MASSACHUSETTS INSTITUTE OF TECHNOLOGY

ARTIFICIAL INTELLIGENCE LABORATORY

Memo No. 331 May 1975

THESIS PROGRESS REPORT:

A SYSTEM FOR REPRESENTING AND USING REAL-WORLD KNOWLEDGE

by

Scott E. Fahlman

ABSTRACT

This paper describes progress to date in the development of a system for representing various forms of real-world knowledge. The knowledge is stored in the form of a net of simple parallel processing elements, which allow certain types of deduction and set-intersection to be performed very quickly and easily. It is claimed that this approach offers definite advantages for recognition and many other data-accessing tasks. Suggestions are included for the application of this system as a tool in vision, natural-language processing, speech recognition, and other problem domains.

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the laboratory's artificial intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-75-C-0643.

TABLE OF CONTENTS

1. About This Paper

In my thesis proposal [4], I described a plan for implementing the frame systems of Minsky [12] and applying these to various recognition tasks. Late last Spring (1974) this research took a surprising turn. One of the many schemes I had tried for representing class-inclusion (IS-A) hierarchies--a scheme using Quillian-like nets of parallel processing elements--proved to have so many interesting properties and applications that the study of this system has become the central focus of my research. Among other things, this new representational system eliminates the need for a network of demons for suggesting recognition hypotheses, and it makes the checking of hypotheses much easier.

After a month or two of playing with the new system (to convince myself that it really was what it seemed to be), I decided that it was time to write up my preliminary results. For a variety of reasons, it has taken me almost a year to produce this write-up. The set of ideas that make up the system became very large very quickly, and simply would not stand still long enough to be written about coherently. Always there was one more loose end to tie up, one more example to work out, one more idea to fit in. And the very process of writing about the system, on the few occasions when I actually got started, seemed to trigger major collapses and reorganizations. All this has led to rapid progress in the research itself, but not to a paper.

At long last, I have been able to get a sort of snapshot of the current state of my research down on paper. The old problems remain: there are many loose ends and inconsistencies, especially in the newer ideas presented toward the end of the paper, and there has been a considerable evolution in my thinking even as this was being written. I hope that all of this will not prove to be too confusing. A more coherent presentation will be possible later, after things have had a chance to settle.

2. Overview

On its simplest level the new descriptive system consists of a large set of *nodes* that can be marked in various ways. These represent the objects, classes, and other conceptual entities that the system knows about. Connecting these nodes, and representing the relationships between the node-concepts, are several types of *links*. Ruling over the whole network is the central processor (CPU), a serial computer that broadcasts commands to the nodes and links via a common party-line bus. Markers are passed and propagated from node to node along the link paths under the strict control of the CPU. Since the markers can travel along many branches in parallel, the system can mark very large trees of nodes in only a few steps.

So far, this sounds like Quillian's network system. The difference is that by mating this type of network with a set of fairly rigid organizational principles--a hierarchy of IS-A relations, for example--I have been able to establish much more precise control over the flow of markers through the net. Quillian's system might typically be asked to find all of the associations between LONG and MAN; my system would typically be commanded to mark every man whose grandfather's mistress had a long nose. (This is not to say that Quillian's system could not *in principle* follow complex paths or mine free-associate--the difference is mainly one of emphasis.)

So much for the inspiration. The perspiration has been shed in the attempt to find exactly the right set of organizational principles--the right way to represent various sorts of knowledge--using these nodes and links. This task is not complete as yet, but enough of the system has taken shape to suggest that the finished product will help to solve (or circumvent) a number of long-standing AI problems. A brief summary of some of the system's more interesting features:

-Type or IS-A hierarchies are efficiently implemented. Members of a class inherit all of the characteristic properties of that class and of its super-classes, merely by "plugging in". Access to these properties is fast and does not require pre-computation, redundant storage, or lengthy deduction at access time. This circumvents the problem that so plagued Charniak (among many others) of how many "obvious" inferences should be made when a new fact is received, and how much should be left undone until the need arises.

-The system can handle many orthogonal IS-A hierarches, tangled together to any desired extent.

-It is very easy in this system to find the intersection of two or more classes. These classes can be explicit groupings represented in the IS-A hierarchy, or they can be implicit groupings based upon some common property of the members. Thus, one can feed the system a list of properties observed in a sample, and have it return a list of all known

objects or classes that exhibit such properties. There is more to recognition than this--the list of properties may be incomplete, imprecise, exceptional, or in error--but this mechanism can carry much of the load. In particular, it performs much more elegantly the task I had previously assigned to the swarm of "suggestion demons": the discovery of hypotheses worthy of further consideration.

-When an individual is assigned (perhaps hypothetically) to a class, it inherits the descriptive properties of that class. Often some features of this description will *clash* with features already present in the object, either directly observed or inherited from some previous assignment. The system can detect such clashes very quickly and easily, and can then begin looking for excuses, or whatever. This quick clash-detection is obviously useful in testing recognition hypotheses.

-The same clash-detection mechanism can be used to enforce *restrictions* upon the possible occupants of the various "slots" in a frame, a relation, or a verb. Such a restriction can be very general ("animate object") or very specific ("left-handed Bulgarian tuba player"). Any attempt to make an assignment that violates such a restriction is immediately detected by the system. (One can, of course, force the assignment to occur anyway.) This mechanism would appear to have many uses, especially in linguistics.

-Each piece of information in the system exists in some *context*, and is only visible if the context is *active*. This allows the system to maintain many distinct world-models. Information not in active contexts is completely out of the way and does not slow down the accessing machinery.

-The contexts themselves form a tangled hierarchy with different levels of generality. One can thus be operating in a very general context with a lot of information available or in a very specific context with much less potential for ambiguity and confusion. The system focuses its attention by moving up, down, and around on this hierarchy. This would appear to be a much smoother mechanism than, say, switching between a few distinct mini-worlds.

-While I don't want to press this claim too strongly, this system or something similar would appear to be a plausible metaphor for *human* intelligence. The elements seem rather neuron-like, and because of the parallelism they could be rather slow (milliseconds). The system seems to mesh neatly with the requirements of a linguistic system. And, to me at least, it *feels* right: the things that ought to be easy turn out to be easy. (I'll point out some examples of this as I go along.) I hope in the coming months to see what light the psychologists can shed on this issue, and to gather together any hard data that I can find, pro or con.

Is the special parallel hardware really necessary to reap these rewards? I think so. The essential step in all of these operations is the marking of all the nodes in some tree or other. These trees are seldom more than ten or so links deep, but they are bushy, containing thousands or

even millions of nodes. The parallel scheme marks such trees in time proportional to the depth, while the time to mark serially grows as the total number of nodes. It is my system's ability to make liberal use of tree-marking that gives it any special properties it might have. A recognition does indeed have a sequential component, but it is a sequence of parallel tree-markings. Of course, we can simulate such a system on a serial machine for testing purposes, especially if the total size of the knowledge base is kept reasonably small.

3. The Symbol-Mapping Problem

Suppose I tell you that a certain animal--let's call him Clyde--is an elephant. You accept this simple assertion and file it away with no apparent display of mental effort. And yet, as a result of this simple transaction, you suddenly appear to know a great deal about Clyde. If I say that Clyde climbs trees or plays the piano or lives in a teacup, you will immediately begin to doubt my credibility. Somehow, "elephant" is serving as more than a mere label here; it is, in some sense, a whole package of properties and relationships, and that package can be delivered by means of a single IS-A statement.

In principle, such behavior can be achieved through the use of some form of deduction. Each fact is a separate entity, and new facts are produced by knocking together two old ones. Thus, if we have "All elephants have wrinkles" and "Clyde is an elephant", we have the right to deduce that Clyde has wrinkles. In one form or another, this has been the standard AI approach.

But having the right to deduce some fact is not the same as having the job done. Much ingenuity has been devoted to the search for fast deductive mechanisms, but the problem remains intractably combinatorial. And when is all this work to be done? If we are to detect the obvious clashes between new facts and old, some deduction must be done at once upon receiving new information. But we clearly cannot afford to deduce *all* of the consequences of the new fact--to do so would take a very long time and would hopelessly clog our memory with useless trivia. There seems to be no good way of deciding how far this process should go. And can we really believe that all of this frantic deduction goes on while the listener believes that he is simply accepting a single, straightforward fact? The fastest machines bog down when faced with a few hundred facts. Can millisecond-speed neurons succeed with millions of facts?

I could not help feeling that something was missing here--that the discrete-fact deductive approach was never going to solve the elephant problem. Does this matter? Well, consider where AI has been successful and where it has, so far, failed. The triumphs have been in areas like calculus where the symbols being dealt with carry very little semantic baggage. Where there are rooms to traverse, or tables full of blocks, or missionaries and cannibals, these are abstracted and stylized: all but a very few properties are eliminated. Such a pitiful number of facts can indeed be handled on an individual basis.

But what of the real world, full of shoes and ships and sealing wax, where elephants have not only size, shape, and color, but also wrinkles, blood, flies, and an insatiable lust for peanuts? The list is practically endless, and that is exactly the point. If the phrase "common sense" means anything at all, it must certainly include an awareness of these "fringe" properties. In any given situation, one of these insignificant details may be of pivotal importance. I think that our

critics--Dreyfus, for instance--are right in claiming that AI has made little real progress on this front.

If fact-by-fact deduction is inadequate, is there perhaps some way of dealing with whole groups of facts at once? This possibility first occurred to me while I was playing with the multiple data-base contexts of CONNIVER, which allow whole sets of facts and demons to be made visible or invisible by a single declaration. I also had in mind an image derived from the "pure" and "impure" code in a computer system: The elephant description should not be copied and modified for each elephant; rather, it should be kept "pure" and individual elephants should be described by "plugging in" to this description and adding a small package of local assignments and modifications to the general model. This image was strongly reinforced and influenced by Minsky's frame theory. Eventually, all of this led to the packet system described in my thesis proposal.

But packets were not the right answer either. It is all well and good to make the elephant description appear and disappear, but the real problem is to turn all of the ELEPHANT properties into CLYDE properties. I call this the *symbol-mapping problem*. For a while, I thought that I could leave the owner of a property unspecified, but this led to terrible problems of ambiguity, especially where the property was really a relation between two more-or-less equal entities, as in FATHER-OF. I had been playing with various parallel hardware schemes for implementing packets efficiently, and I decided to try something similar for symbol mapping. This was meant to be an exploratory step, but the result was so successful in so many ways that I stayed with it. (Incidentally, I later realized that packet activation was just a special case of symbol mapping and that it required the same sort of mechanisms for success.)

4. Nodes and Links

We will now consider in some detail the nature of these nodes and links, and how they can be used to solve the symbol-mapping problem. This did not materialize all at once: I am skipping over about a half-dozen intermediate steps in the system's evolution.

The various concepts in the system--for now we can limit these to individual objects like CLYDE and groupings like ELEPHANT--are represented by *nodes*. These are relatively simple hardware units. Each one contains a permanently-assigned unique *serial number* and a half-dozen or so independent flip-flops called *marker bits*. I mentioned earlier that the nodes could be marked in various distinct ways; this is done by setting one or more of the marker bits in a given node unit. These bits are designated by letters; thus, we speak of a node as being "marked with an A-marker" if its A bit is on.

Each of the nodes is connected via a common party-line bus to the CPU, from which it gets its orders. The CPU can specify any node by serial number and can order that node to alter or report its marker status. More often, a command will be broadcast to any and all nodes containing some particular marker or combination of markers. Sometimes the CPU will just want to know if any such nodes exist; sometimes it will want these nodes to change their marker state in some way; sometimes it will want the selected nodes to report their serial numbers over the bus. If many nodes try to report at once, they are queued up in order of their serial numbers (or perhaps in order of their position along the bus) and they report in one-by-one.

These nodes are analogous to LISP atoms in several respects. The actual *word* "elephant" is not a part of the ELEPHANT node, but is attached to it as a property. (The details of this will be discussed later.) Instead, the node we will call "ELEPHANT" represents the *concept* of elephant and is known to the system by its semantically sterile serial number. A few nodes may have some special meaning to the CPU (like NIL in LISP) or some direct association with the raw output symbols of the I/O system, but most derive their meaning from the way they are connected with other nodes.

These connections are represented by hardware *link units* running between the nodes in question. For now, let us think of these links as coming in many different flavors, corresponding to the different possible relations in the system. Thus, to represent "Clyde is an elephant" we simply connect an IS-A link from the CLYDE node to the ELEPHANT node. Similarly, we might have an OWNS link from ROCKEFELLER to STANDARD-OIL, a FATHER-OF link from ABRAHAM to ISAAC, or a COLOR link from ELEPHANT to GRAY. These links have a direction, in the sense that a relation like OWNS or FATHER-OF is not commutative, but markers can be sent across them in either direction. (We will later see a way of getting along with only a single, more complicated kind of link.)

These link units are considerably more than mere labeled wires. Each is a hardware unit that is connected to the party line bus, and each is able to carry out simple commands received from the CPU. These hardware units are connected to the nodes that they are supposed to be linking by dedicated private lines--actual wires that are not shared with any other link unit. Over these private lines the link units can sense and alter the marker status of the attached nodes. When I speak of a "link" I will usually be referring to the hardware link unit and to the attached private lines as a single entity.

The commands sent from the CPU to the link units have two parts: first, a specification of which links are to respond; then, a statement of what these selected links are to do. Ususally, a whole set of links will be selected at once, on the basis of their type and the marker status of the attached nodes. A command might, for instance, be addressed to all IS-A links whose incoming node (the X in "X IS-A Y") is marked with bit A and whose outgoing node (Y in the above example) is not so marked. The CPU can sense whether any links claim membership in this group. The command follows: in this case, it might specify that each of the selected links should mark bit A of its outgoing node. Because of the private line connections, all of the selected links can perform this operation simultaneously. The net effect is to propagate each of the A markers in the system across any adjacent *outgoing* IS-A links, all in a single machine cycle.

Now, if we mark CLYDE with bit A, and we then repeat the above operation until nothing more happens, we will have placed an A marker on every node that represents a class of which Clyde is a member, either directly or by transitivity of IS-A. If, instead, we mark MAMMAL initially and propagate the markers across IS-A links in the *opposite* direction, we will have marked every node that represents a kind of MAMMAL and every individual MAMMAL. Regardless of the number of nodes marked in such an operation, the time required is proportional only to the number of links forming the longest branch of the tree. It is very hard to think of a reasonable chain of IS-A relations longer than ten or fifteen links--try it! What all this means is that, given this hardware, we can use tree-marking as an operation with no more trepidation than a LISP user would feel in using a COND or a SETQ. Even if the tree is, say, the tree of all physical objects. Even if the elements of the system operate as slowly as, say, neurons.

And this, I claim, breaks the back of the symbol-mapping problem. When we tell the system that Clyde is an elephant, it simply creates an IS-A link from CLYDE to ELEPHANT. (In the process, it checks for clashes--more about this later.) ELEPHANT, of course, is linked to many more general categories: MAMMAL, ANIMAL, QUADRUPED, HERBIVORE, PHYSICAL-OBJECT, and so on. Each of these classes has its own set of characteristic properties, represented by property links attached to their nodes, and all of these properties are to be inherited by CLYDE. Now, if we ask the system for, say, Clyde's color, it won't know which node to look at, but it

doesn't matter! In a single sweep, it marks all the nodes superior to CLYDE; then, it asks any COLOR-OF links attached to *any* of the marked nodes to mark (with a different bit) the associated color node; finally, it asks the marked color to report itself to the CPU. Thus, it becomes unimportant whether a property is attached directly to CLYDE, or is really attached somewhere far up a chain of IS-A links. There is much to be added to this picture--exceptions, for instance--but this is the key idea.

The opposite marking process--from a class to its sub-classes and members--has its uses as well. Suppose we have an OWNS link from KING-ARTHUR to EXCALIBUR, his famous magic sword. Suppose that we want to know if any kings own any weapons. We simply mark with bit A every KING, mark with bit B every WEAPON, and look for an OWNS link between these two sets. Out would pop the ARTHUR/EXCALIBUR link, along with any others that the system might happen to know. To mark all weapon-owning kings we would mark all weapons (bit A), then all their owners (bit B), and finally all kings (bit C). Any node marked with both B and C is a winner.

Before we go on to the more complex issues that arise in trying to use these nodes and links properly, perhaps it would be worthwhile to briefly consider whether we can, in fact, build a system like this using existing technology. The problem, I believe, is not with the node and link units themselves--these are indeed expensive, but not ridiculously so given LSI technology--but rather with the private lines connecting them. As the system learns new things, it will need to create new connections and occasionally destroy old ones (though some form of cancellation might substitute for link-destruction). This link-alteration can afford to be rather slow, since it corresponds to a long-term memory alteration; there are tricks we can use to represent a *limited* number of quick, temporary alterations until they are no longer needed or can be wired-in permanently.

There would appear to be two possibilities: either we must have some device or process that can connect (or grow) a wire directly between a node and a link terminal, or we need some kind of crossbar switch. Imagine a set of wires--one from each node--running north/south in a plane. Above this is another plane, made of link-wires running east/west. Wherever two wires cross, a connection can be made or broken; this is called the *cross point*. It would obviously be too expensive to put an actual switch at each such point, but perhaps there is a cheaper way of making and breaking contact at such points--some sort of electrochemical goo that turns locally conductive when appropriately zapped and stays that way until it is somehow unzapped. It would help somewhat to group nodes and links into richly connected neighborhoods, with relatively few connections running to other neighborhoods, but this might be more trouble than it is worth.

What I'm trying to say, then, is that it would be far too expensive to implement this system on a large scale with currently available hardware, but that the development of a much cheaper implementation technology is not inconceivable. In the meantime, the system is well enough defined to serve us as a precise metaphor, and it can be simulated

for testing pruposes. (Of course, in the simulation, tree-marking will again be proportional to the number of nodes in the tree, but we will be able to tell how long the hardware would have taken.)

5. The IS-A Hierarchy

The backbone of the descriptive system--the rigid structure which keeps everything else from collapsing into a heap--is the network of IS-A links. These links tie all of the nodes in the system into a single tangled hierarchy of groups, subgroups, and individual members. With perhaps a few exceptions (to be noted later) every node is connected by at least one IS-A link to some more general category that includes it: every concept has an identity. Eventually, all of these upward-moving IS-A chains converge upon the system's most general node, named THING, which includes everything (every-THING) else.

Before proceeding, let me make explicit a convention that I have already used once or twice, and will use a lot more in the remainder of this paper: IS-A links will be spoken of as pointing upward. The MAMMAL node is *above* ELEPHANT and *below* ANIMAL. The more general categories are the *superiors* or *ancestors* of the less general ones. THING is at the *top* of the tree; CLYDE and other individuals are at the *bottom*. Links representing other properties run more-or-less sideways. This will save a lot of words in the long run. Such an image is possible, of course, because the IS-A links do not form loops. (Exception: we may say A IS-A B and B IS-A A if we want to indicate that both nodes represent the *same* concept. This is useful in certain perverse cases.)

Probably the most important aspect of the IS-A tree is the way properties are inherited. Consider a node like COW. (One grows weary of elephants after a while.) Properties and relations attached to the COW node itself, *or to any node directly above it in the IS-A tree*, are meant to apply to *all* cows (though particular individuals can cancel particular inherited properties). Properties hung from nodes *below* COW apply only to *some* cows. In Figure 1, for instance, we see that *all* cows are warm-blooded and have udders (barring exceptions, that is), but that only *some* cows are black-and-white, while others are brown. There are not, to the system's knowledge, any purple cows, but this is not ruled out. If asked what color cows are, the system would first mark upward from COW, but would find no COLOR links there. It would then mark downward from COW, find several COLOR links, and report that cows in general have no particular color, but that some cows are brown and some black-and-white.

It is important in choosing marking and accessing strategies, always to respect the transitivity of the IS-A links. If COW IS-A MAMMAL and MAMMAL IS-A ANIMAL, then COW IS-A ANIMAL, whether or not there is a link *explicitly* saying so. This effect is achieved if we are careful always to let upward or downward marker sweeps run to completion; we should be very suspicious of any accessing strategy that sends markers some particular number of steps up or down the tree.

A corollary of this transitivity is that we are always free to split an IS-A link, for instance to add the node UNGULATE between COW and MAMMAL. Once we have added COW IS-A UNGULATE and UNGULATE IS-A MAMMAL, the

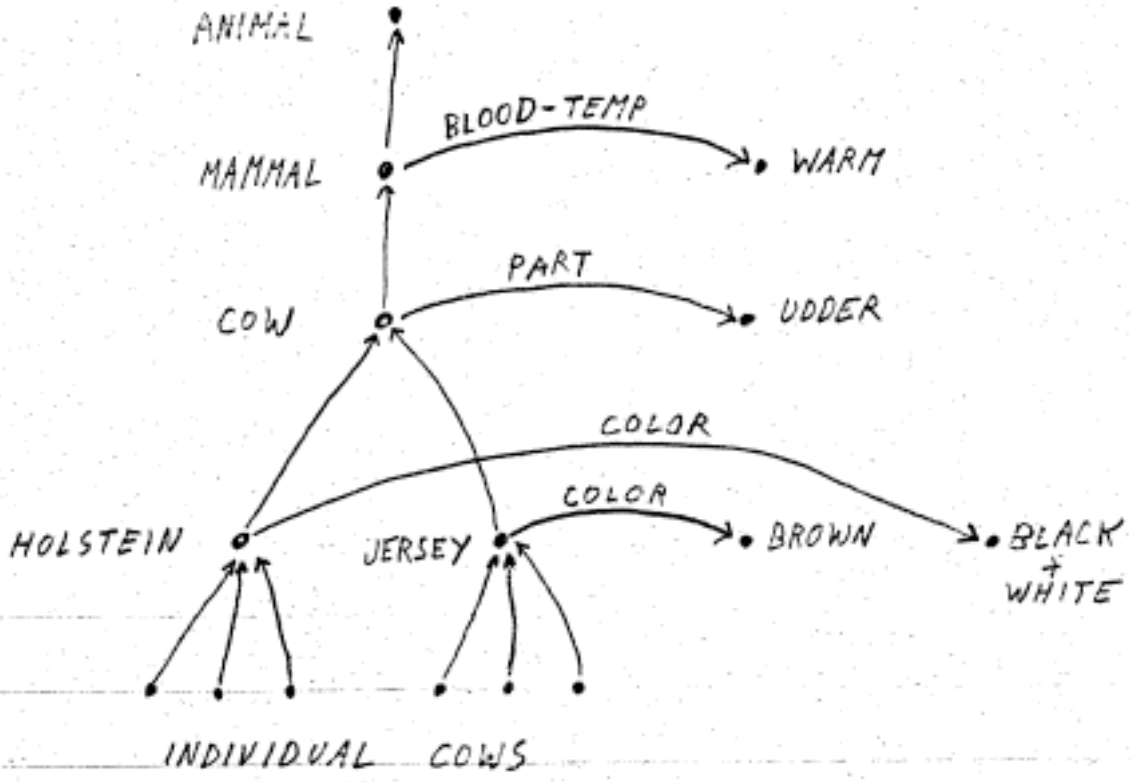


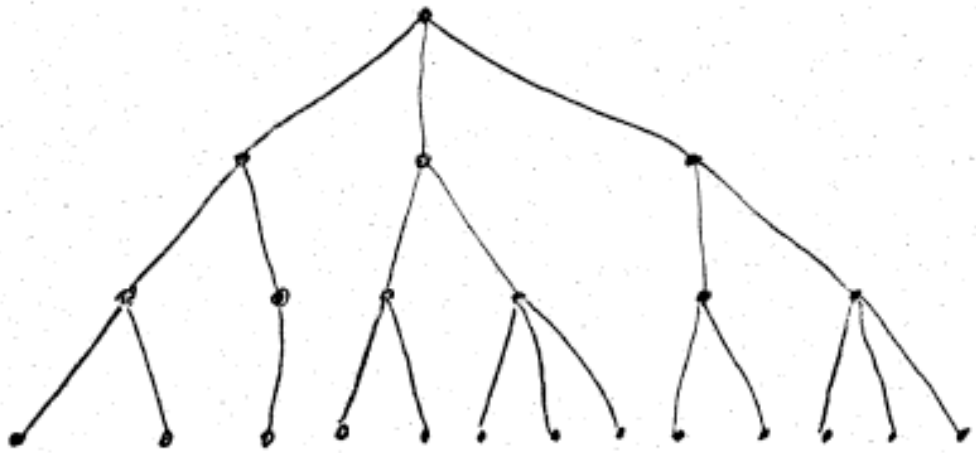
Figure 1 - a portion of the COW tree.

original IS-A link from COW to MAMMAL becomes redundant. The system can reclaim this link or leave it in place, whichever is easier. As you can see, it makes sense to talk about one node being above or below another, or between two others, or being off on some unrelated branch of the IS-A tree, but it makes very little sense to talk about *distances* up and down the tree, since these can be measured along various paths and changed at will.

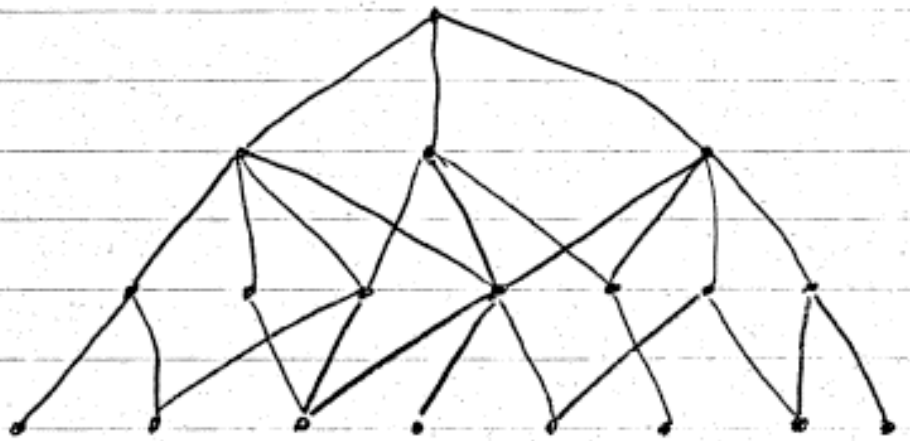
The IS-A network is a *tangled* hierarchy. By that I mean simply that a given node may have more than one immediate superior, but this simple fact can have some interesting global effects. In the more familiar non-tangled hierarchy, a node may have many IS-A links fanning out below it, but only one link to a superior. This convention leads to downward-branching trees like the one in Figure 2a, in which no two branches ever recombine once they have separated. Figure 2b shows the tangled hierarchy, in which the branches do tangle back together at various lower nodes.

The difference is more dramatic if we look at the tree from the point of view of some typical node--call it X--in the middle of the tree (Figure 3). In either case, the descendants of X fan out into a profusion of subclasses and members, but looking upward from X it is a different story. In the untangled hierarchy, X has only a single strand of superiors, while with tangles the tree of X's superiors can fan out to considerable dimensions before it starts converging back towards the THING node. X is thus in a position to inherit a very large and diverse set of properties. Without some sort of parallel accessing system this could be an embarrassment of riches, but with the nodes and links described earlier, a large tree can be marked and accessed as easily as a single strand of nodes.

It might seem that there is some danger of runaway markers finding their way through the tangles and marking everything. We can avoid this by taking care to mark *only* upward or *only* downward from a node, but never both in a single sweep. (Remember that there are no significant loops of IS-A relations in the system.) If, for some reason, we should want to mark *both* the ancestors and descendants of a node, we should do this with two separate sweeps, and then convert the markings of one set to match the other.

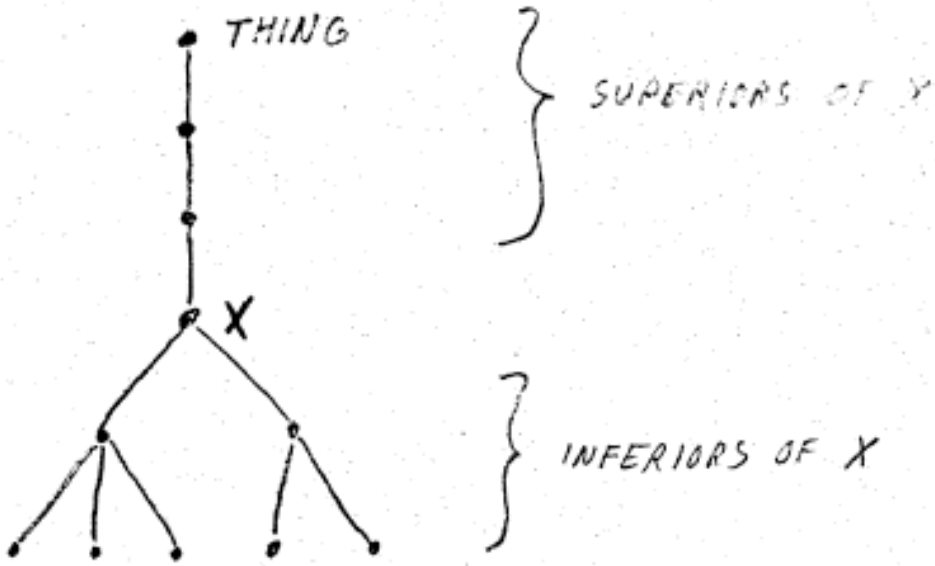


(a) UNTANGLED HIERARCHY

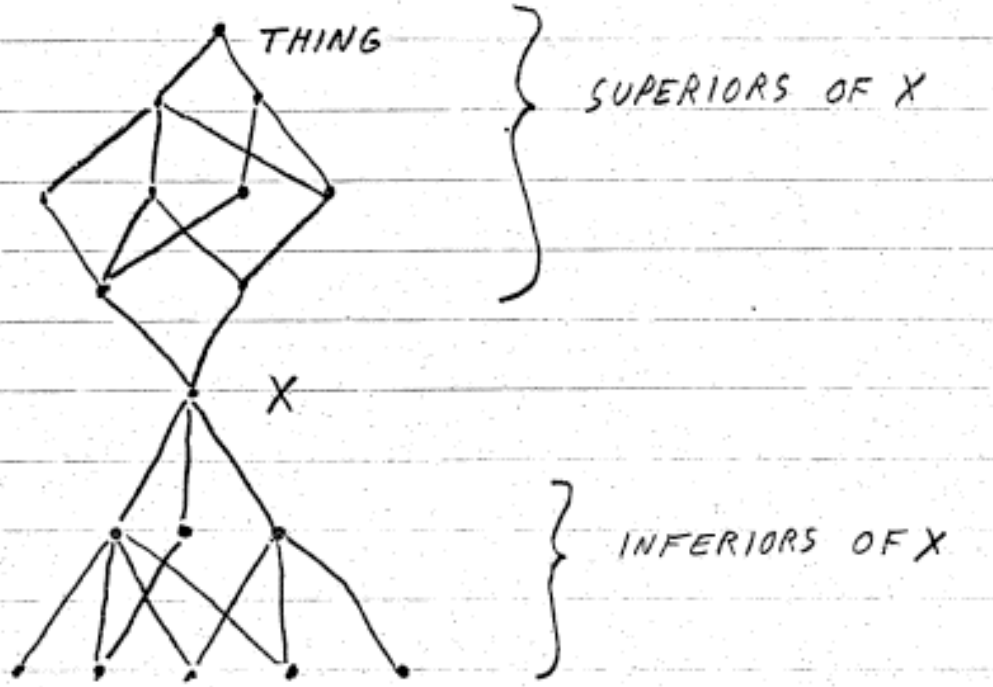


(b) TANGLED HIERARCHY

Figure 2 - Tangled and untangled hierarchies, viewed globally.



(a) UNTANGLED HIERARCHY



(b) TANGLED HIERARCHY

Figure 3 - Tangled and Untangled hierarchies, as seen from lexical node X.

6. Exclusive Sets and Clashes

The need for tangles in the IS-A hierarchy arises from the fact that most group nodes can be divided into sub-groups in many different ways, according to more-or-less orthogonal sets of features. The PERSON node, for example, can be divided on the basis of age, sex, occupation, race, height, and so on. While we would not normally have occasion to tangle together the results of any single split--MALE and FEMALE, for example--we may very well want to recombine the results of *different* splits, as when we recombine MALE and CHILD to get BOY. Figure 4 shows a part of the tree that hangs below PERSON. The existence of a recombined node like BOY not only gives us a way of including an individual in both of the superior categories with a single IS-A link; it also gives us a place to hang any properties that are characteristic of the *combination* of MALE and CHILD.

The arcs in figure 4 indicate various *exclusive sets* among the subgroups of PERSON. An individual or group may have an IS-A link to *one* member of *each* of these groups, but only one. Any attempt to link--directly or indirectly--to a second member of an exclusive set should be detected by the system as one form of *clash*.

The actual clash may occur far up the tree from the offending node. To take a rather far-fetched example, let us try to imagine a marsupial mushroom. These two classes do not directly clash with each other, but if we trace the IS-A chains upward, we find ourselves marking both PLANT and ANIMAL, which are part of an exclusive set under LIVING-THINGS. We might be able to make sense of this by treating one class or the other as a metaphor or analogy--perhaps we have a fungus that protects its spores in a pouch--but we are not free to accept this description literally, as we would if the object were, say, a marsupial herbivore (no clash).

We would, of course, like to have some quick way of detecting these clashes. Even if this only catches gross absurdities, that is at least a step toward some sort of common sense. The trick is to create two distinct flavors of nodes, *exclusive* and *non*-exclusive (or regular) nodes. These types can either be inherently different or permanently marked with an exclusion bit or labeled with an appropriate link, whichever is easiest in a given implementation. A node like PERSON, which encompasses several exclusive sets, becomes a regular node with several exclusive nodes hung below it (figure 5).

A clash is detected whenever marking upward from a single node causes markers to pass through an exclusive node from two different incoming IS-A links. Suppose we want to add a new IS-A link above a node X that already has some connection to one or more higher nodes. First, we mark with bit A all of the superiors of X according to the *old* links. Then, we send bit B propagating up the *new* link in such a way that when a B-marker enters an A-marked node, it deposits a B but goes no farther. We then ask any exclusive nodes with both A and B marks to report in. If

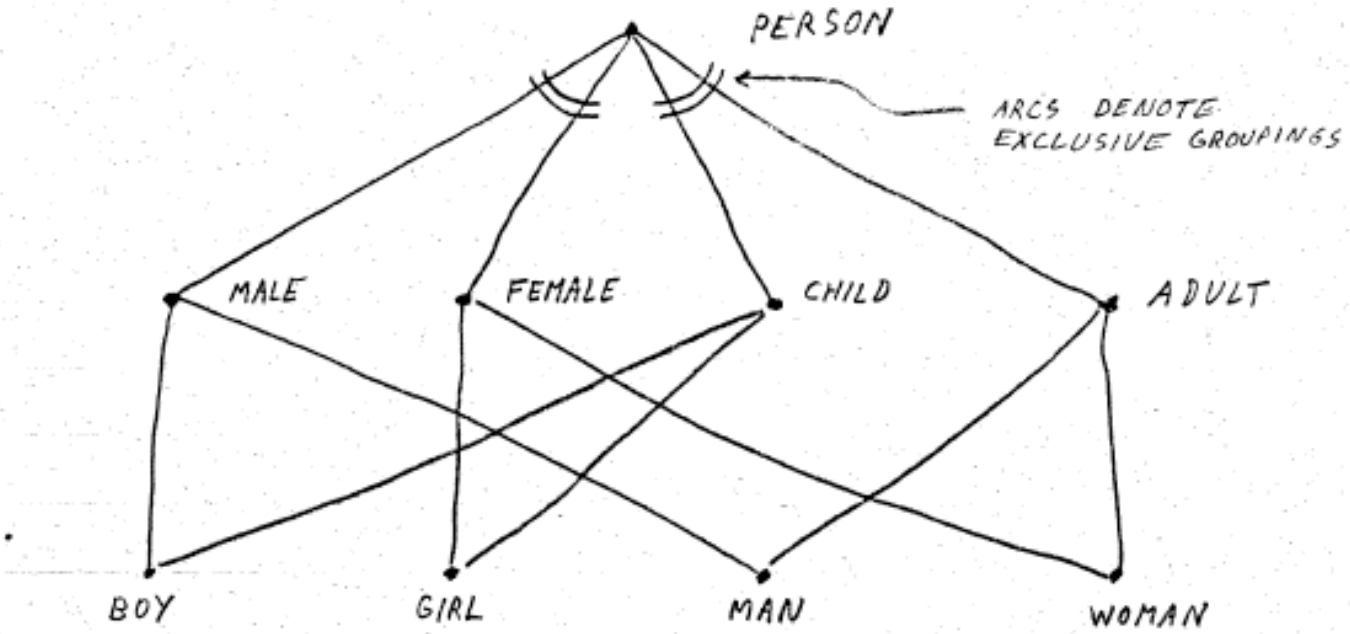


Figure 4 - a portion of the PERSON sub-tree.

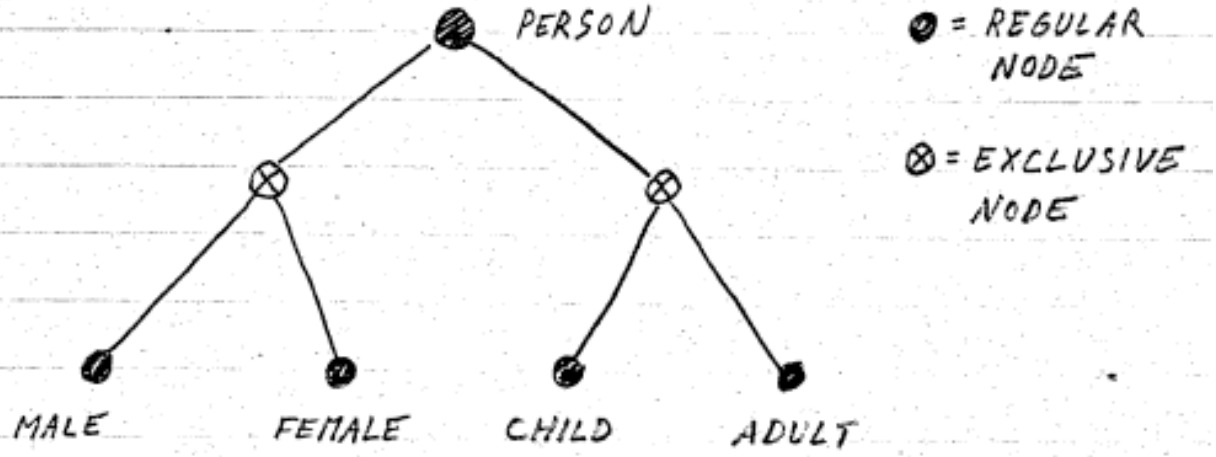


Figure 5 - Fine structure of PERSON node-cluster.

there are any, these are the clashes.

Any specific suggestions that the system might have for dealing with a particular clash can be hung from the exclusive node representing the clash, but usually we will just resort to general methods for reconciliation. Depending on the evidence we have for asserting each of the clashing links, we might simply reject one or treat it as a metaphor. Sometimes we can turn off the alarm and ignore the clash, but this usually just leads to more clashes as we try to reconcile the properties of the incompatible classes. (Blood or sap? Cellulose cell walls or not?) Sometimes, as with the euglena or the sex of an earthworm, we will have to create a new set to represent the individual in question and add this to the set of possibilities under the exclusive node; this most often occurs when there is a twilight zone between the categories or some sort of hybrid of them, but it can represent a completely new possibility.

The mechanism so far described catches only clashes between *explicit* categories to which an object is assigned; clashes between the *properties* of an object can be more difficult. If we say that a person has a long nose, and later assign him to a class that implies the possession of a short nose, we might not notice the discrepancy which would be obvious if long-nosed people and short-nosed people were considered as distinct classes. We can't flood markers out to all of the parts and properties of a description and look for clashes, since the *long* of "long nose" would clash improperly with the *short* of "short finger", or whatever.

I have not yet really attacked this problem, since it depends on certain details of the basic link format that have not yet been worked out. For now, just let me say that clashes between an object's superior classes are trivial to find; clashes between *global* properties of an object (herbivorous-carnivorous, large-small) appear to be detectable by a single sweep as well; clashes between the properties *of parts* would appear to take a separate sweep for each part. I don't think this is too inconsistent with human capabilities. One final note: any particularly important part-properties can become whole-properties, as when a person with large muscles becomes a muscular person or a person with yellow hair becomes a blonde. We will see how such transformations occur in a later section.

The exclusive nodes give us a way of indicating what an object is *not*. If we are told that Clyde is not a mammal, we simply create an exclusive set containing CLYDE and MAMMAL. (See Figure 6.) Any subsequent attempt to connect CLYDE to MAMMAL, or to some sub-MAMMAL such as ELEPHANT, sounds the alarm.

Andee Rubin [15] has suggested, in a similar context, the use of *complete exclusive sets*, and Rick Grossman [5] has built an entire descriptive system using such sets. These are just like regular exclusive sets, except that it is mandatory for an individual to fit into exactly one of the sub-classes in the set. This convention comes into play when we have ruled out all but one of the possibilities in the set: if I tell you

that CLYDE is a vertebrate, but that he is not a fish, reptile, bird, or amphibian, you can be *reasonably* sure that he is a mammal.

The problem, as you can see, is that there very few classes of anything (outside of mathematics) that can be divided so cleanly that we are sure that we have listed *all* the possibilities. Even in so crystalline a dichotomy as sex we find some isolated cases of hermaphroditism, XXY chromosome sets, and so on. But "reasonably sure" is usually good enough, and the process of elimination does seem to play a part in our reasoning, especially where the allegedly-complete exclusive set has only two or three members. Sherlock Holmes, for one, used this sort of reasoning quite often, and with excellent results.

It is a simple enough matter to create a third type of node to represent these complete sets and to treat them as exclusive nodes for clashing purposes. What I have not yet found--and do not really expect to find--is a fast parallel way of detecting when an object has closed off all but one of the possibilities of some complete set somewhere, so that we can rush in and plug the object into the remaining possibility. This lack does not worry me too much: it seems to me that when humans reason by elimination, they proceed laboriously, case-by-case, sometimes making a list or counting on their fingers. In view of this, it would be surprising if the machine were *too* good at such reasoning. We will see a bit more about this later, when we consider how the machine digests new information.

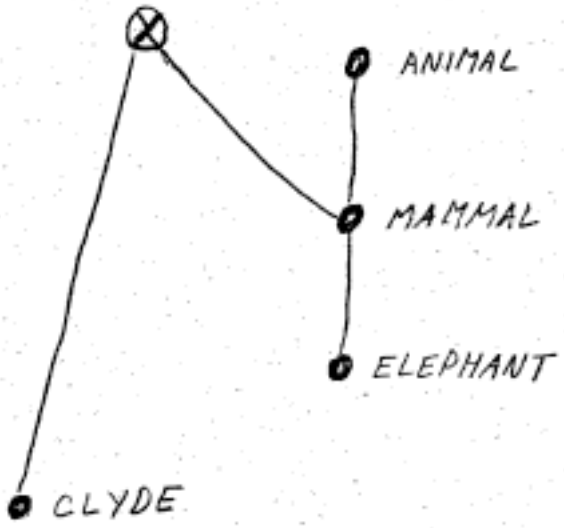


Figure 6 – "Clyde is not a mammal."

7. Equivalent Forms and Winston-Learning

Before I go on to introduce more machinery, I would like to clear up a few remaining points about the IS-A hierarchy, and especially about how it grows and changes as new nodes and properties are added. For now, I will continue the practice, introduced earlier, of representing each property or relation--COLOR, OWNS, FATHER-OF--by its own type of link. In a later section we will see what these links *really* look like.

The first point to consider is that the transitivity of the IS-A relations makes it possible to represent certain relationships in a variety of ways. In Figure 7, we see that when a number of nodes claim membership in some set of superior groups, we can create a new class representing the intersection of these superior groups and simply link the individual nodes to this. (The picture seems to say it better than the words.)

Such transformations are *not* mandatory. All of the nodes in the left-hand diagram are hooked up to the proper superiors, and the system can get along this way indifinitely. But, especially when many individuals belong to a set with many superiors in common, making the transfoamation can save a large number of links at the cost of only a single new node. Thus, while there is no particular urgency about it, the system can benefit from spending some of its spare time looking for situations that can be profitably transformed in this way. The exact threshold of profitability in such cases depends upon the details of the implementation: the relative cost of nodes and links, whether old links can be recycled, the cost of new connections, and so on.

Figure 8 shows a similar equivalency based on properties rather than group membership. Where a number of individuals have some common property or group of properties, it is possible to make such properties characteristic of a group. Individuals, instead of having to claim each of these properties directly, simply join the group by using an IS-A link. I will refer to this process--creating a group-node and moving an appropriate set of properties from lower nodes up to the group--by the term "generalization" and to the new node as a "generalization node".

These self-reorganization processes, I believe, are responsible for the creation of the vast majority of the network's mid-level nodes. A node like ELEPHANT is created as soon as someone has seen two or three individual elephants, to serve as a place to attach the very large number of properties that these individuals share. The simplest way to establish such a node is to commandeer the node that represents the first or most typical member of the group to be the new group node, leaving its properties intact, and creating a new subordinate node for the individual. Properties which, as more elephants are added, are found to be idiosyncratic rather than general are passed back down the tree to the new individual node.

These generalization classes are more than a mere device for saving

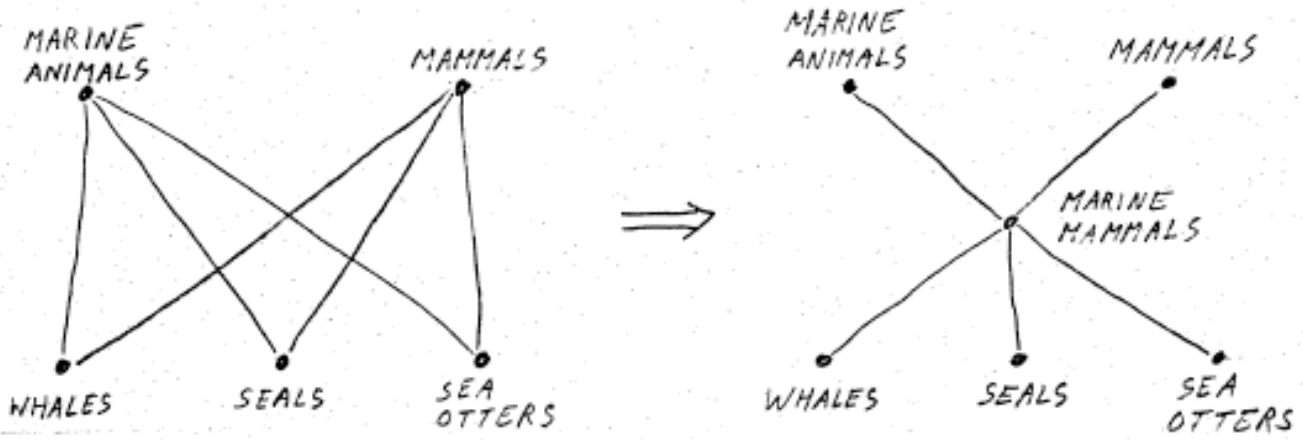


Figure 7 - Reorganization of group memberships.

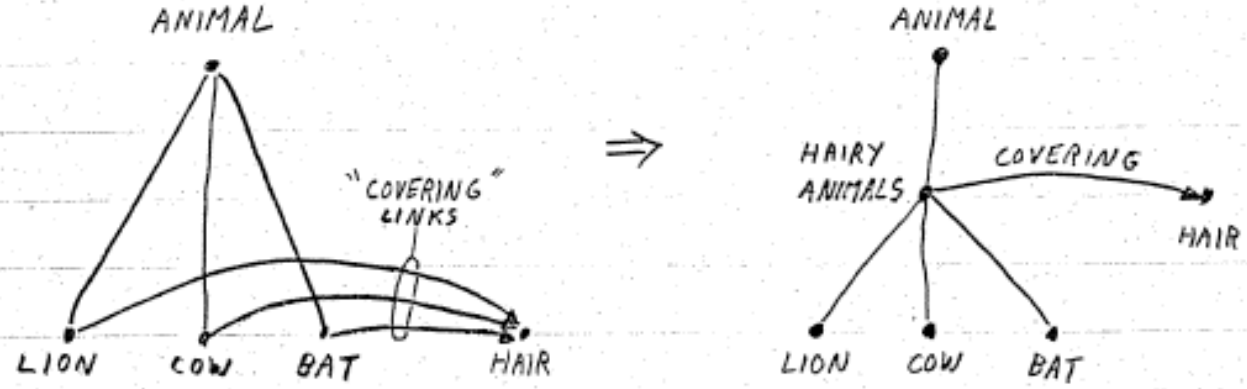


Figure 8 - Generalization of common properties.

a few links: combined with a suitable recognition strategy, they play an enormous role in organizing our perceptions. Consider how well the human species would have fared if the ferocity of each individual animal had to be evaluated independently. By the time some particular saber-toothed tiger has taken a bite of you, it is probably too late. It is a far better strategy to assign a new individual to some group on the basis of the readily-observable information and to assume that the rest of the group's properties hold as well, even though these properties have not been observed directly. If any of these assumed properties have been explicitly contradicted by observation, of course, the system must either change the identification or make explicit note of the exceptional properties.

So we see that by properly creating and using these generalization nodes, we can make a few observations do the work of many, in much the same way that we got CLYDE IS-A ELEPHANT to do the work of many statements. The success of this approach depends upon the domain, of course. In domains where the properties are rather obvious and form coherent and widely separated groupings--animals for example--it will be very successful. In domains where the properties are themselves hard to recognize and where the groups blur together and contain many exceptions, things will be much more difficult. So while everyone agrees that ELEPHANT is a valid and useful class, there is much less agreement about SCHIZOPHRENIC or LIBERAL (the philosophy, not the party). Still, I am convinced that most of the categories needed for common sense reasoning are of the coherent, ELEPHANT type. Overzealous use of generalization and an unwillingness to recognize exceptions are dangerous, as well: these are the sources of bigotry and racism, along with many other forms of too-rigid behavior. The system must try to find the best balance.

So the concept ELEPHANT is not some innate ideal category (though PHYSICAL-OBJECT may be); nor is it some precise mathematical predicate ("able to mate successfully with the Prototype Standard Elephant, kept in a vault in France"), though progress in genetics may reduce it to that; ELEPHANT is merely a shorthand for a bunch of properties that seem to occur together often enough for the shorthand to be useful.

This way of looking at concept-nodes is an extension of (and is directly descended from) Winston's description-learning system [23]. I have emphasized the process by which generalization nodes are created in the first place, as the system itself searches out parts of the net that can be profitably reorganized. Winston studied the process by which a useful generalization can be passed from a teacher to a learner, once it exists somewhere in the culture. Instead of having to search blindly for places to put generalization nodes, the learner is told, for instance, that several different structures are all to be considered members of the class ARCH. By showing the learner a carefully chosen sequence of members and non-members of the class, the teacher is able to control with great precision which individual properties will be moved up to the new general node.

It has been observed many times that the language itself carries a substantial load of Winston-instruction in its collection of words. Not every node has a name, but every name has a node. By observing what objects the name is applied to, the learner can gradually ascertain the boundaries of the intended class, and it is then usually a simple matter to deduce which properties should be moved up. Random conversation can thus serve as a training sequence, though not a very efficient one. More difficult concepts are better conveyed by more efficient means: explicit definitions, as when we state that a mammal is a warm-blooded vertebrate with hair and so on. This can be converted more or less directly to a piece of network.

In any event, there seems to be reason to hope that if we can find the right set of self-reorganization procedures, along the lines discussed above, we should be able to get the system to do most of the work in creating its own knowledge net. Much of the rest can perhaps be taught instead of programmed, though there will be some minimum of "innate" structure upon which the rest is built. This is an old dream, of course, but it seems more plausible in the context of this type of system. I have not even begun to work out the details of the reorganization strategy.

8. Digestion and the "Every" Problem

One more loose end to tie before we go on to bigger things: As the logicians in the audience have no doubt already noticed, the system described so far has no good way of representing such statements as "All purple mushrooms are poisonous." We can state that *all* mushrooms are poisonous by simply hanging a POISONOUS property on or above the MUSHROOM node. We can state that purple poisonous mushrooms *exist* by creating a node below MUSHROOM and hanging PURPLE and POISONOUS from this. (It may also be necessary to somehow declare this set non-empty.) But the fact that this class exists does not mean that we are required or even allowed to move every purple mushroom down into it, any more than we are allowed to move every gray animal down into ELEPHANT. I call this situation the "every" problem. The solution is not too difficult. We label certain nodes as being *mandatory* and divide the properties of these nodes into two classes: *essential* and *incidental* properties. (We will see later a variety of ways of attaching these labels.) The PURPLE-MUSHROOM class is a mandatory subclass of MUSHROOM; it has PURPLE as an essential property and POISONOUS as an incidental. (See Figure 9.)

Now, whenever we add a new member to MUSHROOM or add a new property to some mushroom, we check the various mandatory subclasses of MUSHROOM (and of its superiors) to see whether the individual fits into any of them. To fit, it must explicitly satisfy all of the essential properties of the mandatory node: in our example, the individual must be PURPLE. Note that merely failing to clash with PURPLE is not good enough, but being some *kind* of purple is. Once we have found a mandatory subclass which fits an item, we *must* move that item down into that subclass. The individual then inherits the incidental properties of its new class--in this case POISONOUS.

Since the properties of a class can be inherited as well as being attached directly, we must also allow the mandatory nodes to claim essential and incidental class membership. Figure 10 illustrates this. If any individual claims membership in *all* of the essential superior classes of a mandatory node--in this case, if it is both a MUSHROOM and a PURPLE-OBJECT--then we move it down and let it enjoy membership in the incidental (POISONOUS-PLANT) classes as well. The two systems can, of course, be combined: to fit into a mandatory class an individual must show all of the essential properties *and* belong to all of the essential super-classes.

This scanning process, initiated by the addition of new information (links) to the system, is an example of what I call *digestion*. The clash-checking process and the search for complete exclusive sets with only one remaining possibility are also digestive processes. These digestive processes fill the same ecological niche as the IF-ADDED demons of other systems, but they are more limited in power: there are only a few types of digestive processes, and only a single way of representing each. To me, this seems like an advantageous limitation. Once in a great while the system may need to change or augment its central program for digestion

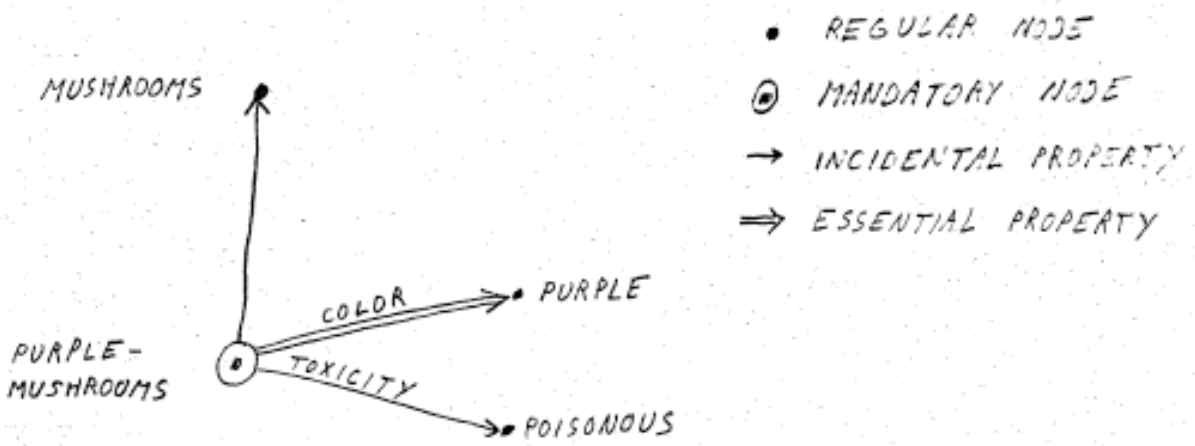


Figure 9 - "all purple mushrooms are poisonous."

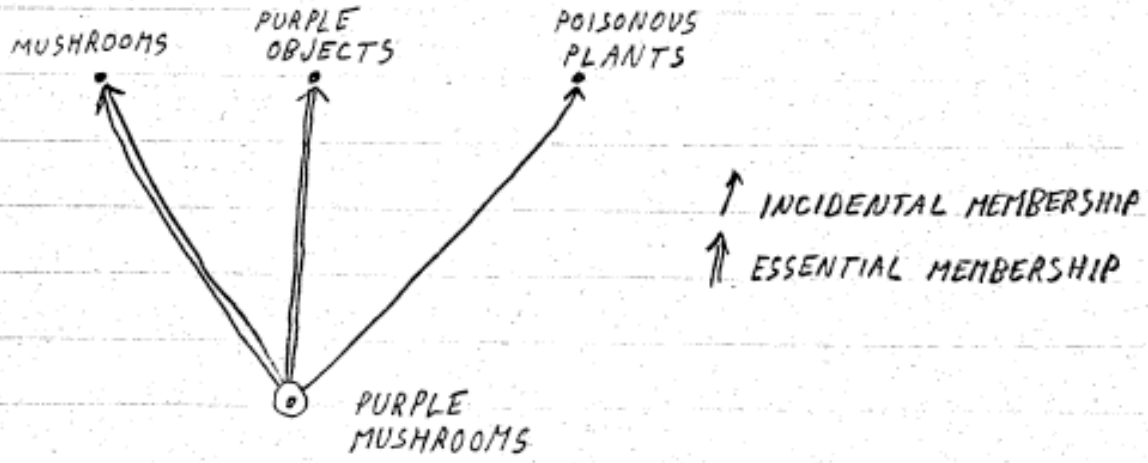


Figure 10 - "all purple mushrooms are poisonous."

control to accomodate some new process, but it is not continually having to write new demon programs or to arbitrate inter-demon disputes. I suppose, in the absence of sufficient hard experience, it is a matter of taste, but I find the manipulation of explicit data structures to be a much less fearsome process than the automated production of debugged programs.

Unfortunately, the digestive process is both tedious and open-ended. Tedious, because the parallel tree-marking process cannot carry much of the load. Some phases of the operation--the marking of all possibly-relevant mandatory nodes, for instance--can be done in parallel, but ultimately the system must sift through possibilities on an individual basis. (I think so, anyway.) Open-ended, because the essential properties may themselves have to be deduced from other things, *ad quasi-infinitum*. The non-membership of an item in some class under a complete exclusive node may also require some effort to establish.

Clearly, what the system must do is to perform as much digestion as it has time for, and then proceed to other things. This means that some potentially-deducible consequences of the new link may go undeduced, but completeness is not the highest mental virtue--humans seem to get along well enough without it. When the system is hurried it will overlook a great many consequences; when it can afford to be more contemplative it will overlook much less. If a particularly promising digestion must be abandoned, the system can mark it somehow and come back later, when it has some spare time. And, of course, the system can come back on the basis of need--looking for the necessary essential properties when it wants to establish some incidental property, for instance, or investigating some particular area that is suddenly of interest.

All of this digestion business is just a form of logical deduction, and not a very efficient form at that. I have taken some pains to include it because it does seem to play a role in human thought, and because Quillian's networks have so often been criticized as being unable to handle quantifiers and the like. As we have seen, such things can be integrated into the present system without undue strain.

But the application of such precise logical statements (*All* purple mushrooms?) seem to me to belong to the domain of abstract sequential thinking, rather than to the kind of intuitive, seemingly instantaneous thought that characterizes, say, the recognition of an elephant. This is why I am not particularly bothered about my inability to find a fast parallel way of carrying out the digestive process--why, in fact, I have not really looked very hard for such a method. I just don't believe that speed in this area is necessary for human-level intelligence.

Perhaps all of this will be clearer if viewed from a slightly different angle. What we really have are two different, though compatible, families of intermediate nodes created by two distinct processes. Some are created by generalization from below, as we saw in the previous section. These nodes reflect the empirical observation that certain sets of

properties appear together in many individuals. ELEPHANT is such a node. Individuals are placed in the ELEPHANT group not on the basis of any *one* property, but because they fit in that group better than they fit anywhere else, and because they do not clash with the ELEPHANT requirements. Any set of properties will do, as long as they are sufficient to rule out the other competing classes; the rest of the ELEPHANT properties are then inherited.

This process of moving a sample down into a class on the basis of whatever properties are readily available is what I have been calling recognition and, as we will see, it can be quite fast. The price paid for this speed is some residual uncertainty that we are really in the right place. We don't really have a *decisive* test for ELEPHANT, but we can trade time for greater certainty: as we gather more and more properties that fit the ELEPHANT description, it becomes increasingly unlikely that we are dealing with some other animal.

The other nodes, the mandatory ones, are created not by grouping together lowly individuals, but by splitting apart higher nodes on the basis of some particular property. Once MUSHROOM has been divided into subgroups on the basis of color, we can look for incidental properties characteristic of the members of these groups--*all* the members. This process can more legitimately give rise to sweeping generalities than can upward-grouping. The mere fact that we have found a lot of poisonous purple mushrooms does not give us the right to say that all purple mushrooms are poisonous, but if we have carefully divided all known mushrooms by color and have found all the purple ones to be poisonous, we can at least state our rule for *known* mushrooms. If we are prepared to believe that known mushrooms are all that we will encounter, we have merely to set up a mechanism for applying this rule, mechanically and inexorably.

The real difference, then, is in how the two systems handle the inevitable uncertainty that arises when we try to deduce a property not directly observed, merely because the world has always been consistent in the past. The upward generalization merely summarizes past experiences; the act of faith occurs when the recognition program assigns a new member to this class on the basis of only a few of the necessary properties, assuming that the rest will be present. In the downward splitting process, however, the risk is taken in formulating the rule itself--in attempting to derive a universal truth from some collection of observations. If we choose to accept this rule, we can then proceed with absolute certainty by logical deduction (or digestion). Ironically, this certainty makes the recognition task harder, not easier. The deductive system is brittle, and since the slightest error can lead to vast absurdities, the digestion must proceed very carefully. The generalization-recognition process is much more forgiving and, I believe, plays the central role in the intuitive-level thought that so far has eluded us.

9. In Search of the Missing Link

So much for amateur epistemology--it's time to get back to nuts and bolts. Specifically, it's time to consider what we really want to use in place of our *ad hoc* menagerie of link species. Clearly, we must find a way of representing all possible inter-node relations with a single type of link (or at most a very small number of varieties). Occam's razor aside, this is a matter of practical necessity: the number of relations in use is huge and ever-growing, and the inventory problem would be enormous.

This inventory problem is solved easily enough by the use of a single, standard type of hardware link with a register into which is written the name of the relation it represents--FATHER-OF or whatever. Alternatively, we could make our link units with three distinct private-line terminals: two running to the two nodes being linked, as before, and the third running off to a node representing the relation. This latter form is preferable, since the relation is itself represented by a node and can thus take its proper place in the IS-A tree, acquire properties, and so on. We might, for instance, want to note that FATHER-OF is a kind of RELATIVE-OF.

This is all right as far as it goes, but it is not enough. A relation is more than just a magic word whose invocation takes us from one node to another--a relation must also have a *meaning*. By its very existence, a statement like "X is the father of Y" tells us something about X: that he is an adult male animal. To say that MARY is the father of Y would generate a clash, since we know that the name Mary is typically reserved for females. We also know a bit about Y: that he or she is considerably younger than X and is of the same species.

So each relation supplies a set of *roles* or *slots* into which the items being related must fit, and to which they become connected by IS-A links (perhaps implicit ones) when the relation is stated. These roles may be very specific classes or very general ones. The WEIGHT-OF relation, for instance, has one role that can only be filled by a certain kind of unit--some number of grams, pounds, or tons--and another role that can be filled by any physical object. The latter category is very large, but not universal: ideas will clash, as will substances. Iron, you see, does not have weight--it has density; a *piece* of iron *is* an object and will not clash in this role.

The meaning of a relationship involves more, even, than the imposition of roles upon the relatees. If we use FATHER-OF in a strictly human context, we find that it conjures up a whole constellation of other relations, roles, and even stereotyped actions. We find that both X and Y are parts of a structure called a family, which has slots for an adult female (X's spouse, Y's mother) and for numerous other people, all connected by a dense network of relations. We find, in most cultures, that X is supposed to support Y while Y is young, and that Y is supposed to act respectfully toward X. And so on.

The problem, then, is to find a representation for links that gives us access to as much of this meaning as possible without doing a lot of deduction. Just as we wanted to simply plug into the ELEPHANT node and inherit all of its properties, we now want to simply hook up X FATHER-OF Y and get all the rest for free. (This is not exactly something for nothing--the hardware, you will recall, is the rather costly secret ingredient.) Of course, we can not let our zeal for efficiency lead us into representations in which certain things cannot be expressed at all, though we will continue to concentrate the system's efficiency where it seems to be most needed.

The battle is far from over. The mechanisms presented from here on are simply my current best guesses, subject to change. Still, I don't think they are too far from the ultimate goal, in spirit if not in detail.

10. The Three-Pronged OF-Link

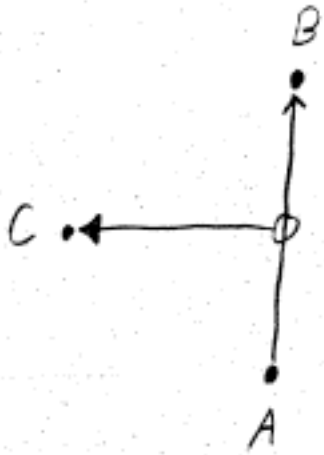
Quite early in the system's development I hit upon a link configuration that seemed to do fairly well all of the tasks I had in mind for it. I call this structure the 3-pronged OF-link. With a few embellishments, this single link type has been used in almost all of my thinking about descriptions and their uses. Other formats have proven to be preferable for particular purposes, but none has shown a comparable generality. An added beauty of the OF-Link is that it is practically a one-to-one mapping of the way property relations are normally expressed in English. The 3-pronged OF-link will be described in this section and will be used in most of the examples in subsequent sections.

Lately, however, I have had some second thoughts about the OF-link. As I have turned my attention from static descriptions of objects, parts, and properties to more complex systems of frames and varying contexts, the OF-link has become increasingly awkward to use. It is still adequate, but I have begun to feel that a link based more on the context-subcontext relationship--"in" rather than "of"--would be preferable. Perhaps, too, I have been too fanatical in seeking a single universal link-type, rather than a few types. I will return to some of these possibilities later on, but for now the OF-link will serve us. I have mentioned these doubts merely to indicate how far all of this is from being chiseled in granite.

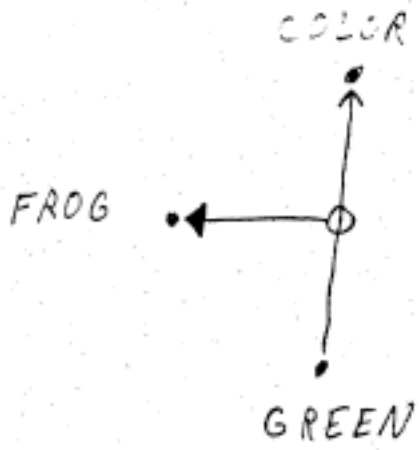
Figure 11a shows the basic OF-link. The relation it represents is usually stated as "A is the B of C" or, if A is not unique in playing this role, as "A is *a* B of C". This format can be used for properties, relations, and for subdescriptions or parts. The rest of Figure 11 illustrates this: "GREEN is the COLOR of (every) FROG"; "ABRAHAM is the FATHER of ISAAC"; "a TRUNK is the NOSE of an ELEPHANT".

Now, an important point: *Each* of these 3-pronged links carries an implicit IS-A link from A to B. Thus GREEN IS-A COLOR, ABRAHAM IS-A FATHER, and a TRUNK IS-A NOSE. Properties and group membership are passed down these links just as they would be over a normal IS-A link. In fact, we can now eliminate the IS-A link as a separate entity: it is merely a 3-pronged OF-link with the OF-prong left unused. So the OF-link both states a property relation and locks the *player node* A into the *role* B. If the player clashes with the requirements of the role, this will be detected by the mechanism described earlier; thus, our suspicion of the statement "MARY is the FATHER of CLYDE".

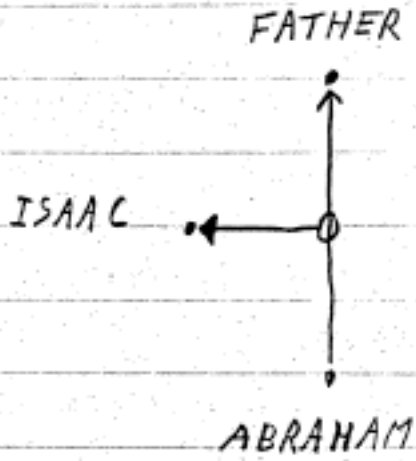
The variability of the little words--a, an, the, every--in the "A is the B of C" formula seems to bother some people. These words do not reflect any difference in the link itself, but are rather signals to the listener about the context and the nature of the things being linked. "The" signals uniqueness, while "a/an" does not; thus, we have "*the* father of Isaac" but "*a* terminal of resistor R1." Individuals--Isaac, for example--get no article at all. (We will see later how such individual nodes are labeled.) To speak of the color of *a* frog is ambiguous: we mean



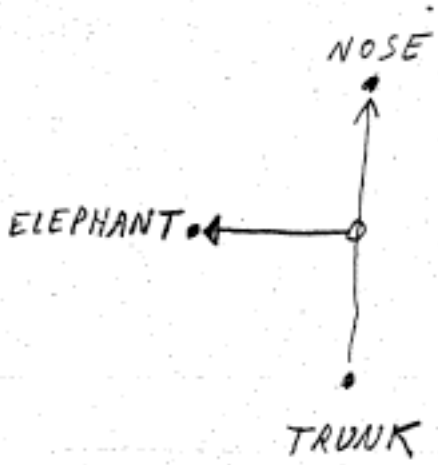
(a)



(b)



(c)



(d)

Figure 11 - Examples of the 3-pronged OF-links.

every frog, and to avoid confusion we had better say that, lest the listener assume we mean some particular, but unspecified, frog. Despite the wording, the links in Figure 11 are all the same and are used in the same way.

These links allow us to create vertical layers of description, as shown in Figure 12, a fragment of the description of an electronic network. A TERMINAL is a PART of an ELECTRONIC COMPONENT; a TRANSISTOR is a kind of ELECTRONIC-COMPONENT and an EMITTER is one of its TERMINALS; T1 is a particular TRANSISTOR and E1 is its EMITTER. T1 inherits the properties of the COMPONENT and TRANSISTOR classes; E1 inherits the properties of TERMINAL and EMITTER. We see here, for instance, that T1 is made of COPPER, because all terminals are.

In the last section I suggested that a relation imposes a role on *both* of the related nodes. We have seen how the OF-link casts the player node into a role, but what about the node at the end of the OF-prong, which I call the *owner*?

There are two answers to this. First, note that if I say to you "E1 is the EMITTER of T1," you know at once that T1 is a TRANSISTOR. Looking at the EMITTER node, we can see that it is defined as a part--a TERMINAL--of a transistor, and of nothing else. So if we--or the system--are told that T1 has an emitter, we are perfectly justified in assuming that T1 is a transistor, and can create an IS-A link saying so. This is a relatively trivial piece of digestion.

Things are more difficult if we are told that "W1 is a WHEEL of X." All sorts of things have wheels: cars, bicycles, even clocks. So if we look above the WHEEL node we find links with OF-prongs running off to dozens of other nodes. X might be any one of these. All we can do, in lieu of further information, is to assign X to a class general enough to contain all the possibilities--MAN-MADE-OBJECT, perhaps. If, later, we learn more about X or more about the *kind* of wheel W1 is, we can then move X down to a more specific category. *Training* wheels, for instance, appear only on bicycles.

The second mechanism is more direct. Certain roles come in *reciprocal* pairs: husband-wife, parent-offspring, owner-property, etc. These paired nodes are explicitly tied together by a mechanism I will explain later. Now, whenever the system assigns a player to one role in a pair--"X is the PARENT of Y", for instance--it must immediately create another link assigning the owner to the reciprocal role--in this case, "Y is the OFFSPRING of X". Each of these roles, of course, carries its load of properties and represents a potential clash. This use of two links seems rather wasteful, but it gets the job done neatly. Note that if we know the sex of either the parent or the offspring, we would move the individual in question down into one of the mandatory sub-classes based on sex: SON or DAUGHTER, MOTHER or FATHER. Note, too, that the reciprocal pair of roles may not be immediate superiors, but might be somewhere up the

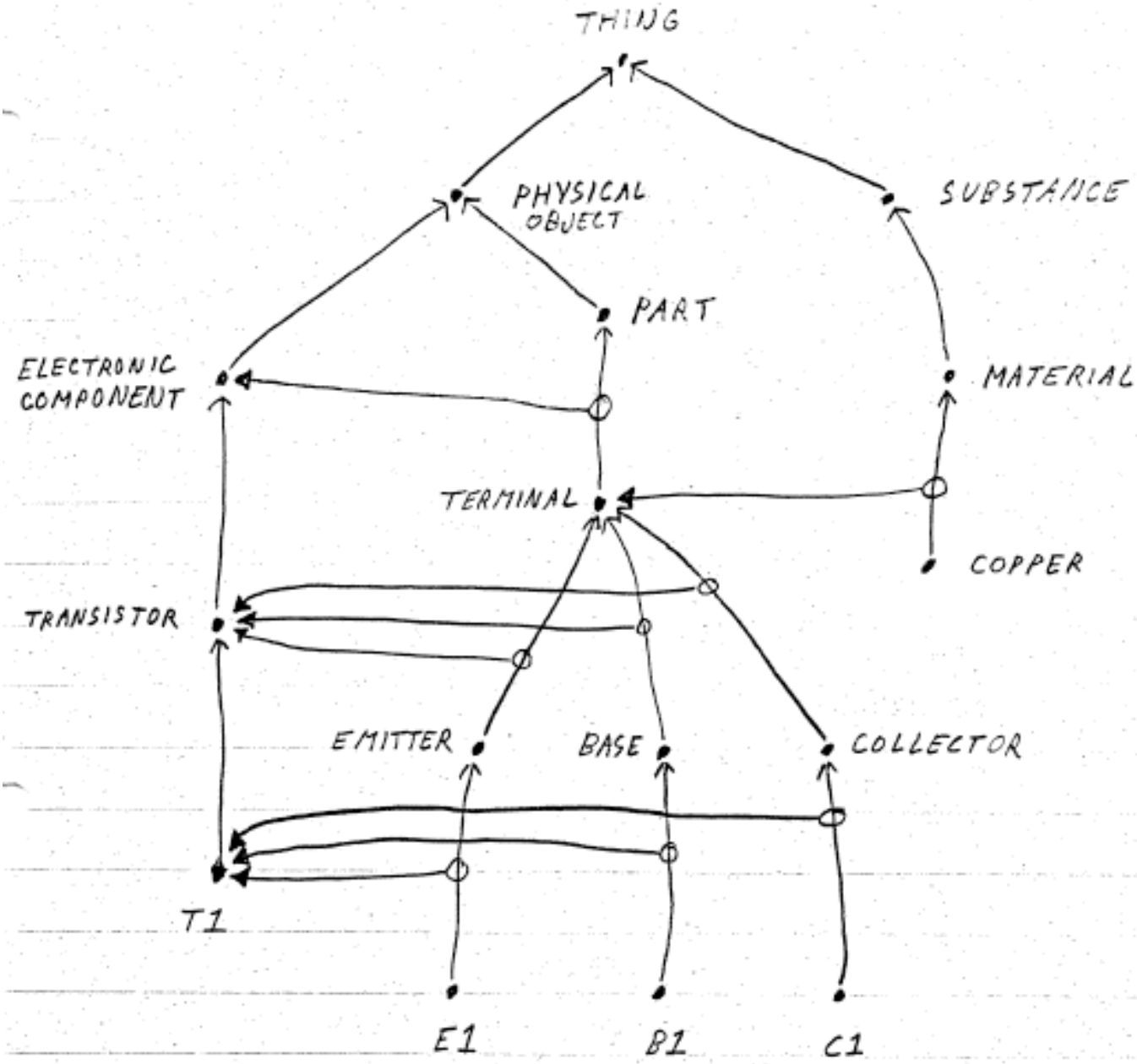


Figure 12 - a fragment of the description of an electronic network.

tree. Still, the search takes only one sweep.

None of this explains how we are to represent a relation like GREATER-THAN, which does not fit into the OF format. Such things are represented by a more frame-like structure which I will explain several sections farther on, after more machinery has been developed.

11. Accessing Formulas

If I want to direct the system's attention to one of its internal nodes--perhaps so that I can tell it some new fact about this node--I have to find some way of specifying which node I mean. I do not have access to the node's internal serial number. The node might or might not have an unambiguous name that is known to both me and to the machine. If no such name exists, how am I to point out the node in question?

The answer, of course, is to construct a phrase that describes the location of this node in terms of other nodes that can be specified by name. Returning to Figure 12, I might specify the node E1 by the phrase "the emitter of T1", rather than by its name. Either form is completely unambiguous, and I indicate that I intend no ambiguity by my use of the word "the". I might instead specify "*a* terminal of T1" which would refer to E1, B1, or C1. If I spoke of "*the* terminal of T1" the system would become unhappy: the word "the" signals that it should find exactly one node, but instead it finds three. It would then have to ask which terminal I mean.

I call such node-specifying phrases *accessing formulas*. Some other examples would be "the length of the nose of Max's grandfather's mistress" or "the mouse that murdered the cat that guarded the cheese in the house that Jack built". The production and decoding of such formulas is one of the major tasks of any natural-language system, and is relatively straightforward in an OF-link system. In this section we will be concerned only with phrases made from *static* relations; verbs like "murdered" and "built" will have to be deferred until after the discussion of context mechanisms.

Let us first consider how to find and mark "the terminals of T1" in the structure of Figure 12. First, we will assume a trivial mapping from names like TERMINAL and T1 to the associated concept nodes. (We will see how to use context to separate multiple word-meanings in a later section.) Our first impulse might be to send a command to all links of the form " ? is a TERMINAL of T1" to mark their player-nodes. This will not work, for there are no such links. The TERMINAL link is attached not to T1, but to a superior node.

A better plan is to mark all of T1's superiors, and then mark across any TERMINAL links connected to any of these. But this is still not right: we have marked EMITTER, BASE, and COLLECTOR--not the desired E1, B1, and C1. We have to move the markers back down to the level of T1, but we can't just sweep downward from the nodes we have already reached. That would mark the terminals of *every* transistor, not just T1. The proper procedure is to move the markers downward over only those links whose OF-prongs are attached to nodes that we marked on the way up. This will bring the markers back to the proper level--no farther--and keep them from wandering off to unrelated terminals. The markers go up the left side of the ladder, across, and down the right side. This is called a *ladder*

sweep.

Sometimes the markers will not make it all the way back down. If we ask for "the MATERIAL of E1" we will get COPPER. We could create a new node below COPPER to represent "the COPPER of E1" but there is no point in this unless we have something to say about it--that it is very pure, for example.

This raises an interesting point: in general, nodes are only created as they are needed. I know that every person has a liver, but if I have ten thousand person-nodes in my head, I do not necessarily have ten thousand liver-nodes. I only have liver nodes for those livers that I happen to know something about. And yet, if you tell me that Clyde's liver is purple, it is a simple matter to create the node L1 and assert (L1 is the LIVER of CLYDE) and (PURPLE is the COLOR of L1). Actually, the L1 node would not be given the name "L1" or any other name. It would just be "Clyde's liver."

Long chains of these relations should in principle be no harder to follow. If we have "the length of the nose of the mistress of the grandfather of Max" we simply use one ladder sweep to get from MAX to the node representing his grandfather, another to get to the mistress, and so on. I am assuming that the system can convert "Max's grandfather" to "the grandfather of Max" without too much trouble.

What if we ask the system to mark "all men whose grandfather's mistresses have long noses"? First we mark all men. Then, as above, we send out a chain of markers from each man to his grandfather, then to the mistress, to her nose, and finally to the nose's length. These steps can be done for all men in parallel. Some of these chains will end on the node LONG and others on the node SHORT. (We are assuming quantum noses here.) Many of the chains will die along the way: some grandfathers probably had no mistresses--at least ones that the system knows about. We then simply place a new marker on LONG and propagate it back along the marked chains to the original men. So the men whose chains ended on LONG become marked, and the rest do not. This is tedious, but not unreasonably so, given the problem. There are other marking strategies which would work equally well.

By now, a picture should be emerging of how all of this might be used in a recognition program. If we are faced with a creature that has webbed feet, a flat bill, and irridescent green feathers on its head, we simply mark all web-footed creatures with one bit, all flat billed creatures with another, and all green feather-heads with a third, then look for the intersection--the set of all nodes with three marks. If we have more features than marker bits we can mark a few features, intersect, mark the still-viable candidates with a single bit, and repeat until we're done.

Of course, this intersection-finding is not the whole story of recognition. One man's large nose may be another man's medium, and aside from such blurry boundaries we will often get errors and freakish

exceptions--the famous three-legged cow, for example. Finding the intersection of properties is a very quick and fairly reliable way of getting an initial hypothesis, but we still need to devote some effort to evaluating and improving the fit. Also, in fields like vision, there must be some interaction between the evolving hypothesis and the feature extractors. This representation is a powerful tool for a recognition system to use, but many of the problems outlined in my original working paper [4] still must be worked on.

This fast intersection is a part of what I meant when I said the system "felt right" as a model of human recognition. Most people, when confronted with a recognition task, feel that the hypothesis emerges in a sudden flash of inspiration--if it comes at all. Before this flash is a period of aimless input gathering, and after it may be a period of fitting and weighing, but there normally seems to be very little possibility-by-possibility search, unless all else has failed. I submit that the sudden appearance of a single node at the intersection of a bunch of features is about as close to a flash as we are ever likely to get. In most other systems, including my suggestion-demon network, the finding of a hypothesis involves much more work than the subsequent verification.

I don't have too much to say about the process of generating an appropriate accessing formula for a given node. It seems like a relatively straightforward process, though perhaps a bit more complex than the decoding procedure. In a way, it is like giving directions to a motorist: First you tell him to go to some mutually-known landmark near the destination--in this case a node with a name that he knows--and then you give him the series of street names corresponding to the best path from the landmark to the goal. This is something of an art: the best path is the one most easily followed, not necessarily the shortest. You might describe certain intersections along the way, so that the traveler can verify his progress. Thus we might have "the length of the nose of Max's grandfather's *young Austrian* mistress." The adjectives may provide the listener with new information, but if he knows (or knows of) the lady, they will instead provide a redundancy useful for checking whether he is following the path properly.

Obviously, if the person creating the formula is to be so considerate of the listener's needs, he must have a good model of the listener's information network. The words and the nodes used to anchor the formula must have a meaning for the listener as well as for the speaker. Speaking to my brother I might say "Champ's mother", but to a stranger I would have to say "the mother of the German Shepherd my family used to own." I must divide all of my knowledge into packages on the basis of who might know any given item--everyone, just my family, all scientists, all M.I.T. students, and so on--and I must operate within the proper subset of these packages when framing an utterance for some particular audience. We will see a possible implementation for this in the section on contexts. Of course, I have just scratched the surface of the formula-generation problem; this looks like an interesting thesis for somebody.

12. Handles and IN-Links

In the next few sections I will describe a number of embellishments to the standard OF-link. These new mechanisms are, in general, not designed to represent new properties of real-world objects. Rather, they implement what, in my previous working paper, I called the *auxiliary information* in a description: statements about how the descriptive information is to be interpreted and used. Some of these new mechanisms may seem to be weakly motivated at first, but they will be very useful in certain applications we will encounter later.

Before proceeding, I should point out that once links have reached a certain level of complexity it becomes possible to simulate some of these mechanisms. For instance, instead of introducing a four-pronged link, it might be possible to get the same effect using a pair of three-pronged links. In deciding what new link-types to present here, I have tried to optimize for ease of understanding, introducing new mechanisms where they seemed to represent natural and often-used relations, but not for particular quirky cases. The boundary is largely a matter of taste. In a hardware implementation the boundary would be a matter of economics, and it might fall in a very different place.

The first of these new mechanisms is something I call the *handle* node of a link. Each link comes factory-equipped with a full-fledged node of its own. This node can either be thought of as an integral part of the link-unit or as being attached to a fourth prong--the effect is the same in either case.

This handle node represents the *item of information* stated by the attached link. Statements about this piece of information can now be represented by simply attaching the appropriate links to the handle node. About a statement X, we can now say "X is a LIE" or "WORLD-ALMANAC is the SOURCE of X". It is now a simple matter to label the essential and incidental links of a mandatory node. In short, the handle gives us a way of getting hold of the associated link--hence the name.

Note that not every link can have something said about it--this would lead to an infinite profusion of links. We can not even afford to give every handle node an identity link--an exception to the rule stated in Section 5. Perhaps every handle-node, by virtue of being a handle-node, should be considered to have an *implicit* link stating "X is a LINK", but I doubt that this would make much practical difference.

We might be tempted to say that handle nodes are optional, that they are only attached to those links that we have something to say about. This is impossible, however, because the handle nodes serve another purpose as well: they provide us with a way of marking sets of links. At any given instant, we may want only some subset of our knowledge to be *active*; the inactive links should play no part in any computation. This allows us, for example, to have multiple inconsistent world models represented in the

same knowledge net, but using different sets of links. By marking one bit in the handle nodes of the active links, and by ordering that links not so marked ignore all CPU instructions, we can achieve the desired effect. We will see in the section on contexts how such activation markers propagate.

Actually, we will see a bit about this topic right now, as we consider the *IN-link* (or, sometimes, the *IN-prong*). Every one of our OF-links has yet another prong, running off to a node called its *context*. It is along this prong that markers travel to activate the link. Normally, if a link's context is activated, the link will be activated as well.

The relation between a piece of information and its context is normally specified by the English words "in the context of" or just "in". Thus we have "ABRAHAM is the FATHER of ISAAC *in* the BIBLE" or "GRAVITY is an INVERSE-SQUARE-FORCE *in* NETWONIAN-PHYSICS". If we are discussing some other book or Einsteinian physics, these particular contexts would not be activated and these pieces of information would play no part in the processing. There is very general context, GENERAL-KNOWLEDGE, that contains those facts that have no place else to live. The context nodes themselves are tied into a tangled hierarchy based upon the relation "is a sub-context of" rather than "is a kind of", but that's an issue I don't want to get into yet.

Note that a frame (in the Minskian sense) is a kind of context, binding together a group of relations. Thus we have "MARY is the MOTHER of CLYDE *in* the JONES-FAMILY" or "MURRAY is a BASSOONIST *in* the BOSTON-SYMPHONY-ORCHESTRA" or "SUSAN is the GUEST-OF-HONOR *at* the PARTY". (Well, you can't win them all! I think that "at" replaces "in" to flag a particular spatio-temporal context, but this is tricky.)

Sometimes it is hard to know where the boundary between OF and IN falls. Is the crankshaft a part *of* a car or a part *in* a car? Both, I suspect. This question will be treated at length later on, as will the whole issue of part-whole relations. In some of the context schemes I have tried it is useful for a node to have many IN-prongs. In this case, we don't have *prongs* at all, but separate *IN-links* running between the link's handle node and the various context nodes. This, too, we will cover later.

In fact, I have said all that I really can about IN-links without opening up the whole context issue, and we aren't quite ready for that yet. I just wanted to give you a quick look at the machinery that exists to meet the context problem, so that it won't seem too strange when we do need it, and so that I can sneak a few IN-links into examples in the meantime.

13. Exceptions

This entire descriptive system is based upon the principle that it is more efficient for an individual to inherit a large group of properties from an appropriate superior class than to state all of an individual's properties directly. Much of this advantage is lost if the entire set of properties must be used as is, without any modifications. The most valuable classes--those with the largest bundle of properties--would have practically no members. In the system I have described, it is trivial to add new properties but somewhat harder to cancel out inherited ones.

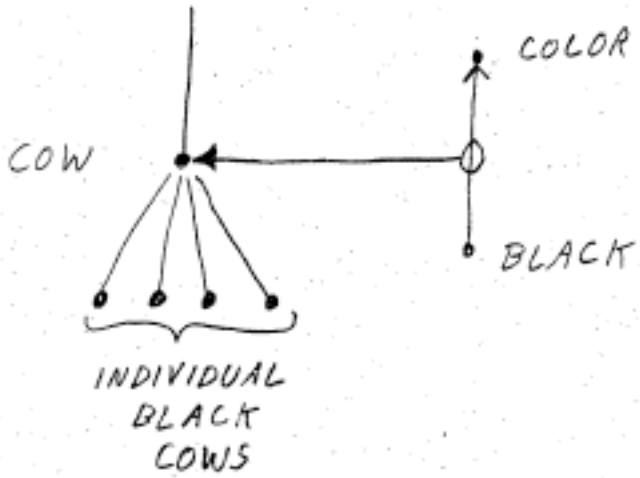
The tangled hierarchy system eliminates some of the need for a property-cancelling mechanism. Let's assume that all of the cows we have ever seen are black. This color is attached to the COW node and is inherited by all individual cows. Then we see a brown cow. Instead of trying to cancel the BLACK property and add BROWN for this individual, we can reorganize the COW node as shown in Figure 13. We now have two kinds of cows, black ones and brown ones. Both are perfectly legal classes, not exceptions. The rest of the COW properties remain with the superior COW node and are inherited by both types of COW. The exception has been *normalized*.

Normalization has its place, but it is not the whole story. For one thing, we would be reluctant to change important high-level nodes every time some random exception comes along. Imagine, for instance, a feathered hippopotamus. You have never thought of such a thing before, and never will again (if you are lucky). Is it reasonable to suppose that for the sake of this single image you have permanently divided the class of hippopotami into feathered and non-feathered varieties?

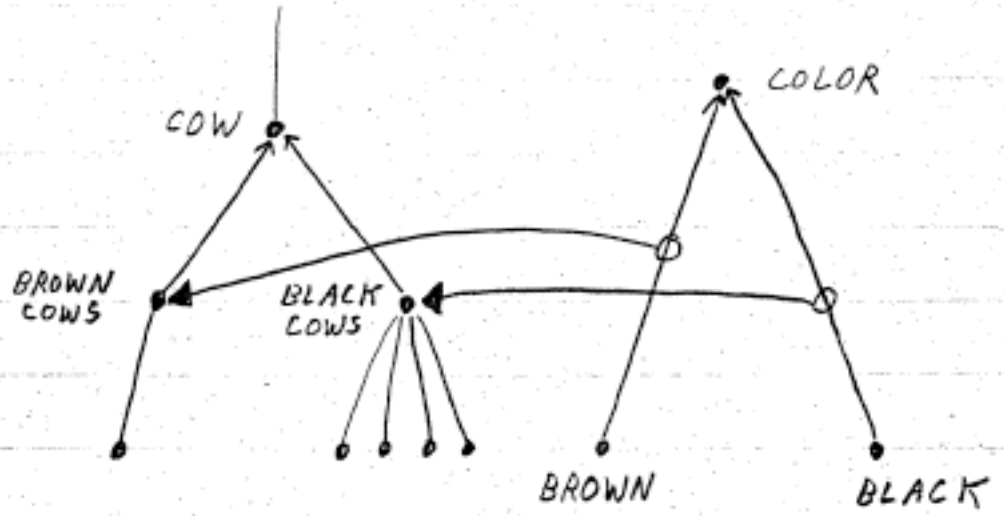
For another thing, the normalized sub-classes are too equal. Often we *want* the system to assume some specific default value for a property unless this default is specifically contradicted in the description. You may know that albino elephants exist, but if I say "elephant" to you, you will almost certainly think of a gray one. It is possible to label one of the sub-classes as the typical one, and to place new class-members into this sub-class, but this becomes very clumsy if a class has many default properties.

So we still need a true exception mechanism: some way for a given node to cause certain superior links to remain silent when an access is made to some property of that node. To achieve this, we have a new link-type: the *exception link*. (Actually, we could do this job with an OF-link, but I feel that the separate link-type makes for a clearer presentation.) The exception link runs from the node representing the exceptional individual to the handle node of the link that is to be cancelled. Figure 14, for instance, tells us that Clyde is a *white* elephant.

Whenever we are about to begin a ladder sweep to find some property



(a) BEFORE REORGANIZATION



(b) AFTER REORGANIZATION

Figure 13 - Reorganization of COW node to allow brown cows.

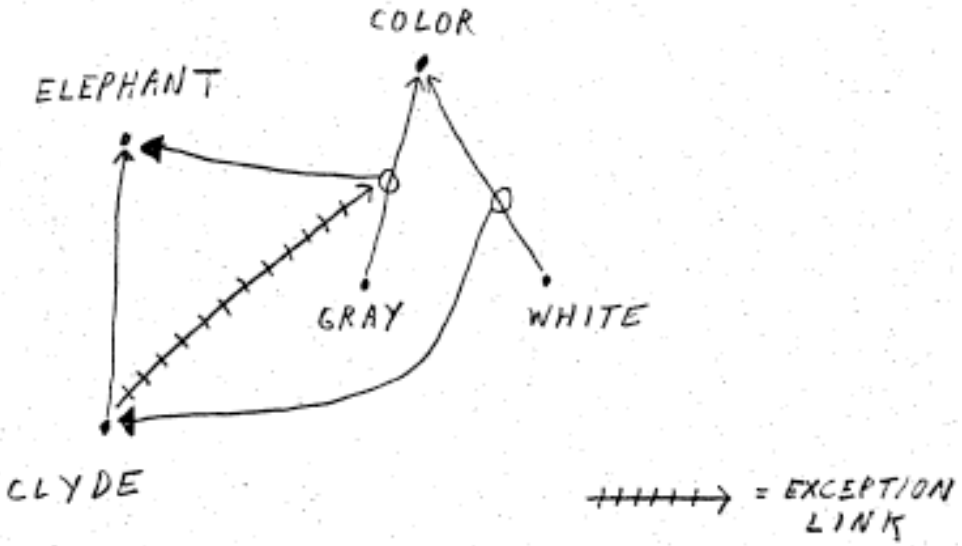


Figure 14 - "Clyde is a white elephant."

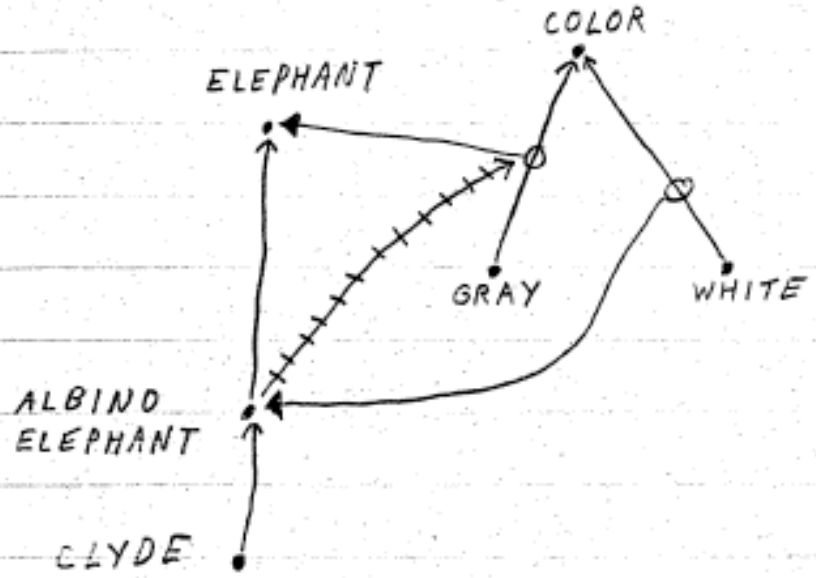


Figure 15 - "Clyde is an albino elephant."

of CLYDE, we first send out markers over CLYDE's exception links to set a cancellation bit in the handles of the cancelled superiors (or perhaps to clear an activation bit). Then the sweep proceeds as usual, except that the CPU commands are addressed only to non-cancelled links. Note that the cancellation signal must reach a link *before* any markers have crossed it; it would be very hard to track down and neutralize all of the propagating descendants of an illegally-passed marker.

In fact, the above picture is over-simplified. An exception can be anchored to a class-node as well as to an individual node, and it should be inherited by the class-members, just like any other property. In Figure 15, for example, we see the *class* of albino elephants, of which CLYDE is a member. Note, too, that because of this system's habit of sometimes expressing properties in terms of group membership, an exception link might have to run from a node to one of its own superior IS-A links. This looks dangerous: we have exception markers cutting off the path to superior nodes, and superior nodes--in mid-sweep--generating their own exception markers. It would appear that the only way to avoid potentially chaotic races between markers is to proceed carefully, step by step. First the exception markers are sent out; then, over the still-valid upward links, the first layer of superiors is marked; then, the exception markers go out from these nodes; then the next layer of superiors is marked; and so on. This is orderly, but now an upward sweep takes twice as long.

Can the exception and superior marking steps be done simultaneously? I think so, since a node would have no reason to cut its own *immediately* superior IS-A links--any more than it would have reason to cancel its own properties--but only those links *above* one of its superiors. Thus, from any given level, we would be sure that the exception marker would reach a link in the current step, and the regular markers would find the bridge already burnt when they arrive one cycle later. There is still potential for trouble if there are redundant paths of different lengths to a superior node, but this is a perverse situation and I suspect that it can be dealt with by forbidding such redundancies where possible and by adding kludges where necessary.

Given an efficient exception mechanism, we can afford to make liberal use of default assignments, with all the advantages Minsky has claimed for them, and exemplars of the sort I described in my previous paper. The various links in a description can be labeled according to how reluctant the system should be to take exception to each. Some may be sacrosanct: no individual may alter such a node. Others may be mere default conditions, such as the red color of an individual's image of BALL; these can be changed with complete impunity. Such properties are, of course, attached to the handle nodes of the links in question. These *discrepancy values* play an important role in recognition: The more exceptions an individual must make to fit into a particular category--especially *important* exceptions--the greater is the likelihood that the individual would be happier somewhere else.

14. Immediate Connection

There are really two distinct senses in which properties and memberships can be assigned to a node. The first is the sense we have seen in all of the examples so far: the properties and memberships attached to a node are meant to apply to all of the inferiors of that node. In addition to this, however, we need a way of saying things *about the node itself*--about its characteristics as a group or about the way it functions as a role in a larger description. For some reason, this distinction was a very hard one for me to grasp: it was the first of these auxiliary mechanisms to be developed and it caused me a great deal of trouble until I figured out what was going on.

Consider, for instance, the statement that a person has two eyes. We might represent this as shown in Figure 16. (The wiggly line is an IN-link.) But if the number property were connected to the EYE node in the usual way, each individual eye would inherit the property of twoness. This is clearly not what we intended by this statement. The twoness refers to the EYE node itself--specifically, to how that node is to be filled in the context of a particular person--and not to any member of the class EYE.

To represent this distinction, we create a new kind of link-to-node connection, which I will call *immediate connection*. In contrast to the old kind of connection, which is meant to be inherited by subordinate nodes, immediate connection indicates that the property or membership is meant to apply only to the node in immediate contact with the link, in that node's capacity as a group or role. I prefer to think of these *connection senses* as being a property of the connection itself--perhaps the link unit prong is attached to a different terminal of the node unit--but it could just as easily be an "immediacy bit" associated with the prong or even a distinct family of link units. In the diagrams I will represent immediate attachment by a double chevron near the point of link-node contact, as in figure 16.

It is up to the various accessing programs of the CPU to treat the two connection senses differently. During ladder sweeps and other property-accessing operations, only the normal connections are of interest. The immediate connections are consulted during the various digestive processes of the system when, for instance, the possibility of adding another EYE node to the CLYDE description is being weighed. This is not to say that we can *never* use a parallel sweep to find immediate properties: we might, for instance, want to list all the parts of an elephant that come in pairs. But the immediate connections stay out of the way in normal property-finding sweeps.

Immediate connections can be used to represent many things. As we have seen, they provide a way of indicating how many players a role may have in any given description. Similarly, we can label certain roles as being optional, such as a moustache on a man's face. Some of the mechanisms we have mentioned earlier, such as the labelling of certain

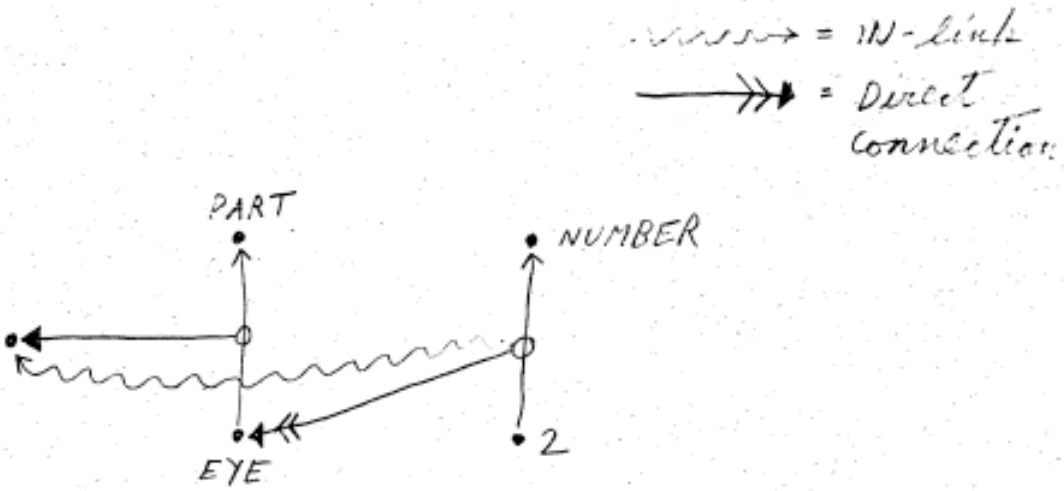


Figure 16 – "Two is the number of eyes in a person."

nodes as mandatory or exclusive or the statement of a node's reciprocal are also cases of immediate attachment. A group may be labelled as empty or a member of a group may be labelled as typical. A node may be tagged as representing a group or an individual in a similar manner.

This whole business of individuals is more subtle than it might appear. I was tempted at first to make all of an individual's properties connect immediately to the individual's node. After all, there can be no question of these properties being inherited by an inferior node, for individual nodes have no inferiors. (The different appearances of an individual at different times could be treated as inferiors of that individual's node, but I have found the context system to be more useful for representing the changing properties of an individual.)

But we can change an individual node to a class with surprising ease, as when we refer to someone as a Hitler or a George Washington. The boundary can often be very blurry. Is Lassie a dog or a kind of dog? I understand that there were many dog-actors who played this role over the years. T1 may be a specific individual transistor in a circuit diagram, but if that circuit is manufactured, each unit will have its own version of T1, and some of these may have exceptional properties. So the individuality of a node may change from context to context.

This being the case, I decided that to impress individuality upon hundreds of incoming property links was a mistake. It is no harder to use normal connections on individual nodes, so that to create a metaphor the system merely has to hook in an inferior node and create an exception to the single INDIVIDUAL property of the superior. I do not yet have a good answer for the question of how we know, for example, that when I call someone a Hitler I am referring to his personality and political philosophy, rather than to his blood type or place of birth. Metaphor is a complex mental operation.

There may be a few more uses for immediate connections that I haven't really worked out yet. Consider the statements "a resistor is a component *in electronics*" and "a crankshaft is a component *in a car*". If we have a specific resistor we are still in electronics, but if we have a specific crankshaft we are in the context of *some particular* car. In one case we create an instance and in the other case we don't. Perhaps this difference can be flagged by connecting the IN-link immediately to ELECTRONICS and normally to CAR, but the context system is still too muddled for me to be sure of this.

15. Node Names and Speech Recognition

At this point in the presentation it would be nice to pull everything together and demonstrate its operation in a few extensive examples of *static* description frames--those that do not have a need for multiple contexts to represent time-varying features. Examples of this type might include the description of an electronic network, a web of family relationships, or the description of a disease in terms of its symptoms. Unfortunately, I do not have any very complex examples worked out yet. The tools I have described so far are still new, there are still some pieces missing (minor ones, I think), and the unsettled state of the context system keeps sending tremors through everything else, since it is tied to the format of the basic link. Once the context mess is straightened out, the development of such examples will be my first priority.

I would, however, like to present one example that is at least partially developed: the mechanism for associating names with nodes. This will involve some hand-waving, but hopefully not an intolerable amount.

The first point to be made is that the node representing the *concept* COW and the node representing the *word* "COW" are distinct, though linked together. A COW has four legs and eats grass; "COW" has three letters and rhymes with "how". COW IS-A MAMMAL; "COW" IS-A NOUN. "COW" is the NAME of COW. A NAME IS-A WORD. COW is the MEANING of "COW". And so on.

A concept node may have many name nodes or none at all. We have seen how accessing formulas can be used to reach unnamed nodes or those whose name might not be shared by both the speaker and the listener. I would guess that less than ten percent of the nodes in a person's head have names--assuming, of course, that people *have* nodes in their heads. Unless it is a nonsense word, every word has at least one meaning; some have many. We will see later how the context system can be used to eliminate this ambiguity in most cases, and also how multiple languages can be handled.

We must attach to each word node a spelling and a pronunciation--one a sequence of letters, the other a sequence of phonemes. There are at least two possible ways to represent such sequences. One way is to use a master WORD frame with perhaps twenty role nodes representing absolute letter positions: FIRST-LETTER, SECOND-LETTER, and so on. The spelling of "COW" is then easily indicated by filling these roles: C is the FIRST-LETTER of "COW", O is the SECOND-LETTER of "COW", W is the THIRD-LETTER of "COW". The pronunciation could be stored by a similar method. Very long words could be broken up and stored as a sequence of sequences.

Such a structure may have its uses, but I don't really believe that people use absolute-position schemes for storing spellings or pronunciations (except, perhaps for a few crossword-puzzle fiends and Scrabble hustlers). If we did, it would be a simple matter for us to list

every word we know that fits the pattern -AT-E-Y, simply by intersecting the sets of words with these particular letters in the specified positions. Any word node that collects all four marks is a winner. (Did you think of "battery"?) In fact, most people cannot even tell you the seventh letter in "Mississippi" without counting.

All of this suggests that these sequences might instead be stored in a list structure, with a specified first member and a string of other members linked by SUCCESSOR-OF relations (see figure 17). Note that the individual letters in "COW" are not the alphabet letters themselves, but *instances* of them. This makes it possible to speak of "the second S in Mississippi" as a distinct entity.

The list-structured sequence has some important advantages in representing phoneme sequences. Speech is an irregular incoming stream, and a burst of noise might cover an unknown number of phonemes. This sort of thing would wreak havoc upon a system tied to absolute letter positions. The list structured system, on the other hand, could pick up the next intelligible sequence, locate all words containing that sequence *regardless of its position within the word*, and proceed from there.

Consider how the system might recognize the word "today" as it is usually mumbled. The incoming phoneme sequence might look like this: T-sound; some nondescript vowel; T or D sound; long A-sound. The CPU orders every word node to mark its first phoneme node if that node is a T-sound. The mark is passed to the second node if that node is any member of the vowel class; otherwise the mark is erased. Then on to the T or D, and finally on to the long A. If this is not the end of the word, the mark is killed anyway. Probably only one mark will have survived the journey, especially since we may have pre-selected the original set of words according to part of speech or subject area. The mark is then sent back to the word that spawned it, and we have a winner.

In effect, the string of phonemes attached to each word-node is acting as a finite-state acceptor for the incoming stream, and all of these acceptors are acting in parallel. If the governing CPU program is smart enough, these acceptors can be quite flexible in dealing with such things as indistinct word boundaries. One set of marks can continue on with the old set of words, while another set starts from the beginning of what might be a new word. Both sets can be stepped along at the same time until one or the other dies out. This does not necessarily happen in strict real-time: the incoming sequence is almost certainly buffered so that the system can recover from bad decisions--an over-zealous pruning of the initial possibility set, for instance. Obviously, it will take a lot of work to implement these vague ideas, but this system does seem to make the task more manageable.

It is fun to speculate about this sort of system. Perhaps, for instance, the concept nodes live in one part of the brain and the word nodes live in another. (Remember my earlier speculation about the use of

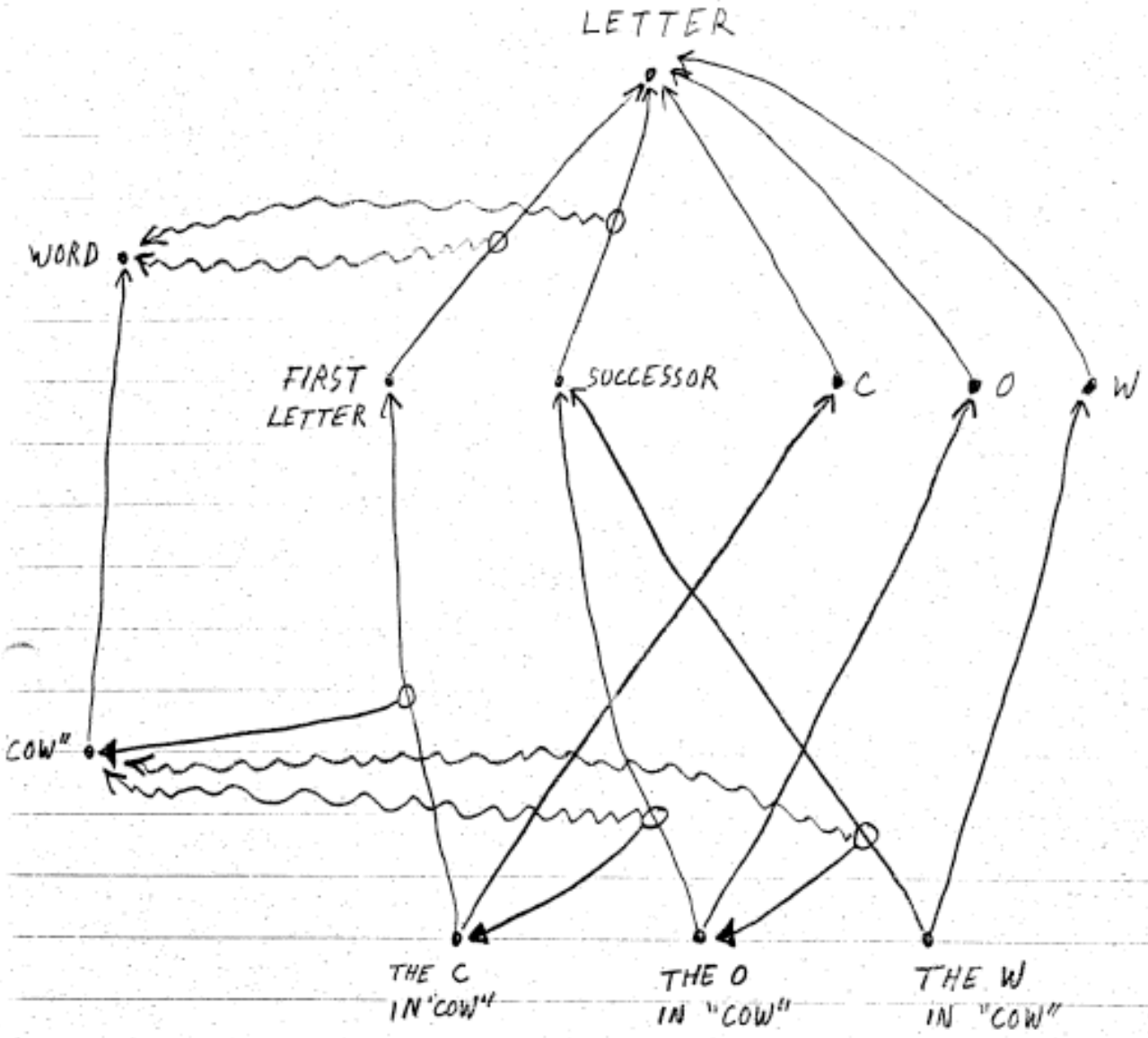


Figure 17 - "a cow by any other name..."

neighborhoods to save crossbar capacity?) Suppose, too, that the supply of direct connections between the two areas is inadequate for modern needs, that it gets used up in early childhood. It might then be very hard to form a new name-meaning association quickly, and the associations that are formed might be very indirect and unreliable, like a phone connection from Boston to New York by way of Alaska. This might explain the plight of people like me who can remember both names and faces, but have a terrible time keeping the two tied together. The list-structured form I have suggested for words might also explain why, when we can't quite remember a word, it usually comes back if we can get the first letter. The second letter, unless it is very unusual, gives us much less help. Such ideas are not to be taken seriously, however.

16. The Context Hierarchy

The word "context", as I have been using it in this paper, refers to a number of related topics. First, there is what I call *absolute context*: pieces of information (links) can be valid in some contexts and completely inoperative in others. This is similar to the use of "contexts" in CONNIVER [10] to allow multiple inconsistent world-models to co-exist in the same data-base. Then, there is the use of contexts as *habitats*. In any given context we will expect to find certain things, and these expectations can be an important tool in the disambiguation of inputs. In a zoo, one is predisposed to see lions, tigers, and elephants; on a farm, one expects cows, pigs, and chickens. Finally, there is the use of some sort of context-related grouping mechanism to create complex frame-like structures of wholes, parts, sub-parts, and all of the relations that tie these parts together.

Taken as a whole, the context package is by far the least-developed part of the system. At present, I have some ideas about how each of the above-mentioned context effects can be achieved separately, but I have not yet found a way of pulling everything together into a coherent system with well-defined areas of responsibility. I don't think that I am too far from such a synthesis, but it did not seem wise to further delay this paper while I work on this. So what we will see in this and the following sections are a collection of fragments that look to me like pieces of the answer, but that do not fit together very well. They should, at least, serve to illustrate what kinds of things need to be done in this area, if not exactly the right ways of doing them.

We will begin by considering the absolute context system. By "absolute", I mean that in a given context a fact is either present or completely absent--never anything in between. This is different from the habitat system, which pushes certain items into the foreground, but which never completely rules out any possibility. As we will see, both systems use the same context hierarchy, but in different ways.

We have already seen most of the machinery that we will need for implementing absolute contexts. We have seen how, by setting or clearing certain bits in the handle-node of a link, we can label that link as active or inactive. We have seen that the CPU can specify, in its commands to the link units, that only those links marked as being active should respond. We have seen how activation markers can be passed along IN-links from a context node to the various links that reside in that context. What we have not seen is anything about the nature of these context nodes or about how an appropriate set of them can be activated at once.

I think that we want to use the same nodes to represent both conceptual entities and contexts. Consider a node like TEXAS. It is a perfectly good conceptual node--a STATE of (and *in*) the USA--with an appropriate set of properties: an area, a population, a capital, and so on. Yet TEXAS also functions as a context: *in* TEXAS certain laws are in effect,

and certain other facts are true which are not true everywhere. It would, of course, be possible to create distinct (but connected) nodes for TEXAS-the-thing and TEXAS-the-context, but I don't at present see any advantage in this. As long as the CPU program is able to keep track of which marker bits it is using to tag active contexts, and which are being used to trace out properties, there should be little danger of confusion. To its superiors, inferiors, and properties, TEXAS is a thing; to the facts attached to it by IN-links, TEXAS is a context. So, for now, let us assume that every node is potentially both an entity and a context, though some nodes might not make any particular use of one or the other of these roles.

Context-nodes can represent many types of grouping. Some are spatial: a given fact might apply in Texas, on the moon, in the desert (*any* desert), or in my office. Some are temporal: a fact may be true in 1848, in the Nineteenth Century, on (any) Sunday, or in the reign of Hammurabi. Some contexts refer to various subject areas or to other conceptual mini-worlds, real or imaginary: a fact might be true in chess, in Newtonian physics, in war, in *War and Peace*, in Greek mythology, or in Clyde's opinion. Note that most of these contexts also serve as entities, with their own IS-A links, properties, and so on.

So we now have a large and varied set of context nodes. Each piece of information is hung in some context by an IN-link, and it becomes active or inactive according to how we have marked the context node. Inactive links are completely out of the way--they have no effect on processing speed.

Does this solve the problem? Not really. It will very often be the case that two contexts will differ in a few facts or a few dozen, but will have many thousands of facts in common. This is especially true in the representation of the sequential stages in a narrative or in a plan developed by a problem-solving system. (See [3] for examples.) It would obviously be wasteful to re-copy each of these shared facts into every context that uses it, or even to encumber each fact with multiple IN-links. It is much better to include the shared facts in a single large *super-context*, to which the sharing contexts can be attached as *sub-contexts*. Each sub-context inherits all of the facts in the super-context, except those which it specifically cancels; the sub-context may, of course, add new local facts of its own. (This idea is descended from the CONNIVER data base [10], by way of my own packet system [4].)

TEXAS, then, is a sub-context of USA; DALLAS, in turn, is a sub-context of TEXAS. Anything generally true in the USA--that women can vote, for instance, or that slavery is illegal--is also true in TEXAS, unless there is specific information to the contrary. Note that this inheritance of facts from a context to its sub-context is not the same as the inheritance of properties along IS-A links: the CAPITAL *of* the USA is WASHINGTON; the CAPITAL *of* TEXAS is AUSTIN. Like the IS-A relation, the sub-context relation is both directional and transitive.

All of this suggests that the sub-context relation forms its own hierarchy using hardware sub-context (SUBCON) links. This hierarchy is distinct from the IS-A hierarchy: it has its own links, but it uses the same collection of nodes. Since we have usurped the up-down direction for the IS-A hierarchy, we will speak of the SUBCON hierarchy as running left to right, with the more inclusive contexts to the left: USA is to the left of TEXAS, while DALLAS is to the right; 1812 is to the right of NINETEENTH CENTURY. The system sweeps markers leftward from a node to mark all of its super-contexts, and rightward to mark all of its possible sub-contexts.

To activate a given context, then, we simply mark it as active and sweep the activation markers to the left, marking all of that node's super-contexts as well. The links hung in *all* of the marked contexts are then activated, and this is the set that is used in any subsequent computation.

Recall that under certain unusual circumstances--for instance, to find what color cows are--we had occasion to sweep property markers *downward* from a node. Similarly, we will sometimes have reason to sweep activation markers *rightward* from a node. Suppose, for example, we want to know if casino gambling is legal in the USA. The system would first activate the USA context normally by sweeping leftward, but would find nothing relevant to the question. Then it would mark rightward, activating the sub-contexts of USA, and would find the answer: legal in NEVADA, not legal in other states. It seems unlikely that the system would ever want to compute in this condition, since directly contradictory sets of facts would be active simultaneously. We will see some other uses for rightward sweeps later, in the section on habitats.

It seems clear that this SUBCON hierarchy should be tangled. We might, for instance, want to create a context representing the desert in Texas in the summer of some particular year, letting this new context inherit all of the information resident in its disparate super-contexts. We can then hang in the new context any information valid only for this specific combination of contexts. The situation is closely analogous to that in the tangled IS-A hierarchy, except that resident information links have replaced properties as the things being inherited.

I am not sure whether the context hierarchy ends in a single leftmost node, analogous to the THING node of the IS-A hierarchy. Clearly there are nodes for EVERYWHERE and ALWAYS, the most inclusive nodes for space and time. Probably there is a node named GENERAL (or whatever) that encompasses all subject classifications. A REALITY node might cover everything that the system believes to be literally true, as opposed to FICTION or to the SINCERE-BUT-MISTAKEN-BELIEFS-OF-OTHERS. But exactly how these nodes are related, and whether they are all sub-contexts of an even more general node, is at present a mystery to me. I don't see what such a node would be good for, except to tie up loose ends.

There are a few important points yet to be made about this new hierarchy. First, note that *if A is considered a part of B when A and B*

are treated as entities, then A is a sub-context of B when both are treated as contexts. The Battle of Gettysburg, for instance, is a part of the American Civil War; the Battle of Gettysburg context is a sub-context of the American Civil War context. Anything true in the war generally--that the infantry uses muzzle-loading rifles, say--is also true in the battle, unless it is contradicted. Once again, this is *not* the same as the inheritance of properties: the duration of the *battle* was not four years. We will see more about the subject of parts vs. sub-contexts in the section on frame-building.

The other point is that *every IS-A link functions as a SUBCON link*, but that the reverse is not true. The Battle of Gettysburg is a *sub-context*, as well as a *kind*, of battle. Anything true in battle generally is also true in this particular battle. The Battle of Gettysburg is also a SUBCON of WAR, by way of the AMERICAN-CIVIL-WAR node and its upward IS-A link. CHESS is a sub-context of BOARD-GAMES, and ELEPHANT of ANIMALS. Thus, when we speak of a leftward marker sweep, we really mean leftward and upward; rightward really means rightward and downward. To put it another way, the SUBCON link seems to be a weaker form of the IS-A link: SUBCON can carry context-related markers, but not those used for property search; IS-A links can carry both types. This is a recent addition to the system, and I'm not sure I have it exactly right, but it is clear that something of this sort is needed. Are THING and GENERAL the same node? I don't know.

There is a lot more thinking to be done about all of this, especially about how far we can push the analogy between the IS-A hierarchy and the context hierarchy. Are there exclusive sets of contexts and context-clashes? Mandatory sub-contexts? Do the principles governing the growth and change of the context net resemble those governing the IS-A net? There must clearly be a context-exception mechanism to cancel the inheritance of a fact from a super-context when necessary. What are the details of this? All of these are questions for the future, and until these details are worked out, the context hierarchy will continue to be just an image, not a coherent theory.

17. Habitat and Recognition

In the previous section we saw how context mechanisms operate to control certain aspects of the reasoning and data-retrieval processes. Context plays quite a different function in recognition processes. Every context supplies us with certain expectations about the entities that inhabit it. These expectations can serve to limit *a priori* the set of possible descriptions against which an incoming stimulus may be matched. In most environments, only a very small fraction of the potential set of descriptions will be considered. Of course, this pre-selection is not absolute. When faced with a surprise, the system must loosen its expectations, or abandon them altogether and proceed to make the identification on the basis of sensory information alone. But in the normal course of its operation, the system will find that its expectations are confirmed in the vast majority of cases, and that they can save it a lot of work.

Such a pre-selection system is of obvious value in a recognition system that must compare descriptions one-by-one to the incoming sample, looking for a match. It is less obvious why pre-selection would be useful in a system like this, which can *simultaneously* weigh any number of descriptions against the incoming set of features. But this assumes that the sensory processors have already assembled a sufficient set of features for an unambiguous identification. It is far more likely that the first wave of information to arrive for an object will contain only a few crude features, and that to obtain more will require a specific request to the sensory system and the expenditure of more effort. Clearly, it will take more and better features--and thus more sensory effort--to identify an object from a very large set than from a very limited one. Of course, the requirements on the set of features will also be influenced by how specific we want the identification to be and how certain we want to be that the choice is right.

So the mechanism of context-driven pre-selection is what allows us to go through life giving only the merest glances to the majority of the things around us. To do otherwise would leave much less time for looking at the interesting or surprising things we encounter. A knee-high moving brown blur in a city park is a dog. A black compact 8-inch blob on a desk is a telephone. Large colorful moving blobs in the street are cars. And so on. We are free to take a closer look at any of these objects if we want to, but usually the gross features and the context are enough to satisfy us.

The resolution of ambiguity by context is even more important in the recognition and understanding of natural language. Consider the following two sentences:

"Babe Ruth went into the locker room to look for his favorite bat."

"Count Dracula went into the crypt to look for his favorite bat."

Now, it is pretty clear that Babe Ruth is seeking a piece of wood and Dracula a flying mammal, but what is the machinery that tells us this? We cannot look more closely at the word "bat" to determine which sort of bat it is, nor can we call up the author and ask. Clearly, the issue is decided by context: the first part of the Babe Ruth sentence places us in a BASEBALL context, which is inhabited by wooden bats, but not furry ones; in the VAMPIRE-STORY context, the reverse is true. In a neutral context, inhabited by neither flavor of bat, we would have to guess which meaning is intended or defer judgement until more information arrives. The process of deciding which context to invoke is at least as interesting as that context's subsequent use; we will see the details of this process later.

To see how this habitat mechanism works, it is best to consider the most obvious form of the problem: the habitats of animals. At first glance, it appears that habitat is just another property, like size or color. THE-EVERGLADES is the HABITAT of ALLIGATORS, and so on. If we are trying to identify an animal from a verbal description, we would use the statement that it lives in the Everglades just as we would use, say, the statement that it is green or has long pointy teeth: it is merely another fact to throw into the intersection-finding process. In this case, habitat is a particularly valuable feature, since it is relatively hard to tell alligators from crocodiles and caimans on physical grounds alone. (This assumes that we care to be this specific.)

Usually, of course, we do not have to be told that a creature inhabits the Everglades, or wherever. More commonly, we would assume the habitat property because we happen to be standing in the Everglades when we encounter the beast, or because its description appears in a novel whose setting is the Everglades. ("There, in the fetid water before me, was a malevolent green shape, at least eight feet long, with row upon row of saber-like teeth. I reached for my Bowie knife...") So, while we are physically or mentally operating within an Everglades context, we can mark all of the Everglades inhabitants and operate only within this set while looking for intersections of other features. This is the pre-selection I described earlier. We can leave this set of expectation markers in place until we are ready to move on to another context.

The concept of habitat can be extended to include much more than animals and their territories. Desks inhabit offices, oil wells inhabit Texas, unicorns inhabit mythology, mesons inhabit particle physics, biplanes inhabit World War I, the Plague inhabits Medieval Europe, and Julius Caesar inhabits the Roman Empire in the First Century B.C. An entity may inhabit many contexts: lions, for instance, inhabit both African grasslands and zoos. Though we have not yet seen the details of representing actions, it should be noted that these, too, have their habitats: jousting in the Middle Ages, blowing out candles at a birthday party, harvesting grain in September.

This view of habitat as merely another feature makes it easy to see how the system would handle surprises. In my previous paper, I described

how a recognition system might deal with one or two anomalous features in an otherwise routine identification by looking for an *excuse*. A cow might appear to be three-legged because one leg is hidden by another, or it might have stepped on a land mine. Even if we can't verify these excuses, the fact that such plausible explanations exist gives us some confidence in sticking to our judgement that the beast is a cow, as long as it is a good cow in all other respects. Of course, if there were a truly three-legged cow-like animal, we would probably never begin to doubt our observation of three-leggedness. It is only when the set of viable identifications vanishes that we begin to circle back, looking for input features that can profitably and plausibly be doubted. And since things can so easily be moved, habitat is one of the easiest features to doubt. An alligator in my office is still an alligator, as long as it is a good alligator in all other respects.

So let us assume that I am walking into my office and there, on the fetid floor before me, is a malevolent green shape, at least eight feet long, with row upon row of saber-like teeth. I begin to intersect these features with the already-selected set of inhabitants of my office--chairs, tables, plants, other people, computer terminal, paper, electronic junk, etc.--and find that the intersection is empty. Since I have quite a good view of the thing, and since surprises around MIT are not uncommon, I decide to throw out the expectation marker and to reintersect using only the sensory data. (This re-starting of the recognition process may be the source of what we call a double-take.) This time, a clear candidate emerges: an alligator or one of his close relatives. I reach for my Bowie knife...

Note that habitat can be used just like any other property to group entities together into classes. Thus, we have FARM-ANIMALS, OFFICE-FURNITURE, and RENAISSANCE-PAINTERS. Note, too, that a context can be created specifically to cover the habitats of some particular thing or group of things. Thus, we have grasslands, tsetse-fly country, and the Age of Sail. The process of creating such a context to contain a commonly-associated set of entities is closely analogous to the creation of an entity to represent a commonly-associated set of features.

This habitat system uses the same context hierarchy that we developed earlier for absolute contexts. To mark all of the inhabitants of some context--AFRICA, for instance--we would first mark all of that node's super-contexts and sub-contexts (in separate sweeps), then would mark the inhabitants of all of these contexts. Thus, in an AFRICA context, the recognition process would be dealing with inhabitants of AFRICA itself, of its various parts (ETHIOPIA, KENYA, THE-SAHARA, etc.), and of more inclusive contexts such as OLD-WORLD. Note that this process would include entities that inhabit the OLD-WORLD generally, but not those that inhabit non-African *parts* of the OLD-WORLD--to reach these contexts a marker would have to travel to the left and then to the right, which is not possible if the sweep is done properly. The recognition process would be primed to see lions, ostriches, and dromedary camels, but not tigers, rheas, or Bactrian

camels. A general context such as AFRICA doesn't limit the set of possible entities very strongly, but it will help to eliminate many of the potential confusions and ambiguities.

By choosing a context at an appropriate level of generality, we can focus the system's attention as finely or as diffusely as we desire. If the context is BOARD-GAMES, the system will easily interpret any reference to such items as Boardwalk, Park Place, pawn, or rook, but the term "king" is ambiguous: it could mean either a chess king or a checkers king. If we move into the more specific CHESS context, this ambiguity disappears, but now any reference to Boardwalk is a *non-sequitur* which can only be understood by popping up to a more general context or by abandoning the expectation mechanism altogether. It is not always clear which of the superior contexts one should move to in such a case: the reference could be to an actual physical boardwalk, after all.

So by moving from node to node in the context hierarchy, the system can activate whatever set of expectations is appropriate for a given situation, including what is needed, but excluding everything else to minimize the potential for ambiguity. This seems much smoother than the older view of switching between a few discrete mini-worlds that have little or nothing in common.

We are still faced with the problem of choosing which context to activate at a given time. Often, of course, the context is supplied. In natural language, for instance, we have such context-specifying phrases as "in natural language". At the start of a movie, the screen may light up with the words "Paris, 1789" or whatever. In real life we generally have a pretty good idea of where we are in space and time, unless we have been unconscious or lost. In a conversation, we may have what amounts to a push-down stack of the contexts we have passed through. If we are speaking of games, and then move either to the more specific CHESS context or to some completely unrelated interruption, we will probably keep a record of the earlier context; any local *non-sequitur* may be an attempt to pop back into the GAMES context.

But there remains a large set of cases in which the context is not supplied, but must be recognized: we join a conversation in the middle; the movie begins with pictures rather than titles; we walk into a strange room not knowing what sort of room to expect; we are shown a photograph with no caption; and so on. The "features" used in this context-recognition are the things we find in the context. Though I haven't yet worked out the details, the general idea is to find the most specific context node that contains all of the items mentioned. This can be done by marking to the left from the habitat of each item and then looking for the rightmost point at which all of these trees intersect.

Thus, a mention of base, diamond, pitcher, bat, and ball clearly invokes the BASEBALL context, though each of these words taken in isolation has other possible meanings. Base and acid invokes chemistry. Base and

collector invokes electronics. Pitcher, cup, and fork invokes table settings. Fork, bishop, and check invokes chess. Check, bill, and collector invokes finance. Bill and wing invokes birds. Wing and base invokes the Air Force. So it goes. The pieces of information hung in a context also have a role to play in context recognition: If Eisenhower is President, this must be the 1950's. I think that this process of intersecting inhabitants to select a context captures the essence of what Quillian was trying to get at, with perhaps slightly more precision: the use of random associations between certain items to create expectations that will be useful in resolving ambiguities.

There is an interesting circularity at work here: We are using the context to help us recognize the things in it, while using the things to select the context. Which comes first? The answer, I think, is that both emerge together, like the solutions to a set of simultaneous equations or the junction-labels in the work of David Waltz [19]. Consider the process of recognizing a photograph of unknown objects in an unknown setting, where the objects cannot be seen too clearly. We begin by attacking the clearest objects, trying to see what they are. Let's say that a cow emerges. This suggests that we are in some sort of farm context. This, in turn, helps us to recognize other objects in the picture: a pig, chickens, a tractor, two silos with loading machinery. The tractor and machinery help us to further refine the context: it is a *modern* farm with mixed livestock. Once again, this helps us to recognize objects: the thing on the ground near the cow is an electric milking machine. And so on.

The objects and the context, then, refine each other. The circularity is a converging one, once we get the first bit of the picture by brute force. I call this *two-finger recognition*. One finger (or set of fingers) starts at the most general object node of the system; the other starts at the leftmost context. Gradually, by alternating steps, the object fingers are moved down and the context finger to the right until we have a rather specific context inhabited by specific objects. Clearly, the simultaneous emergence of a face and its constituent parts is another example of the same process. Perhaps this is what the Gestalt psychologists were driving at.

One big question remains: Should entities be connected to habitats by special hardware HABITAT-links? If not, then what should we use? I'm not sure about this at present. My instincts rebel at the thought of creating a huge new network of HABITAT-links. Perhaps we can get away with using the identity links hung in a context to provide us with *implicit* habitat links: If EISENHOWER is PRESIDENT in the 1950's, then EISENHOWER inhabits the 1950's. He also inhabits World War II because he was a GENERAL there, and WEST-POINT of a certain era as a STUDENT. Inhabitants with no other role to play in a context can be simply INHABITANTS. It is clear that this approach would save a lot of links, but I don't yet know how far we can push it. For now, we had better play it safe and think in terms of explicit HABITAT links.

18. Relations and Frames

We have seen some ways of connecting properties to objects, and of hanging facts and entities in the appropriate contexts, but how do we handle the representation of relations like GREATER-THAN, in which neither of the objects being related is a property or a part of the other? And how do we handle complex frame-like descriptions which impose not only a set of requirements and properties upon the role-players, but also relations between them? How do we handle multiple descriptions? What does all of this have to do with contexts, and what changes does it suggest in the form of the basic OF-link? These are questions that I have only begun to explore, and this section will only contain half-baked ideas--not clear answers.

Let us begin with the GREATER-THAN relation. What do we want to say about it? It is a binary transitive relation with two "slots" to fill. Both slots must be filled by entities from the class MEASURABLE-QUALITIES; each slot must be filled exactly once in any particular instance of a GREATER-THAN relation. The first role has the name GREATER; the second, the name SMALLER. Beyond that there is not too much to say. This relation seems to be a primitive that must be defined by consistent use, instead of by other more primitive relations.

A possible network for GREATER-THAN is shown in figure 18. The structure above the dotted line is the static GREATER-THAN description; the structure below the line is a particular instantiation of this. As before, the double chevrons flag immediate connection (properties related to the role itself), and squiggly arrows are IN-links. Since GREATER and SMALLER are singular roles, the CPU program must insure that each is filled only once (a form of digestion when new links are added). Since these roles are required, the program is free to assume that the role is filled by something appropriate, even if the player is not explicitly stated in the network. (Every person has a liver, etc.) The creation of the A>B node is necessary because many pairs of items will be hung under GREATER-THAN, and it is essential to remember which GREATER is paired to a given SMALLER. Besides, we might want to say something about the particular relation itself--that it is an assumption, perhaps.

Are the squiggly IN-links shown here the same type that we used to represent "Eisenhower was President *in* the 1950's? I think so, but I'm not sure. The English comes out the same, but that is a dangerous path to follow. The uses are at least analogous: When we are in the context of the A>B relation, the link from A to GREATER is in some sense activated. But I have not played with this enough to be sure that I am not missing some essential distinction. For now, let's call them the same.

Though it is conceivable that we would want to reach into the static part of such a network to create specific exceptions, normally we will use it as a black box: we will only want to get at the terminal nodes representing the relation itself and the slots. In this case, we need to

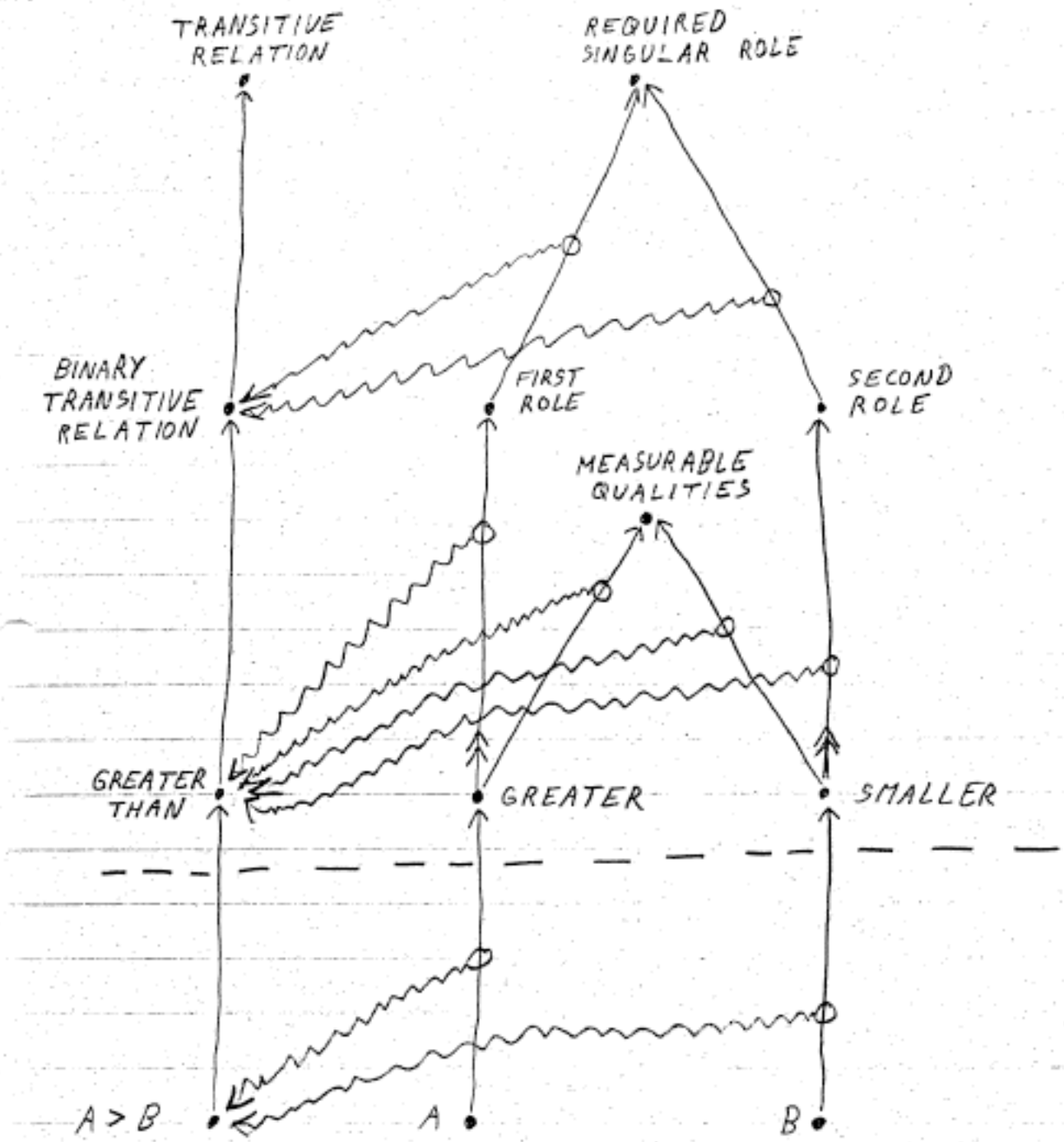


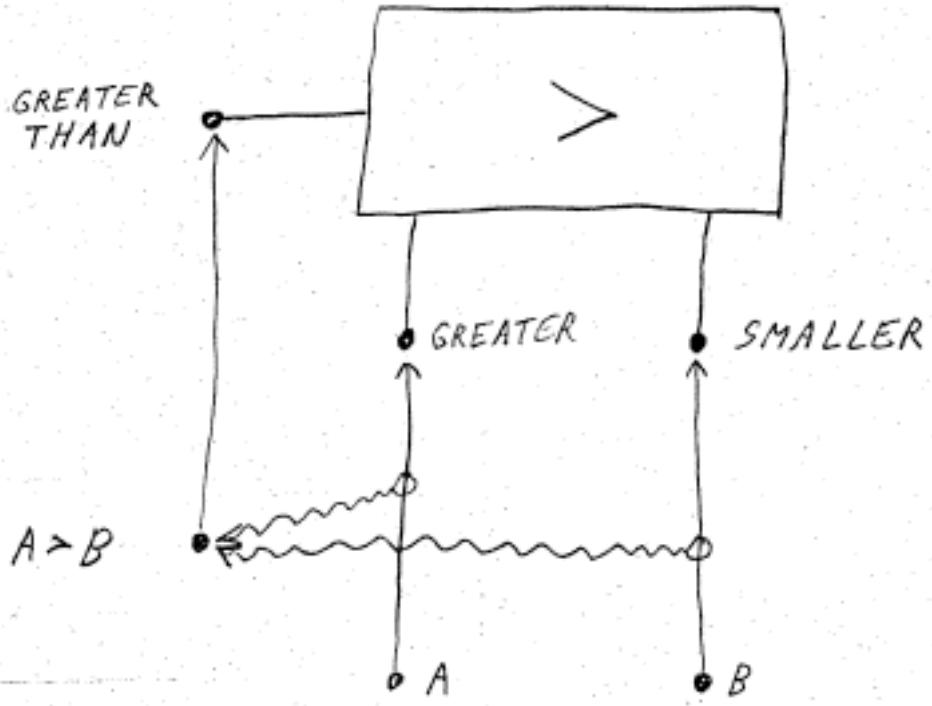
Figure 18 – Structure of the GREATER-THAN Relation.

make attachments to the GREATER-THAN, GREATER, and SMALLER nodes. This suggests that we can save a lot of clutter by drawing the whole GREATER-THAN network as a box, as shown in figure 19a. To avoid entanglements, we may have several GREATER-THAN boxes in a given diagram, but they all represent the *same* piece of network, not copies. Note that BINARY-TRANSITIVE-RELATION could itself be a black box with terminals FIRST-ROLE and SECOND-ROLE, hidden inside the GREATER-THAN box (and in many other boxes as well). If we want to get by with even less clutter, we could represent the whole instantiated box as a labeled arc, as in figure 19b, but this perhaps sweeps too much under the rug.

Now we can define SMALLER-THAN by merely hooking up two boxes (figure 20). We simply switch the roles of FIRST and SECOND and plug into the terminals of GREATER-THAN. SMALLER-THAN then becomes a box in its own right. ABOVE is defined as a binary relation between physical objects (PHYSOBS) with a GREATER-THAN relation, not between the players themselves, but between their VERTICAL-POSITION properties (figure 21). SUPPORT can be defined over two physobs, one above and touching the other (figure 22). Note that the TOUCH box has only a single slot that can accomodate multiple players; it is thus automatically reflexive.

It seems clear that with this sort of representation one can create rather complex descriptions, using pre-existing description boxes as subroutines. A FAMILY box can be created with mandatory slots for one adult male, one adult female, and many optional children of both sexes. Inside this box are smaller boxes for HUSBAND-WIFE, BROTHER-SISTER, PARENT-OFFSPRING, etc. An ARCH box would have slots for blocks and sub-boxes for the spatial relations that hold between them. A SYMPHONY-ORCHESTRA box would have one slot for a conductor, a few for horns, and a couple of dozen for violins, with much internal object description, but not many internal relations. An ELECTRONIC-NETWORK box is a set of CONNECTED-TO relations over the terminals of elements that may themselves be smaller electronic networks. As we will see, actions and events are represented in a form very similar to relations, so we can create structures like BIRTHDAY-PARTY with suitably restricted slots for the people and things involved, a set of internal relations, and some implied actions as well. Such structures, when small, are relations of the sort that we see in Winston-nets; when large, they look like the Minskian frames.

But there are a tremendous number of questions still to be answered about these structures. What are the precise rules of marker propagation for answering the various questions we might ask of these frames? What, if anything, is the difference in behavior between the IN-links used here and SUBCON links? Do we really need both? How do we indicate that the two bases of an arch *do not* touch? That the players in a GREATER-THAN relation may be any measurable quality, but that they must must be expressed in *the same units*? That distinct male children in a family are brothers, but that nobody is his own brother? Once expressed, how are these restrictions enforced? I don't know, but I'm working on it.



(a)

(b)

Figure 19 — Box (a) and Arc (b) Notations Equivalent to Figure 18.

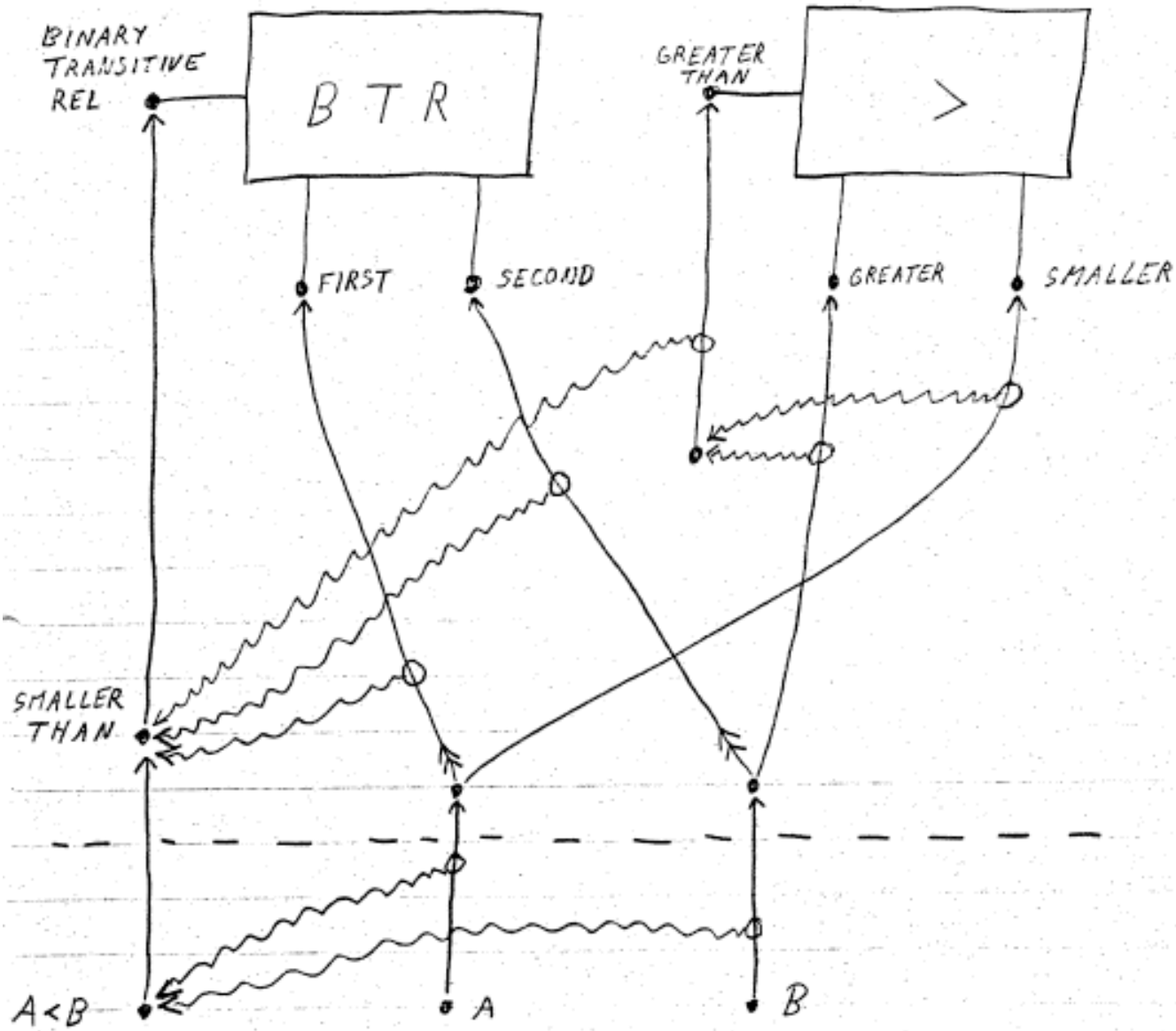


Figure 20 – Structure of the SMALLER-THAN Relation.

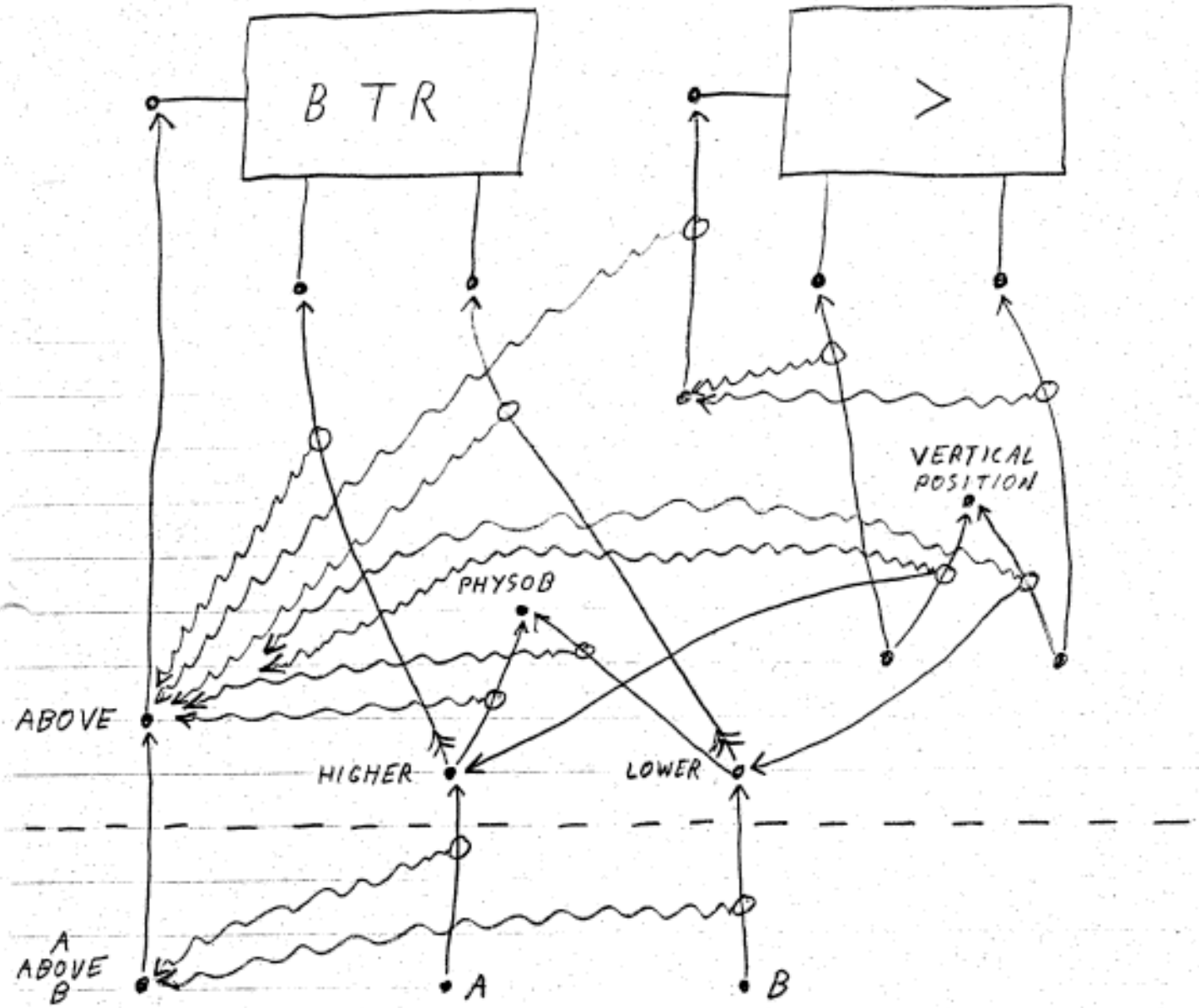


Figure 21 — Structure of the ABOVE Relation.

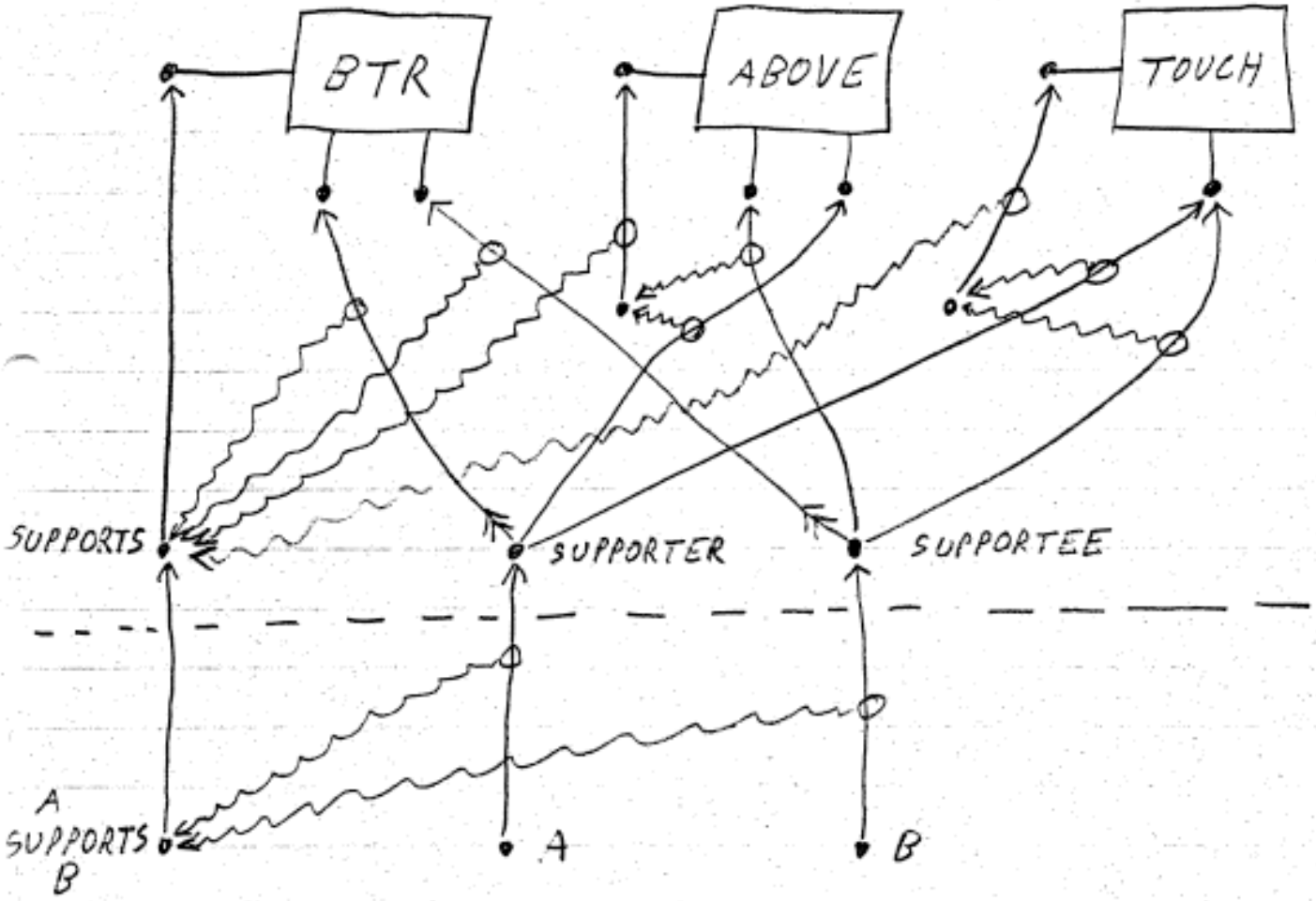


Figure 22 - Structure of the SUPPORTS Relation.

It is tempting to try to express everything in terms of these frame-structures of IS-A and IN-links, doing away with explicit OF-links altogether. ABRAHAM would be the father of ISAAC not because of a specific OF-connection between them, but because ABRAHAM plays the FATHER role in a relation in which ISAAC also plays a role (the SON). Something like the EMITTER of T1 would still be hung off to the right since it is *a part* of T1, but it would be hung only by a SUBCON link, not by an OF-link. To find the X of Y, we would mark some appropriate set of links--the *OF-universe* of Y--as active, and then pick up every node hung under X by one of these active links. The OF-universe would run rightward from Y and its superiors, and would also include the other role links in any relation or frame in which Y plays a part. This isn't exactly right yet, and small differences matter here, but it at least indicates the sort of line I am following at the moment.

The point of all this is not just to reduce the number of link-types by one. I mentioned earlier that the OF-links had certain problems. Chief among these is that, in order for the ladder sweep to work properly, the OF-link rungs must extend all the way down to the level of individual nodes. We cannot say that every American is the COUNTRYMAN-OF every other American; instead we must actually create COUNTRYMAN-OF links between each pair of individuals. We could, of course, have the digestion processes create these links as new Americans are added, but that is not the point. We would like to do without these links altogether. The creation of extra links for reciprocal pairs (HUSBAND-OF and WIFE-OF) is a less extreme form of the same problem. The hope is that membership in a common relation can take the place of explicit low-level OF-links between the various role-players in the relation.

So far, we have had no occasion to use the links themselves--or rather their handle nodes--as contexts, but such situations do arise. To see this, let's take another look at the nature of descriptions. The view I will present here is related to--though not identical to--the ideas expressed in the Merlin system of Moore and Newell [13].

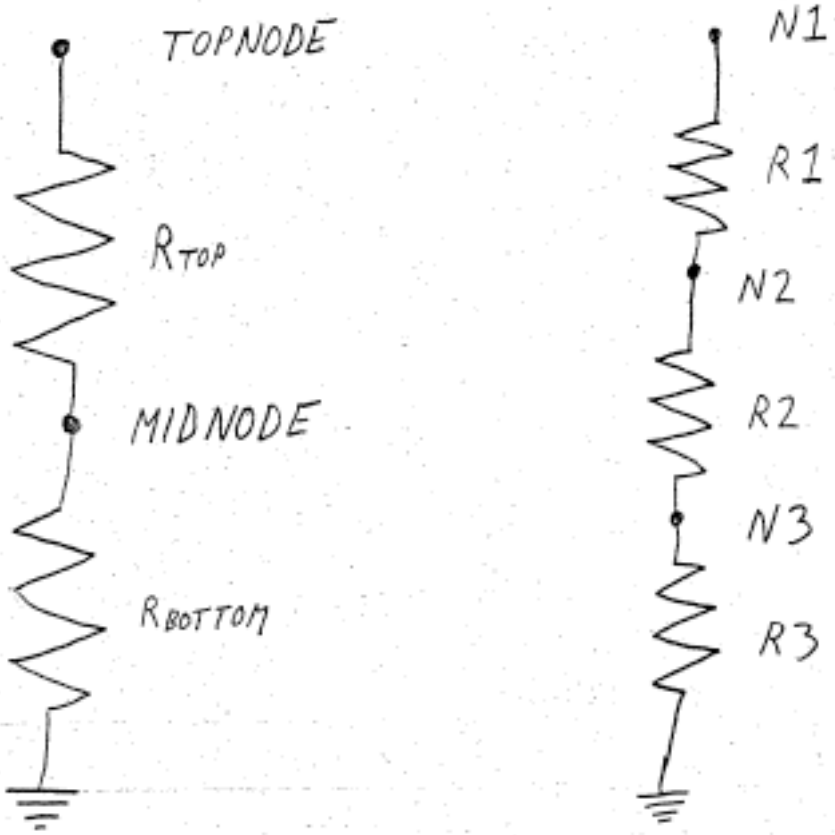
A description can be viewed as having three elements: an assignment statement, a set of modifications, and a set of sub-descriptions. The assignment is just an IS-A link; CLYDE IS-A ELEPHANT, or whatever. The modifications consist of whatever exceptions and additions we must make in the ELEPHANT description to make CLYDE fit: smaller ears, a broken tusk, green stripes, etc. The sub-descriptions consist of a mapping of CLYDE-parts into ELEPHANT-parts; H1 is the HEAD, L3 is the LEFT-HIND-LEG, etc. Note that each of these mappings can itself be a description, with its own set of modifications and sub-descriptions, recursively for as far as we care to go.

In the previous sections, these modifications and sub-descriptions have been hung to the right of the node being described (in this case, from CLYDE) by OF and IN-links, but a strong case can be made for considering these things as parts of *the description of CLYDE*, rather than of CLYDE

itself. This is pretty much a matter of taste as long as each entity has only one major description, or when multiple descriptions cover entirely different features of the thing being described or agree where they overlap. It becomes a matter of considerable importance, however, when the two views of an object are in conflict. Examples of this are the Necker cube; Minsky's example of the generator in a car, which is either an electrical network or a mechanical assembly; and the voltage divider of figure 23, which can be mapped into the standard voltage divider in either of two distinct ways. The point here is that we can use either description profitably, but it would create serious conflicts to have both active at once.

It is, of course, a simple matter to create separate contexts to hold the various inconsistent descriptions, and to activate only one at a time. But it is very tempting to use the handle node of the assignment link as the context node from which the description is hung, as shown in figure 24. This feels right, somehow, but I have had a lot of trouble in integrating this idea with the rest of the system. It is not yet clear whether it is worth the effort.

And that, alas, is the present state of the context system. I warned you that it wasn't going to hang together. It does, however, look like something that will crystallize nicely, given one or two more good ideas--or perhaps the elimination of one or two bad ones.



VD VD1

Figure 23- Standard voltage divider (VD) and circuit to be described (VD1).

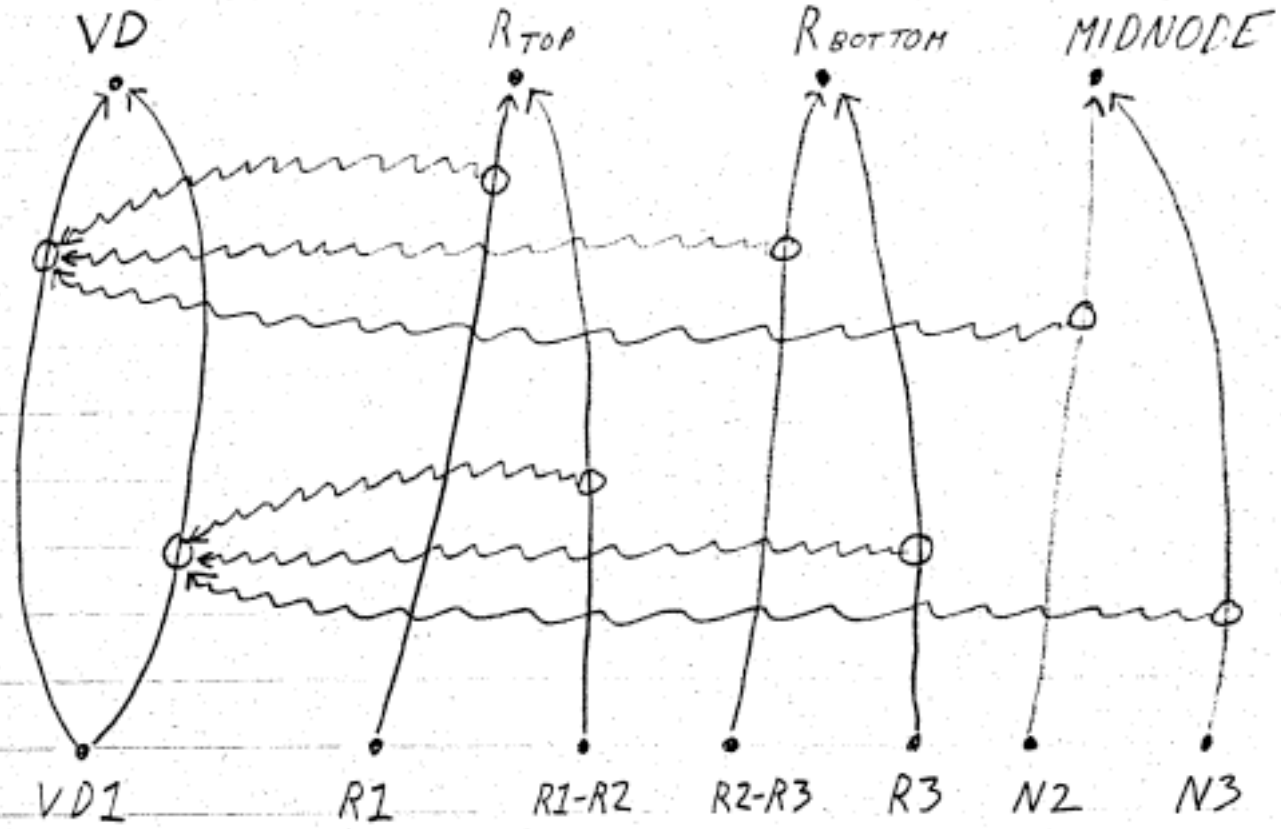


Figure 24 - Portion of the network describing VD1 as a VD in two different, incompatible ways.

19. Actions and Verbs

Now that we have developed the absolute context mechanism for representing multiple states of the world-model, and a mechanism for binding many facts and relations into a large frame-structure, we can at last turn our attention to the representation of events and actions. I include in this category both specific events--the assassination of John F. Kennedy, for instance--and event-types such as "murder" or "eating cucumbers"; the distinction is more or less analogous to that between CLYDE and ELEPHANT. In general, I will use the terms "event" and "action" interchangeably; the distinction is merely that an action-frame has a slot for the agent, while an event may not. I will use the term "verb" to refer to the word that names an event-type or action-type, though the relation is considerably more complex than the simple word-object pairs we saw earlier.

Stated briefly, an event frame is just a relation-frame of the type we saw in the last section, with the addition of a pair of internal context nodes representing, respectively, the state of the world before the event and the state of the world after the event. The MURDER frame, for instance, would have slots for the MURDERER and the VICTIM, both restricted to human beings. The MURDERER is the agent in the frame: there is a CAUSES relation between him and the node representing the action. All of the above restrictions reside in the MURDER frame itself. In the included BEFORE-context we have the statement that the VICTIM is alive; in the AFTER-context the VICTIM is dead. To represent the event of CAIN murdering ABEL, we just create an instance of the MURDER frame and plug CAIN and ABEL into the appropriate slots; this event then inherits the proper relations and properties.

Event-frames have a few other slots that are worthy of note: TIME, PLACE, and EXPANSION. Time and place are probably self-explanatory, but note that they are not always filled in, at least with any precise entry. Thus, the TIME slot for "Cain slew Abel" would simply be filled with PAST or perhaps BIBLICAL-TIMES, though the event (if it really occurred) was probably of very short duration. In "Rome fell", we would have to indicate an interval of several centuries--not a specific date, hour, minute and second. Places, too, may be very specific, very general, or not specified at all.

The EXPANSION slot is included in recognition of the fact that, while we are perfectly free to speak and think of events as instantaneous and indivisible, they in fact are made up of a sequence of sub-events. This sequence, if specified, is the EXPANSION of the event. The sequence of sub-events is tied together in such a way that the AFTER-context of one is the BEFORE-context of the next, and taken together they span the distance between the BEFORE and AFTER contexts of the parent event. Thus, we can say "Cain slew Abel" and leave it at that, or we can go on to add "by stabbing him" or "by drawing his sword, placing the tip against Abel's chest, and pushing it in." The expansion steps, of course, can have their own expansions, as far down as our knowledge and interest take us. Sooner

or later, we will have to treat some level of events as primitive, but an action that is left primitive in one frame may be expanded in another.

None of this is particularly revolutionary. The frame-structures look like case-frames or like Schank's networks [16]; the use of BEFORE and AFTER contexts is reminiscent of systems like GPS and STRIPS; the hierarchy of events and their expansions looks rather like the structure of BUILD-plans or the scenario representations of Bullwinkle [1]. But the present system does offer two clear advantages, not because of any unique structure, but because it is able to employ the representational machinery that we have already developed for static representation: first, the system is able to specify and enforce very precise restrictions on what can occupy the various slots; second, the system can support a complex hierarchy of event-types.

Let's consider the restriction issue first. Most case-frame systems use only the broadest categories to specify what can fill a given slot: animate object, place, physob, and so on. In this system, on the other hand, we can tie a slot into any category-node in the network--general or specific--and can enforce the restriction with the clash-detector. Thus, any two physobs can *collide*, but only flowers can *bloom*, only liquids can *trickle*, only herds of large herbivores can *stampede*. Of course, these restrictions cut both ways: if something is said to be trickling, then we are being told that it is a liquid.

Let me pause here to head off an obvious objection: Such restrictions are often deliberately violated, as when we say that a girl blooms into womanhood, or that sunlight trickles into the forest, or that a football team stampedes onto the field. These are metaphorical uses. It is intended that we will notice the clash and consider the link of girl to flower (or whatever) to be metaphorical rather than literal. Some properties--beauty and delicacy, perhaps--are inherited over this link, while others--photosynthesis and cellulose cell-walls--are not. Exactly how we decide whether a given property makes the jump is an interesting question, and one which I cannot yet answer. The relations and other things in the frame are also inherited over the metaphorical link, insofar as possible. It is so convenient to pick up a large package of relations in this way--even if they are slightly inappropriate--that such metaphors are used often. As the language evolves, a metaphorical usage can become a recognized second meaning of the verb with its own frame, and can sometimes even survive after the original meaning has dropped from use.

A related issue is the use of default assignments to fill slots. These are simply extra restrictions whose links bear a label stating that they may be cancelled with impunity, as we saw in Section 13. Thus the SHOOT frame has slots that are restricted by default to GUN and BULLET, but we are free to cancel these and substitute other pairs of objects: BOW and ARROW, SLINGSHOT and PEBBLE, etc. (This substitution is not entirely free, because the SHOOT-frame contains a non-cancellable internal relation that A LAUNCHES B, and the proposed slot-fillers must conform to this relation.)

Note that all of these comments about slot-restriction can apply to any relation-frame, not just to event-frames.

A far more important aspect of the event-frame system is its ability to stack event-types into an IS-A hierarchy. Thus, RUN and WALK are special cases of TRAVEL-BY-FOOT, which is a kind of TRAVEL, which is a kind of MOVE. All of the restrictions, properties, relations, and implications attached to the more general event-frames are inherited by the lower ones, and each of the lower frames adds its own restrictions as well. MOVE simply says that some physob is in one location BEFORE and another location AFTER. TRAVEL inherits this, adding that the MOVER is moving his own body. TRAVEL-BY-FOOT says how this is done (that is, it adds a restriction to the EXPANSION slot), and RUN adds that it is done at high speed. Implications, too, are inherited: if a certain radar is able to detect anything that *moves*, it certainly can detect anything that *runs*.

But the more specific actions can also have implications of their own: *Running* is very tiring; *walking* is less tiring. *Running* away from the scene of a crime is more suspicious than *walking* away. *Breaking bread* with someone is a friendly act; *breaking windows* at the American Embassy is hostile. *Eating cucumbers* gives you gas; *eating* in general does not. Note that not every action-frame has its own verb. We need to represent the generalized act of eating cucumbers so that we can state the above fact, but we do not need to coin a new word for this act. We do happen to have a word--cannibalism--for eating a member of your own species. So it goes. The mechanisms responsible for all of this inheritance are the same ones that allow CLYDE to inherit MAMMAL properties.

To understand how important such a hierarchy is, consider the problems Schank [16] has without it. He notices that such verbs as RUN and WALK have a great deal in common, and he feels (correctly, I think) that it is essential for the representational system to reflect these similarities. In the hierarchical system, we simply represent these two actions as instances of a more general action that embodies the common information; the actions are thus related, they inherit many of the same implications, but they have their own identities as well. Lacking such inheritance machinery, Schank's system must resort to rewriting every instance of running or walking as a MOVE (or, as he calls it, a PTRANS). This does indeed make evident the similarity between RUN and WALK, as well as the fact that the PTRANS implications apply, but RUN and WALK cease to exist as distinct conceptual entities: the differences in their meanings are translated into slight differences in the expansion-network that is attached to the PTRANS-instance when it is created.

Such an approach requires a great deal of memory to represent the fully-expanded network for each individual action, but that is the least of its problems. Schank himself points out that KISS, for instance, implies things that cannot be deduced from the simple fact that two people are touching their lips together. Since the action KISS does not appear in the internal representation, there are only two possibilities: to hang all of

the KISS-implications in the network representing each kiss as it is created, or to attach these implications to a complex pattern that can be matched against various networks to see if they are kisses. The former has the problem of all antecedent-driven systems: the question of how far to go. The latter has the problem that, in traditional systems, pattern-matching is a very tedious process for patterns of any complexity. For matching to work, the representation must be as canonical as possible--a given act must have only one possible representation. This need for canonical forms is, I think, what drives Schank to some of his more unpopular positions: the insistence that all actions be expanded out to the finest level of detail, and the requirement that the number of "primitive" verbs appearing in the internal representation be kept to an absolute minimun--fourteen or whatever the current number is.

I mentioned earlier that verbs are the words that we use to name various action-type frames, but that the mapping is more complex than the simple word-meaning pairs that we found in the world of objects. I think that the verb itself is represented by a frame of some complexity, with slots for such syntactic satellites of the verb as the subject, the direct and indirect objects, and other entities tied to the verb by such words as "at", "with", and "by". Whatever fills one of these slots in the verb-frame must fill the associated slot in the action-frame. The indirect object in the "GIVE" frame thus becomes the RECIPIENT in the GIVE action-frame, and the object of an AT-phrase is plugged into either the TIME or the PLACE slot. As the parser fills the slots, the clash-detector is hard at work rooting out absurdities and ruling on ambiguous cases, all with its usual high efficiency. The verb-frame would probably also indicate such things as the PAST-TENSE of the verb. Of course, verbs run in families and much of the verb-frame structure can be shared.

I have not worked out the details of this verb-meaning mapping, nor thought at all about the problems of parsing and language generation in this context. All that I am trying to do here is to create a well-behaved semantic representation that can handle the output of a language program and supply that program with a convenient but powerful set of meaning-constraints. The rest I leave to the linguists. I do have one weak conjecture, for what it may be worth: Perhaps the various standard sentence and phrase structures can be represented as sequences, and can play the same sort of parallel finite-state-acceptor game that I proposed earlier for speech recognition. This does not seem too far removed from Augmented Transition Networks [24], and the parallelism might help considerably.

Let me also emphasize that I do not claim--yet--to have a way of representing *everything* that can be expressed in English. Drew McDermott, for example, has pointed out that the before-and-after type of system has problems in dealing with actions in which the ongoing process is the essential part. Also, I have not yet faced up to the intricacies of such actions as PREVENT or CONSIDER, though the insights of Schank may be valuable here. And even when the machinery is fully developed, it will

require much construction of simpler action-frames before we can represent such ideas as EMBEZZLE or RATIONALIZE, whose descriptions must be based upon a rather extensive set of facts about corporate finance or psychology.

There are a few more loose ends to tie while we are in the general area of natural language. First, the problem of multiple languages. It seems clear that we want to use the context mechanism here. Thus, the name of a certain concept is "money" *in* English, "dinero" in Spanish, "argent" in French, "bread" in Early Hippie, "loot" in Hollywood Gangster, and "funds" in Bureaucratic. A sub-context hierarchy makes it possible for dialects to share most of their words--those in the super-context--but to differ in others. It is important to hang such words directly from the concept and not from the English word; had I known this in high school, I might now be fluent in Spanish. Of course, there will need to be a representation for each language's distinctive set of syntax patterns as well.

Another random point is the issue of modifiers. We have seen how to create an ALBINO-ELEPHANT node by cancelling the GRAY color link and adding WHITE. But how do we represent ALBINO itself? I had originally hoped to find a static declarative format for such things. In the original proposal I suggested the use of a *parasitic packet* which, when activated at the same time as another packet, would serve to modify that packet's meaning. This idea died along with the original packet system. It is now clear to me that a concept like ALBINO must be a procedure that tells how to create an ALBINO-X node, given an X from a suitably restricted class. Sometimes it will just be a matter of cancelling one link--a different link for each item to be albinoed--and adding a different one; sometimes it will be more complex, as when we apply BIG to an object. I have not worked out a representation for such procedures, but I suspect that they will look very much like action frames; the desired changes would appear as the difference between the BEFORE and AFTER contexts.

Which brings me to the last point. I mentioned that the tree of actions, their serial expansions, and the further expansions of these steps looked suspiciously like BUILD-plans [3]. Could this be a good way of storing procedural (or "how-to") knowledge in the machine? Each action that is represented in the system would come equipped with one or more suggested expansions explaining how to accomplish that action, along with perhaps some information about the comparative advantages of each expansion. Planning, then, would consist of expanding the top-level action all the way out, until a level of truly primitive actions is reached--impulses to muscles, perhaps. This expansion would be done in the exact context of the given problem--not in some generalized context with unspecified parameters--so that trouble could be spotted immediately. The BEFORE state of an action would serve as a statement of the action's prerequisite conditions.

Such plans would be easy to crawl around on: if an expansion doesn't pan out, simply pop up a level or two and try a different one. And

the system would be very good at finding the appropriate action to get from a current state to a goal state: simply do an intersection/recognition to find the set of actions whose BEFORE and AFTER states match the essential features of the problem. This use of recognition to find an appropriate operator should be much more flexible than the current practice of pattern-matching against the goal-statement. This, too, looks like a good thesis for someone to pursue.

20. Concluding Remarks

In this section I would like to address a few of the global questions that naturally arise concerning this system: Where do we go from here? Does this really have anything to do with human intelligence? And what does it all mean?

The first question is the easiest. The obvious next step is to try to resolve the remaining uncertainties about the context system and the exact format of the links. These issues are strongly interdependent, and I expect that they will all be resolved simultaneously--soon, I hope. I think that I finally understand the problems that must be solved, but I have not yet had time to look for a solution. I don't really believe that the glossed-over parts of the system could be hiding a flaw large enough to bring down the whole structure, but if there is one, it must be lurking in this last major area of confusion. Everyone, including myself, will feel more confidence in the system once the context mess is cleaned up.

After that, it will simply be a matter of tightening down all the screws. A series of medium-sized descriptions--electronic circuits, chunks of medical knowledge, simple scenarios, or whatever--will be worked out *in detail*, and a lexicon will be developed showing how various kinds of conceptual structures map into pieces of network. The procedures for data-access, digestion, recognition, and network reorganization will have to be made into precise recipes, if not into actual programs.

It seems unlikely that any of the above-mentioned examples will be so big that a computer simulation will be absolutely necessary, but I will probably bring one up anyway. It seems a simple enough task, and past experience suggests that it is generally easier to program something than to be continually explaining why such a program would be of no particular value. One area that will require a simulation, if it is to be studied properly, is the self-reorganization system: subtle, large-scale effects may be important here.

Hopefully, the steps described above should suffice to convert the bag of hand-waving ideas described here into a reasonably convincing thesis. If widespread skepticism persists, it may be necessary to develop some larger and more difficult examples--the animal-description recognizer or something on that scale. Such things will have to be done eventually, in any event. Beyond all of this lies the possibility of applying this knowledge-base as a tool in solving other problems: vision, speech, natural language, problem-solving, children's stories, perhaps even adults' stories. But such things are years in the future, at least.

As for the psychological reality of all this, I don't have too much to say. I have not really had a chance to sift through all of the reaction-time studies on class-inclusion and related issues, and I do not look forward to the task. Perhaps if I hesitate long enough, some interested psychologist will get there first. No doubt the available data

will be both profuse and contradictory, and new experiments will have to be done. Let me emphasize that I present this descriptive system merely a possible *kind* of model; even if we are on the right track, there will be many changes needed to fit the model to reality.

A form of evidence that is less compelling but--to me--more interesting, results from considering how well the model predicts the mistakes, ragged edges, and unintentional side-effects of the human knowledge-base. I include in this category such things as odd patterns of human forgetfulness (such as forgetting a word but remembering its first letter or what it rhymes with); the even odder stimuli that can cause a "forgotten" fact to come back; the mistakes that the disambiguation machinery makes, some of which are perceived as jokes (What's black and white and red/read all over?); and the associative mental phenomena that give poetic images their evocative power. These things, I feel, can serve as the optical illusions of the knowledge system. Any system that works must perform the essential tasks correctly--that's what it means to work--but when two systems make the same mistakes, that points strongly to analogous internal structures and procedures.

While I have been working on this system, I have from time to time encountered such human mental oddities, and have tried to explain these in terms of my networks. The results have been generally encouraging, but they are far too scattered for me to discuss at present. I hope that once the bugs are driven out of the context system, I will be able to study this sort of thing more systematically. It is a lot more fun than doing serious (number-crunching) psychology.

My system does have one glaring deficiency in this respect: For certain types of intersection tasks, the machine does better than people do. The machine, for instance, would have no trouble in finding an American city whose name is also the name of an animal (Buffalo), but many people do. And the machine, in theory, would never forget anything at all, except deliberately. Probably much of this can be explained away as the difference between imaginary perfect hardware and unreliable neurons, but the intersection problem appears to be more complex than this. I would very much like to develop a theory to explain why people find certain intersection tasks to be harder than others, but I am completely at a loss right now.

One argument in favor of the system is that it appears to be evolvable. By this, I mean that though the final system is rather complex, it can be reached by a series of easy steps, each of them a slight improvement over the previous state. The simplest type of IS-A net is not too different from a logic net that a very simple animal might use to combine a few simple inputs with a few bits of internal state to produce a given response. A marginally more complex organism might use such nets to store patterns of stimuli that correspond to prey, enemies, and so on; a simple intersection-finding program would then be needed. From this point the course is obvious: stimuli become complex features and properties; the

size of the network and the complexity of the "CPU programs" grow; and one by one the frills--exceptions, contexts, immediate connections, and so on--are introduced. In short, the system would not have to appear full-blown out of nowhere. It might, in fact, be interesting to look at the behavior of various "lower" animals to see whether they appear to be lacking, say, an exception mechanism.

What does it mean? It's hard to say at present, since there is still some small chance that the whole structure will come crashing down. But if it stays up, and if it can indeed perform the kinds of tasks that I have outlined here, then I think it will have demonstrated something very fundamental: That many of the problems that we in AI have been battling against in recent years are in fact artifices of our overall approach--of our dependence upon serial Von Neumann-type machines and the programming techniques that are appropriate to them. That when attacked with the proper tools--with tree sweeps and parallel intersections--these problems fall easily or never arise. That we do not, after all, have to resort to huge aggregations of programs and demons to deal effectively with simple declarative knowledge--assuming, of course, that we have the parallel hardware. I have no doubt that we will encounter a new set of problems--perhaps worse ones--farther on. But after such a long struggle at the same set of roadblocks, it is encouraging to find even a short stretch of open road.

21. Roots

Ideas, like living things, do not appear spontaneously in a vacuum; rather, they are created by the combination and evolution of pre-existing ideas. In this final section, I would like to point out my system's ancestors. I will limit myself to previous work which has had some direct influence upon my thinking; to list all related work would take volumes. I have deliberately placed this section last so that the relation of the older work to the new can be plainly seen.

The four major streams of thought which converge here are Quillian's network theory, Minsky's frames, Winston's learning-nets, and what might be called the PLANNER-CONNIVER tradition. The contributions of the first three are, I think, made clear in the preceding sections: From Quillian [14] comes the idea of using some sort of simple parallel hardware net to represent things and relations. From Minsky [12] comes the idea of the plug-in description with default values, and a clear vision of how such descriptions can be useful. From Winston [23] comes the view of descriptions as clusters of discrete relations--as opposed, say, to pictures--and the concept of the evolution of these clusters over time.

By the PLANNER-CONNIVER tradition, I mean not only the languages themselves [7, 17, 10], but also the various attempts to represent knowledge in them: Winograd's Blocks World [20], my own extensions of it in the BUILD program [3], Charniak's children's stories [2], and McDermott's monkeys [11], just to name a few. CONNIVER was created as a reaction to certain shortcomings of PLANNER, as perceived by its users [18]. Packets were an attempt to remedy certain weaknesses I had found in Conniver, especially in the area of demon-control. And, as we have seen, this network theory has arisen as a result of fatal deficiencies in the packet scheme. So the PLANNER-CONNIVER tradition was the most direct ancestor of the current work, but is the ancestor which it least resembles. I doubt very much whether the theory could have evolved without this chain of intermediate failure-driven steps.

I was finally brought face-to-face with the symbol mapping problem while considering some electronic networks that Sussman, McDermott, and Allen Brown were trying to represent by more conventional means. These simply would not fit into packets, and they forced me to go off in search of better ideas. Sussman and Brown, in the meantime, developed a network representation called AEDES which had much in common with my early nets, but without the special hardware. For a while, the two representations were isomorphic, and AEDES had much to do with the early development of the OF-link.

Certain key ideas came from other sources: My appreciation of the need for *tangled* IS-A hierarchies come from Winograd [20, 21, 22], and I believe that he caught it from Halliday. My notion of description and sub-description has been influenced by Moore and Newell's MERIN system [13]. The OWL system of Bill Martin and his group at M.I.T. [6] and the medical-

diagnosis system of Andee Rubin [15] have provided numerous thought-provoking examples and details. Almost everyone at the MIT AI-Lab seems to be working on some sort of frame-like knowledge representation or an application using frame-like ideas, and all of these systems are different. There has, of course, been some cross-pollination of ideas among these diverse approaches. Of these, the work of Ben Kuipers [8], Mitch Marcus [9], Candy Bullwinkle [1], and Rick Grossman [5] probably have the most in common with my own research, though there are fundamental differences as well. Drew McDermott and Bob Moore have supplied much excellent and useful criticism.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Gerald Sussman, Patrick Winston, and Penny Fahlman. Without their support at critical times, this paper could not have been completed. I would also like to thank Penny for her long hours of typing. During much of the period covered by this paper I have enjoyed the benefits of an I.B.M. Graduate Fellowship.

BIBLIOGRAPHY

1. Bullwinkle, Candace L., "Computer Performance of the Sentence Completion Task", Master's Thesis, University of Pittsburgh, Department of Computer Science, 1974.

2. Charniak, Eugene, "Toward a Model of Children's Story Comprehension", M.I.T. Artificial Intelligence Laboratory TR-266, 1972.

3. Fahlman, Scott E., "A Planning System for Robot Construction Tasks", M.I.T. Artificial Intelligence Laboratory TR-283. Also in *Artificial Intelligence 5*, 1-49, 1973.

4. Fahlman, Scott E., "A Hypothesis-Frame System for Recognition Problems", M.I.T. Artificial Intelligence Laboratory Working Paper 57, 1974.

5. Grossman, Richard W., "Representing the Semantics of Natural Language as Constraint Expressions", M.I.T. Artificial Intelligence Laboratory Working Paper 87, 1975.

6. Hawkinson, Lowell B., "The Representation of Concepts in OWL", forthcoming memo, MIT Project MAC, 1975.

7. Hewitt, Carl, "PLANNER", M.I.T. Artificial Intelligence Laboratory TR-258, 1972.

8. Kuipers, Benjamin J., "A Frame for Frames: Representing Knowledge for Recognition", in Bobrow and Collins, *Representation and Understanding*, Academic Press, 1975.

9. Marcus, Mitchell, "Wait-and-See Strategies for Parsing Natural Language", M.I.T. Artificial Intelligence Laboratory Working Paper 75, 1974.

10. McDermott, Drew V., and Gerald J. Sussman, "The CONNIVER Reference Manual", M.I.T. Artificial Intelligence Laboratory MEMO 259, 1972.

11. McDermott, Drew V., "Assimilation of New Information by a Natural-Language Understanding System", M.I.T. Artificial Intelligence Laboratory TR-291. 1974.

12. Minsky, Marvin, "A Framework for Representing Knowledge", M.I.T. Artificial Intelligence Laboratory MEMO 306, 1974. Also in Winston (ed.), *The Psychology of Computer Vision*, McGraw Hill, 1975.

13. Moore, J. and Allen Newell, "How Can MERLIN Understand?", in Gregg, *Knowledge and Cognition*, Lawrence Erlbaum Associates, 1973.

14. Quillian, M. Ross, "Semantic Memory" in Minsky, *Semantic Information Processing*, MIT Press, 1968.

15. Rubin, Ann D., "Hypothesis Formation and Evaluation in Medical Diagnosis", M.I.T. Artificial Intelligence Laboratory TR-316, 1975.

16. Schank, Roger C., "The Fourteen Primitive Actions and Their Inferences", Stanford Artificial Intelligence Laboratory AIM-183, 1973.

17. Sussman, Gerald J., T. Winograd, and E. Charniak, "MICRO-PLANNER Reference Manual", M.I.T. Artificial Intelligence Laboratory MEMO 259, 1971.

18. Sussman, Gerald J. and Drew V. McDermott, "From PLANNER to CONNIVER-a Genetic Approach", in *Proceedings of the Fall Joint Computer Conference*, 1972, AFIPS Press, 1972.

19. Waltz, David L., "Generating Semantic Descriptions for Scenes with Shadows", M.I.T. Artificial Intelligence Laboratory TR-271, 1972. Also in Winston (ed.), *The Psychology of Computer Vision*, McGraw Hill, 1975.

20. Winograd, Terry, *Understanding Natural Language*, Academic Press, 1972.

21. Winograd, Terry, "Frame Representations and the Declarative/Procedural Controversy", to appear in *Essays in Memory of Jamie Carbonell*, 1974.

22. Winograd, Terry "Five Lectures on Artificial Intelligence", Stanford Artificial Intelligence Laboratory AIM-246, 1974.

23. Winston, Patrick H. "Learning Structural Descriptions from Examples", M.I.T. Artificial Intelligence Laboratory TR-231, 1970. Also in Winston (ed.), *The Psychology of Computer Vision*, McGraw Hill, 1975.

24. Woods, William, "The Lunar Sciences Natural Language Information System, BBN Report No. 2378, 1972.