developing ways to allow a node to remember partial

intersections of marks in such a way that it can incrementally extend them as additional marks accumulate, identifying implications of the marker passing strategies on representational conventions, etc.

9. References

Brachman, R.J. (1978)
"A Structural Paradigm for Representing Knowledge," Technical Report No. 3605, Bolt Beranek and Newman Inc., Cambridge MA.

Fahlman, S.E. (1977)
"A System for Representing and Using Real-World Knowledge," Ph.D. dissertation, Dept. of Electrical Engineering and Computer Science, M.I.T.

Lowerre, B.T. (1976)
"The HARPY Speech Recognition System," Technical Report, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pa.

Quillian, M.R. (1966)
"Semantic Memory," Report No. AFCRL-66-189, Bolt Beranek and Newman Inc., Cambridge, Ma.

Quillian, M.R. (1968)
"Semantic Memory," in Semantic Information Processing (M. Minsky, ed.). Cambridge, Ma:M.I.T. Press., pp. 27-70.

Rieger, C. (1977)
"Spontaneous Computation in Cognitive Models," Cognitive Science 1, No. 3, pp. 315-354.

Wolf, J.J. and W.A. Woods (1977)
"The HWIM Speech Understanding System," Conference Record, IEEE International Conference on Acoustics, Speech, and Signal Processing, Hartford, Conn., May.

Woods, W.A (1970)
"Transition Network Grammars for Natural Language Analysis," CACM, Vol. 13, No. 10, October (reprints available).

Woods, W.A. (1973)
"Progress in Natural Language Understanding: An Application to Lunar Geology," AFIPS Conference Proceeding, Vol. 42, 1973 National Computer Conference and Exposition (reprints available).

Woods, W.A., M. Bates, G. Brown, B. Bruce, C. Cook, J. Klovstad, J. Makhoul, B. Nash-Webber, R. Schwartz, J. Wolf, V. Zue (1976)
Speech Understanding Systems - Final Report, 30 October 1974 to 29 October 1976, BBN Report No. 3438, Vols. I-V, Bolt Beranek and Newman Inc., Cambridge, Ma.

Woods, W.A. (1977)
"Semantics and Quantification in Natural Language Question Answering," to appear in Advances in Computers, Vol. 17, New York: Academic Press. (Also Report No. 3687, Bolt Beranek and Newman Inc., 1977).

Woods, W.A. and R.J. Brachman (1978)
"Research in Natural Language Understanding" - Quarterly Technical Progress Report No. 1 (BBN Report No. 3742), Bolt Beranek and Newman Inc., Cambridge, MA

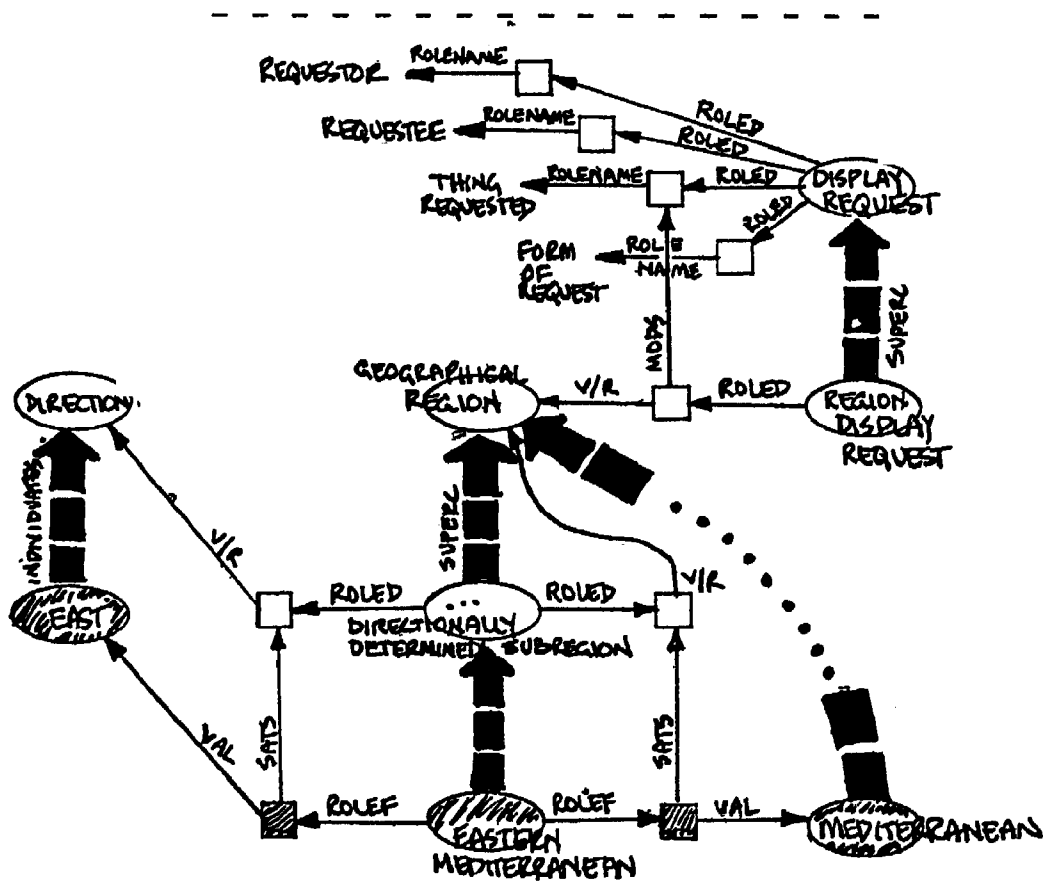

Fig. 1