MIT Artificial Intelligence
Memo No. 252

January 1, 1972

ARTIFICIAL INTELLIGENCE
PROGRESS REPORT

by Marvin Minsky and Seymour Papert

Research at the Laboratory in vision, language, and other problems of intelligence.

This report is an attempt to combine a technical progress report with an exposition of our point of view about certain problems in the Theory of Intelligence

It is identical with pp. 129-224 of the 1971 Project MAC Progress Report VIII.

This work was conducted at the Artificial Intelligence Laboratory, a Massachusetts Institute of Technology research program supported in part by the Advanced Research Projects Agency of the Department of Defense and monitored by the Office of Naval Research under Contract number N00014-70-A-0362-0002 and in part by the National Science Foundation under grant number GJ-1049.

The M.I.T. Artificial Intelligence Laboratory

The A. I. Laboratory is concerned with understanding the principles of Intelligence. Its goal is to develop a systematic approach to the areas that could be called Artificial Intelligence, Natural Intelligence, and Theory of Computation. Here are its main current foci of attention.

ARTIFICIAL INTELLIGENCE

Robotics; Vision, mechanical manipulation, advanced automation. Models for learning, induction, analogy. Schemata for organizing bodies of knowledge. Development of "heterarchical" program control structures.

NATURAL INTELLIGENCE

Models of structures involved in "commonsense thinking". Understanding meanings, especially in natural language narrative. A new educational methodology, based on development of the child's abilities to describe processes.

THEORY

Computational trade-offs between time, memory size, and processor parallelism. Study of computational geometry as a tool for comparing different structures and strategies. Theory of schemata, for analysis of complexities of certain algorithms and languages.

These subjects are all closely related. The natural language project is intertwined with the commonsense meaning and reasoning study, in turn essential to the other areas, including machine vision. Our main experimental subject worlds, the "blocks world" robotics environment and the children's story environment, are better suited to these studies than are the puzzle, game, and theorem-proving environments that became traditional in the early years of artificial intelligence research. Our evolution of theories of intelligence has become closely bound to the study of development of intelligence in children. The educational methodology project is symbiotic with the other studies, both in refining older theories and in stimulating new ones; we hope this project will develop into a center like that of Piaget in Geneva.

As it has crystallized over the past few years, the main elements of our viewpoint can be summarized cryptically:

> Thinking is based on the use of SYMBOLIC DESCRIPTIONS and description-manipulating processes to represent a variety of kinds of KNOWLEDGE -- about facts, about processes, about problem-solving, and about computation itself, in ways that are subject to HETERARCHICAL CONTROL STRUCTURES -- systems in which control of the problem-solving programs is affected by heuristics that depend on the meanings of events.

The ability to solve new problems ultimately requires the intelligent agent to conceive, debug, and execute new procedures. Such an agent must know to a greater or lesser extent how to plan, produce, test, modify, and adapt procedures; in short it must know a lot about computational processes. We are not saying that an intelligent machine, or person, must have such knowledge available at the level of overt statements or consciousness, but we maintain that the equivalent of such knowledge must be represented in an effective way somewhere in the system.

This report illustrates how these ideas can be embodied into effective approaches to many problems, into shaping new tools for research, and into new theories we believe important for Computer Science in general as well as for Robotics, Semantics, and Education.

> Much of the material in this report is also part of a draft of a book on Thinking. For information about subsequent drafts and publication write to the authors at the A. I. Laboratory.

The Laboratory is seeking young workers who believe they can do work of the quality described herein, as staff, graduate students, or post-doctoral fellows.

1.0 Vision and Description

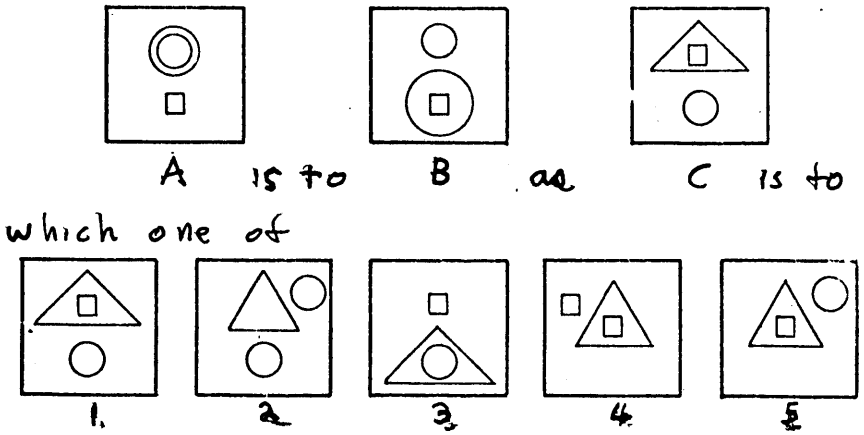
When we enter a room, we feel we see the entire scene. Actually, at each moment most of it is out of focus, and doubly imaged; our peripheral vision is weak in detail and color; one sees nothing in his blind spot; and there are many things in the scene we have not understood. It takes a long time to find all the hidden animals in a child's puzzle picture, yet one feels from the first moment that he sees everything. People can tell us very little about how the visual system works, or what is really "seen". One explanation might be that visual processes are so fast, automatic, and efficient that there is no place for introspective methods to operate effectively. We think the problem is deeper. In general, and not just in regard to vision, people are not good at describing mental processes; even when their descriptions seem eloquent, they rarely agree either with one another or with objective performances. The ability to analyse one's own mental processes, evidently, does not arise spontaneously or reliably; instead, suitable concepts for this must be developed or learned, through processes similar to development of scientific theories.

Most of this report presents ideas about the use of descriptions in mental processes. These ideas suggest new ways to think about thinking in general, and about imagery and vision in particular. Furthermore, these ideas pass a fundamental test that rejects many traditional notions in psychology and philosophy; If a theory of Vision is to be taken seriously, one should be able to use it to make a Seeing Machine!

1.1 Reasoning by Analogy

To emphasize that we really mean "seeing" in the normal human sense, we shall begin by showing how a computer program -- or a person -- might go about solving a problem of "reasoning by analogy". This might seem far removed from questions about ordinary "sensory perception". But as our thesis develops, it will become clear that there is little merit in trying to distinguish "sensation" or "perception" as separate and different from other aspects of thought and knowledge.

When we give an "educated person this kind of problem from an IQ test, he usually chooses the answer "figure 3":



People do not usually consider such puzzles to be problems about "vision." But neither do they regard them as simply matters of "logic". They feel that other, very different mental activities must be involved. Many people find it hard to imagine how a computer program could solve this sort of problem. Such reservations stem from feelings we all share; that choosing an answer to such a question must come from an intuitive comprehension of shapes and geometric relations, rather than from the mechanical use of some rigid, formal rules.

However, there is a way to convert the analogy problem to a much less mysterious kind of problem. To find the secret, one has merely to ask any child to justify his choice of Figure 3. The answer will usually be something like this!

"You go from A to B by moving the big circle down.
You go from C to 3 in the same way by moving the big triangle."

On the surface this says little more than that something common was found in some transformations relating A with B AND C with 3. As a basis for a theory of the child's behavior it has at least three deficiencies:

It does not say how the common structure was discovered.

It appears to beg the question by relying on the listener to understand that the two sentences describe rules that are identical in essence although they differ in details.

It passes in silence over the possibility of many other such statements (some choosing different proposed answers). For example, the child might just as well have said:

"You go from A TO B by putting the circle around the square..."

or

"You go from A to B by moving the big figure down," etc.

Aha! If that last statement were applied also to C and 3 the rules would in fact be identical! This leads us to suggest a procedure for a computer and also a "mini-theory" for the child:

Step 1. Make up a description DA for Figure A and a description DC for C.

Step 2. Change DA so that it now describes FIGURE B.

Step 3. Make up a description D for the way that DA was changed in step 2.

Step 4. Use D TO CHANGE DC If the resulting description describes one of the answer choices much better than any of the others, we have our answer. Otherwise start over, but next time use different descriptions for DA, DC and (perhaps) for D.

Notice that Step 3 asks for a description at a higher level! The descriptions in Steps 1 and 2 describe pictures, e.g. "There is a square below a circle". The description in Step 3 describes changes in descriptions, e.g., "the things around the upper figure in DA is around the lower figure in DB." Our thesis is that one needs both of these kinds of description-handling mechanisms to solve even simple problems of vision. And once we have such mechanisms, we can easily solve not only harder visual problems but we can adapt them to use in other kinds of intellectual problems as well -- for learning, for language, and even for kinesthetic coordination.

This schematic plan was the main idea behind a computer program written in 1964 by T. G. Evans. Its performance on "standard" geometric analogy tests was comparable to that of fifteen-year old children! This came as a great surprise to many people, who had assumed that any such "mini-theory" would be so extreme an oversimplification that no such scheme could approach the complexity of human performance. But experiment does not bear out this impression. To be sure, Evans' program could handle only a certain kind of problem, and it does not become better at it with experience. Certainly, we cannot propose it as a complete model of "general intelligence." Nonetheless, analogical thinking is a vital component of thinking, hence having this theory [Evans, 1964], or some equivalent, is a necessary and important step.

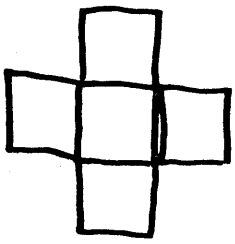
In developing our simple schematic outline into a concrete and complete computer program, one has to fill in a great deal of detail: one must decide on ways to describe the pictures, ways to change descriptions, and ways to describe those changes. One also has to define a policy for deciding when one description "fits much better" than another. One might fear that the possible variety of plausible descriptions is simply too huge to deal with; how can we decide which primitive terms and relations should be used? This is not really a serious problem. Try, yourself, to make a great many descriptions of the relation between A and B that might be plausible (given the limited resources of a child) and you will see that it is hard to get beyond simple combinations of a few phrases like "inside of", "left of," "bigger than," "mirror-image of," and so on.

But let us postpone details of how this might be done [see Evans, 1964] and continue to develop our central thesis: by operating on descriptions (instead of on the things themselves), we can bring many problems that seem at first impossibly non-mechanical into the domain of ordinary computational processes.

> What do we mean by "description"? We do not mean to suggest that our descriptions must be made of strings of ordinary-language words (although they might be). The simplest kind of description is a structure in which some features of a situation are represented by single ("primitive") symbols, and relations between those features are represented by other symbols -- or by other features of the way the description is put together. Thus the description is itself a MODEL -- not merely a name -- in which some features and relations of an object or situation are represented explicitly, some implicitly, and some not at all. Detailed examples are presented in 4.3 for pictures, and in 5.5 for verbal descriptions of physical situations. In 5.6 there are some descriptions which resemble computer programs. If we were to elaborate our thesis in full detail we would put much more emphasis on procedural (program-like) descriptions because we believe that these are the most useful and versatile in mental processes.

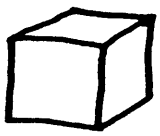
1.2 Children's Use of Descriptions

The theory of analogy we have just proposed might seem both too simpleminded and too abstract to be plausible as a theory of how humans make analogies. But there is other evidence for the idea that mental visual images are descriptive rather than iconic. Paradoxically, it seems that even young children (who might be expected to be less abstract or formal than adults) use highly schematic descriptions to represent geometric information.



We asked a little boy of 5 years to draw a cube. This is what he drew. "Very good," we said, and asked: "How many sides has a cube?" "Four, of course," he said.

"Of course," we agreed, recognizing that he had understood the ordinary meaning of "side," as of a box, rather than the mathematical sense in which top and bottom have no special status. "How many boards to make a whole cube, then?" "Six," he said, after some thought. We asked how many he had drawn. "Five." "Why?" "Oh, you can't see the other one!"



Then we drew our own conventional "isometric" representation of a cube. We asked his opinion of it. "It's no good." "Why not?" "Cubes aren't slanted!"

Let us try to appreciate his side of the argument by considering the relative merits of his "construction-paper" cube against the perspective drawing that adults usually prefer. We conjecture that, in his mind, the central sqare face of the child's drawing, and the four vertexes around it, are supposed in some sense to be "typical" of all the faces of the cube. Let us list some of the properties of a real three-dimensional cube:

Each face is a square.
Each face meets four others.
All plane angles are right angles.
Each vertex meets 3 faces.
Opposite edges on faces are parallel.
All trihedral angles are right angles, etc.

Now, how well are these properties realized in the child's picture?

Each face is a square.
The "typical" face meets four others!
All angles are right!
Each typical vertex meets 3 faces.
Opposite face edges are parallel!
There are 3 right angles at each vertex!

But in the grown-up's pseudo-perspective picture we find that:

Only the "typical' face is square.
Each face meets only two others.
Most angles are not right.
One trihedral angle is represented correctly in its topology, but only one of its angles is right.
Opposite edges are parallel but only in "isometric," not in true perspective.

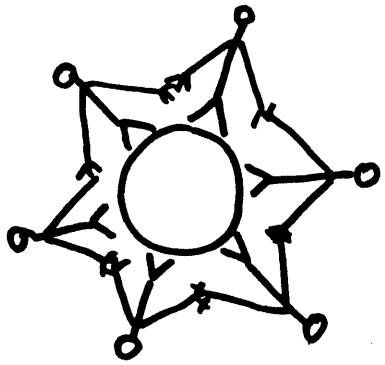
And so on. In the balance, one has to agree that the geometric properties of the cube are better depicted in the child's drawing than in the adult's! Or, perhaps, one should say that the properties depicted symbolically in the child's drawing are more directly useful, without the intervention of a great deal more knowledge.

One could argue that in the adult's drawing, the square face and the central vertex are understood to be "typical." We gave him the benefit of the doubt. Also, one never sees more than 3 sides of a cube, but childen don't seem to know this, or feel that it is important. The parallelisms and the general "four-ness" surely dominate.

Incidentally, we do not mean to suggest that our child had in his mind anything like the graphical image of his drawing, but rather that he has a structural network of properties, features, and relations of aspects of the cube, and that what he drew matches this structure better than does the adult's more iconic picture. In 4.4 we will show how such structural networks can be used a program that learns new concepts as a result of experience.

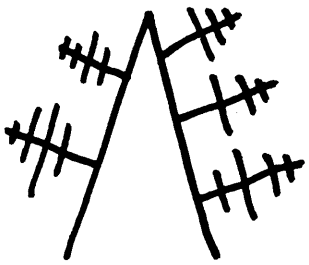
Not all children will draw a cube just this way. They usually draw some arrangement of squares, however, and this sort of representation is typical of children's drawings, which really are not "pictures" at all, but attempts to set down graphically what they feel are the important relations between things and their parts.

Thus "a ring of children holding hands around the pond" is drawn like this, perhaps because the correct perspective view would put some of the children in the water.



Also, in the child's drawing the people are all at right angles to the ground, as they should be!

For the same reason, perhaps, "Trees on a mountain" is drawn this way because trees usually grow straight out of the ground. It doesn't matter if an actual scene is right in front of the child; he will still draw the trees sideways!



A person is often drawn this way, perhaps partly because the body that is so important to the adult doesn't really do much for the child except get in his way, partly because it does not have an easily-described shape.

From all this we are led to a new view of what children's drawings mean. The child is not trying to draw "the thing itself" -- he is trying to make a drawing whose description is close to his description of that thing -- or, perhaps, is constructed in accord with that description. Thus the drawing problem and the analogy problem are related.

We hope no reader will be offended by the schematic simplicity of our discussion of "typical children's drawings". Certainly we are focusing on some common phenomena, and neglecting the fantastic variety and plasticity of what children do and learn. Yet even in that plasticity we see the dominance of symbolic description over iconic imitation.

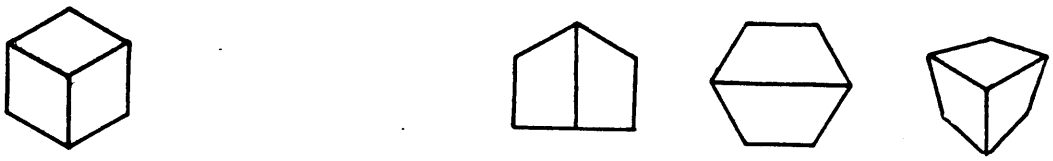
Most children before 5 or 6 years old draw people like this. Find such a child and ask him, "Where is his hair?" and draw some, or say "Why doesn't his nose stick out?" and draw an angular line in the middle of the face.

Chances are that if the child pays any attention at all and likes your idea, these features will appear in every face he draws for the next few months. The hair is obviously symbolic. The new nose is no better, optically, than the old, but the child is delighted to learn a symbolism to depict protrusion.

There is a vast literature describing phenomena and theories of "learning" in terms of the gradual modification of behavior (or behavioral "dispositions") over long sequences of repetition and tedious "schedules" of reward, deprivation and punishment. There is only a minute amount of attention to the kind of "one-trial" experience in which you tell a child something, or in which he asks you what some word means. If you tell a child, just once, that the elephants in Brazil have two trunks, and meet him again a year later, he may tell you indignantly that they do not.

The success of Evans' program for solving analogy problems does not prove anything, in a strict sense, about the mechanisms of human intelligence. But such programs certainly do provide the simplest (indeed, today the only) models of this kind of thinking that work well enough to justify serious study.

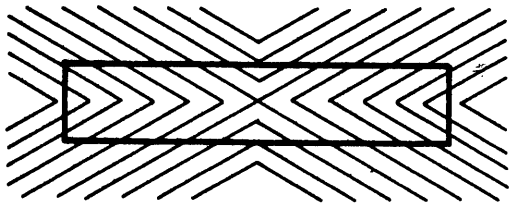
It is natural to ask whether human brains "really" use symbolic descriptions or, instead, manage somehow to work more "directly" with something closer to the original optical image. It would be hard to design any direct experiment to decide such a question in view of today's limited understanding of how brains work. Nevertheless, the formalistic tendencies shown in the children's drawings point clearly toward the symbolic side. The phenomena in the drawings suggest that they are based on a rather small variety of elementary object-symbols, positioned in accord with a few kinds of relations involving those symbols, perhaps taken only one or two at a time. These phenomena are not seen so clearly in the pictures of sophisticated artists, but even so we think the difference is only a matter of degree. While it is possible to train oneself to draw with quantitative accuracy some aspects of the "true" visual image, the very difficulty of learning this is itself an indicator that the symbolic mode is the more normal manner of performance. Even sophisticated adults often show a preference for unreal but tidy "isometric" drawings over more "realistic" perspective drawings:



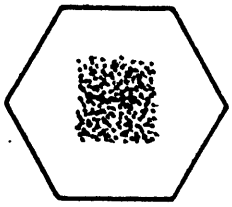
even though a cube is never seen exactly as in (1). In any case, all this suggests that "graphic" visual mechanisms become operative later (if at all) in human intellectual development than do methods based on structural descriptions. This conclusion seems surprising because in our culture we are prone to think of symbolic description as advanced, abstract, and intellectual, hence characteristic of more advanced stages of maturation.

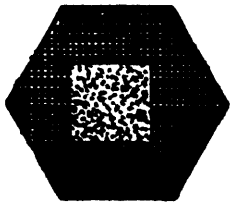
2.1 Appearance and Illusion

Now consider some phenomena that might seem to be more visual, less intellectual. These two figures show the same rectangle.

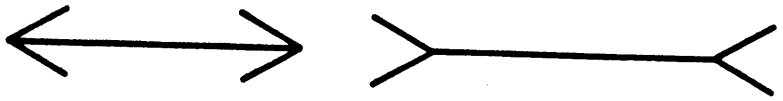


But on the right, the diagonal stripes affect its appearance so that (to most people) the sides appear to lean out and no longer seem perfectly parallel. Such phenomena have been studied with great intensity by psychologists. In the next two figures, the central squares actually have the same grey color, but everyone sees the one at the left as darker.

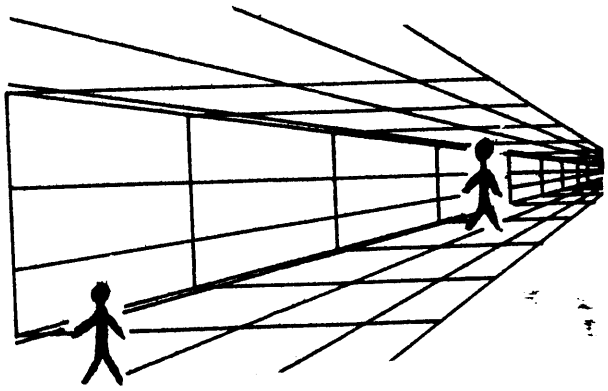




A good deal is known about the effects of nearby figures or backgrounds on another figure. Perhaps most familiar is the phenomenon in which the directions of the oblique segments make the horizontal line in the left figure to be shorter than that in the right figure.

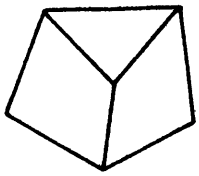


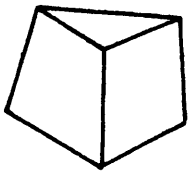
But the strangest illusion of all is this: to many psychologists these phenomena of small perceptual distortions have come to seem more important than the question of why we see the figures at all, as "rectangle," or "square," or as "double-headed arrow!" Surely this problem of how we analyze scenes familiar objects is a more central issue.



Thus one finds much more discussion why the smaller figure looks larger in pictures like this than about why one sees the figures as people at all.

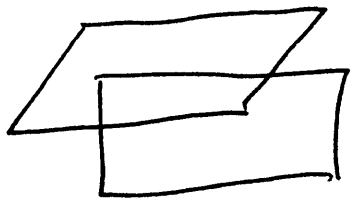
We agree that the study of distortions, ambiguities, and other "illusions" can give valuable clues about visual and other mechanisms. To resolve two or more competing theories of vision, such evidence might become particularly useful. First, however, we need to develop at least one satisfactory theory of how "normal" visual problems might be handled, particularly scenes that are complicated but not especially pathological.





Let us look at a few more visual phenomena. Both of these figures appear at first sight to be reasonable pictures of pyramid-bases -- that is, of simple flat-surfaced five-faced bodies that could be pyramids with their tops cut off. But, in fact, figure B cannot be a picture of such a body. For its three ascending edges (if extended) would not meet at a single point, whereas those of figure A do form a vertex for a pyramid.

So here we have a sort of negative illusion; figure B would not "match" a real photograph of any pyramid-base. However, it could match quite well an abstract description of a pyramid base -- say, one that describes how its faces and edges fit together (qualitatively, but not quantitatively).

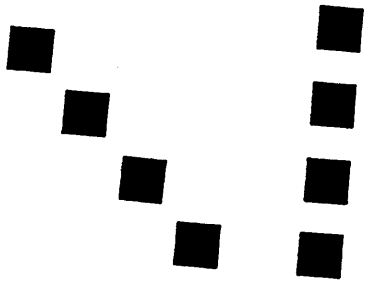


Another topic concerns "camouflaged" figures. The figure "4" embedded in this drawing is not normally seen as such because, we presume, one describes the scene as a square and parallelogram.

Study of this kind of concealment can tell us something about the "principles" according to which our visual system "usually" describes scenes as made up of objects. But once the "4" has been pointed out or discovered, it is then "seen" quite clearly! A good theory must also account for phenomena in which it is possible to change and elaborate one's "image" of the same scene in ways that depend on changes in his interpretation and understanding of the structure "shown" in the picture.

A simpler -- and more interesting -- example of a figure with two competitive descriptions is the ordinary square! Young children know the square and the diamond as two quite distinct shapes, and the ambiguity persists in adults, as seen here. See [Attneave, 19%%].

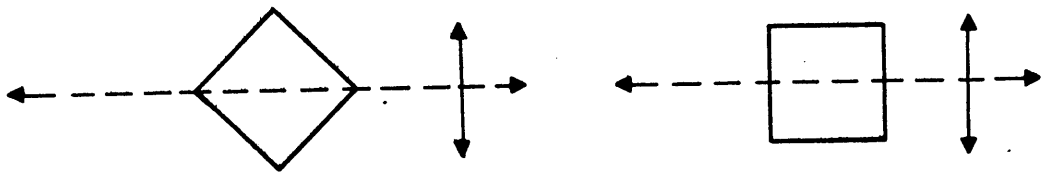
The four objects at the left are usually seen as diamonds, while those on the right are seen as squares.How can we explain this? Since the individual objects are in fact identical, the effect must have something to do with their arrangement. It is tempting to incant the phrase -- "the whole is more than the sum of the parts."



Now consider a descriptive theory. If one is asked to describe this scene, he will say something like: "There are two rows, each with four objects. One is a horizontal row of -- etc". We ignore details here, but suggest that the description is dominated by the grouping into rows, as indicated by their priority in the verbal presentation of the description. In section 4.6 we discuss a program that does something of this sort.

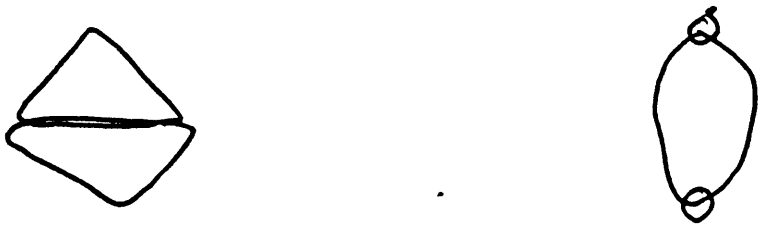
> By "description" we do not usually mean "verbal description"; we mean an abstract data structure in which are represented features, relations, functions, references to processes, and other information. Besides representing things and relations between things, descriptions often contain information about the relative importance of features to one another, e.g., committments about which features are to be regarded as essential and which are merely ornamental. For example, much of linguistic structure is concerned with the ability to embed hierarchies of detail into descriptions: subordinate clause formation and other word-order choices often reflect priorities and progressions of structural detail in the descriptions that are "meant." We will return to this in section 5.

Once committed to describing a row of things, the choice between between seeing squares and diamonds begins to make more sense. Which description does one choose? Apparently, the way one describes a square figure depends very much on how one chooses (in one's mind) the axis of symmetry. Consider the differences in the figures' descriptions in each of the two obvious choices of orientation shown in the next figure.

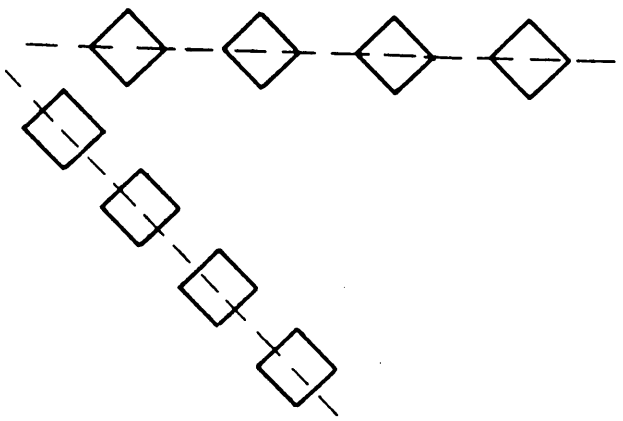


points on axis	sides parallel to axis
one point on each side	two points on each side
made of two triangles	made of two rectangles
unstable on ground	stable -- flat bottom
hurts when squeezed	safe to pick up

These two descriptions could hardly be more different! No wonder that most 3 year olds do not believe that they are the same. In fact, children's drawings of diamonds often come out as



indicating that their descriptive image is a composition of two triangles, or at least that the most important features are the points on the symmetry axes. Our mystery is then almost solved: whatever process set up the description in terms of rows set up also a spatial frame of reference for each group.



Since one has to choose an axis for each square and "other things being equal" there is no strong reason locally for either choice, one tends to use the axis inherited from the direction of its "row." The fact that you can, if you want, choose to see any of the objects as either diamond or square only confirms this theoretical suggestion -- the choice is by default only, and hence would be expected to carry little force.

Once this door is opened, it suggests that other choices one has to make in visual description also can depend on other alien elements in one's thoughts -- as well as on other things in the picture! Every simple figure is highly ambiguous. In a face, a circle can be an eye, a mouth, an ear, or the whole head. There should be no difficulty in admitting this to our theory -- or to the computer programs that demonstrate its consistency and performance. Traditional theories directed toward physical (rather than on computational, or symbolic) mechanisms were inherently unable to account for the influence of other knowledge and ideas upon "perception".

2.2 Sensation, Perception and Cognition

Our discussion of how images depend on states of mind is part of a broader attack on the conventional view of the structure of mind. In today's culture we grow up to believe that mental activity operates according to some scheme in which information is transformed through a sequence of stages like:

WORLD==>SENSATION==>PERCEPTION==>RECOGNITION==>COGNITION==>...

Although it is hard to explain exactly what these stages or levels are, everyone comes to believe that they exist. The "new look" in ideas about thinking rejects the idea that there are separate activities like "perception" that precede and are basically independent of "higher" intellectual activities. What one "sees" depends very much on one's current motives, intentions, memories, and acquired processes. We do not meant to say either that the old layer-cake scheme is entirely wrong or that it is useless. Rather, it represents an early concept that was once a clarification but is now a source of obscurity, for it is technically inadequate against the background of today's more intricate and ambitious ideas about mechanisms.

> The higher nervous system is embryologically, and anatomically divided into stages of some sort and this might suggest a basis for the popular-science hierarchy. This makes sense for the most peripheral sensory and motor systems, in which transmission between anatomical stages is chiefly unidirectional. But (presumably) when we go further in the central direction this is no longer true, and one should not expect the geometrical parts of a cybernetic machine to correspond very well to its "computational parts."

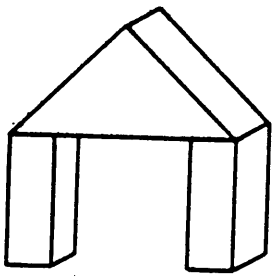
Indeed, the very concept of "part", as in a machine, must be rebuilt when discussing programs and processes. For example, it is quite common in computer programs -- and, we presume, in thought processes -- to find that two different procedures use each other as subprocedures! We shall see this happening throughout section 5. In such a case one can hardly think of either process as a proper part of the other. So the traditional view of a mechanism as a HIERARCHY of parts, subassemblies and sub-sub-assemblies (e.g., the main bearing of the fuel pump of the pitch vernier rocket of the second ascent stage) must give way to a HETERARCHY of computational ingredients.

It is unfortunate that technical theories, and even practical guidelines, for such heterarchies are still in their infancies. The rest of this chapter discusses some aspects of this problem.

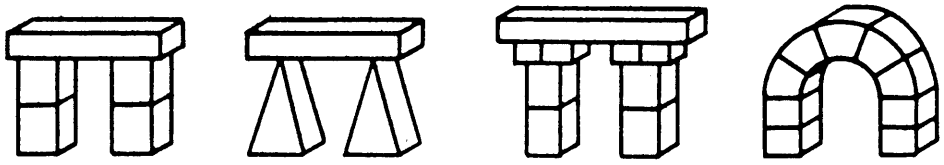
2.3 Parts and Wholes

A recurrent theme in the history of psychological thinking involves recognizing an important distinction without having the technical means to give it the appropriate degree of precision. Consequently, the dividing line becomes prematurely entrenched in the wrong place. An influential example was the concept of "Gestalt". This word is used in attempts to differentiate between the simplest immediate and local effects of stimuli, and those effects that depend on a much more "global" influence of the whole stimulus "field".

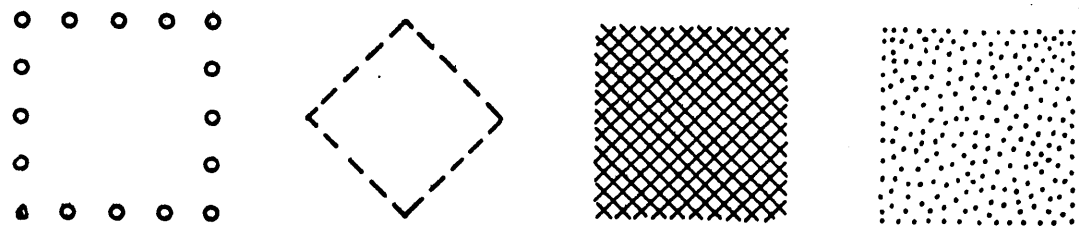
Here is a visual example in which this kind of distinction might be considered to operate: In one sense, this arch is "nothing but" three blocks.



But the arch has properties -- as a single whole -- that are not inherited directly from properties of its parts in any simple way. Some of those arch properties are shared also by these structures:



Obviously the properties one has in mind do not reside in the individual building blocks, they "emerge" from the arrangements of those parts. And one finds this in even simpler situations. Obviously we react to a simple outline square in a way that is very different from our reactions to four separate lines, and rather similar to how we react to such graphically different figures as these:



The question "whence comes the square if not from its parts" is not really very serious here, for it is easy to make theories about how one might "perceive" a shape if there are enough easily-detected features to approximately delineate its geometric form. But there is no similarly easy solution to the kinds of problems that arise when one looks at three-dimensional scenes.

The next two figures are "locally identical" in the following precise sense: Imagine innumerable experiments, in each of which we choose a different point of the picture to look at, and record what we see only within a very small circle around that point.

It is not hard to see that this definition will accept any picture that contains only solid rectangles, but no other kind of picture. So in this sense "rectangle-ness" can be defined in terms of local properties, while connectedness cannot. Try to define "composed-of-a-single-solid-rectangle" in this way. It cannot be done! So we see a difference between two kinds of categories of pictures, in regard to the relations between their parts and their wholes!

The question "Is the whole more than the sum of its parts" is certainly provocative and insightful. But it must be recognized also as vague, relative, and metaphorical. What is meant by "parts" and, more important, what is meant by "sum"?

In the case of the rectangles a trivial sense of "sum" will suffice: not even adding up evidence is necessary, for we can make the decision in favor of rectangle, and let any single exception to our condition on the local "micro-scenes" have absolute veto power. So the "sum of the parts" is simply the agreement of all local evidence. For connectedness we seem to need something more complicated, computationally. We have studied this situation rather deeply in PERCEPTRONS: connectedness is a property that is quite important and very thoroughly understood in classical mathematics ; it is in fact the central concern of the entire subject of Topology.

For example, here are several quite different-looking conditions each of which can be used to define the same concept of connectedness:

> PATH-CONNECTION. For any two black points of the picture, there is a path connecting them that lies entirely in black points.
>
> PATH-SEPARATION. There is no closed path, entirely in white points, such that there are some black points inside the path and some black points outside the path.
>
> SET-SEPARATION. The black points cannot be divided into two non-empty sets which are separated by a non-zero distance -- that is, no pair of points, one from each set, are closer than a certain distance.
>
> TOTAL-CURVATURE. Assume that there are no "holes" in the black set -- that is, white points that are cut off from the outside by a barrier of black points. Then compute the sum of all the boundary curvatures (direction-changes at all edges of the figure), taking convex curves as positive and concave curves as negative. The picture is connected if this sum is exactly 360 degrees. If it is a multiple of 360, this gives the number of objects!

Each of these suggest different computational approaches. Depending upon what resources are available, one or another will be more efficient, use more or less memory, time, hardware, etc. Each definition involves very large calculations in any case, except the fourth, in which one computes simply a sum of what one observes in each small neighborhood. However, the fourth definition does not work in general, but only for figures without holes. And, to be sure that condition is satisfied one must have another source of information (e.g., if one knows he is counting pennies) or else the definition is somewhat circular, because to be able to see that there are no holes is really equivalent to being able to see that the background is connected!

We know exactly what it means for the number seven to be the sum of the numbers three and four. But when we ask whether a house is just the sum of its bricks, we are in a more complicated situation. One might answer:

> "Yes, there is nothing but bricks there".

But another kind of answer could be

> "No, for the same bricks arranged differently would have made a very different house."

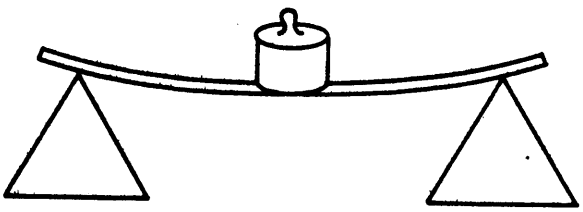
The answer must depend on the purpose of the question. If we admit only "yes" or "no", there is no room for refinement and subtlety of discussion. We do not really want either of the answers "Yes, it is nothing but the sum" or "No, it is a Gestalt, a totally different and new thing". We really want to know exactly how the response, image, or interpretation of the situation is produced: we want an explanation of the phenomenon. And the terms of the explanation must be appropriate to the kind of technical question we have in mind. Sometimes one wants the result in terms of a particular set of psychological concepts, sometimes in terms of the interconnections of some perhaps hypothetical neural pathways, and sometimes in terms of some purely computational schemata.

Thus one might ask, about some aspect of a person's behavior:

> COMPONENTS: Can the phenomenon be produced in a certain kind of theoretical neural network?
>
> LEARNING: Can it be learned by a certain kind of reinforcement schedule according to certain proposed laws of conditioning?
>
> COMPUTATIONAL STRUCTURE: Can this result be computed by a computer-like system subject to certain restrictions, say, on the amount of memory, or on the exclusion of certain kinds of loops interconnecting its components?
>
> COMPUTATIONAL SCHEMATA: Can the outer behavior of this individual reasonably be imitated by a program containing such-and-such a data-structure and such-and-such a syntactic analyser and synthesizer?

The way in which the whole depends upon its parts, for any phenomenon, has a direct bearing on how such questions can be answered. But to supply sensible answers, one needs a stock of crisp, precise, ideas about how parts and wholes may be related!

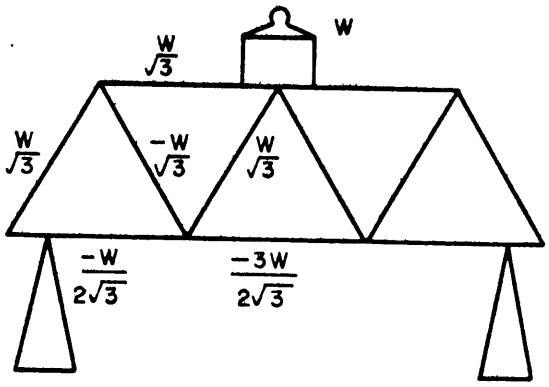
It is important to recognize that these kinds of problems are not special to Psychology. Water has properties that are not properties either of hydrogen or oxygen, yet chemistry is no longer plagued by fights between two camps -- say, "Atomist" vs. "Gestalt". This is not at all because the problem is unimportant: exactly the opposite! The reason there are no longer two camps in Chemistry is because all workers recognize that the central problems of the field lie in developing good theories of the different kinds of interactions involved, and that the solution of such problems lie in constructing adequate scientific and mathematical models rather than in defending romantic but irrelevant philosophical overviews. But in Psychology and Biology, there remains a widespread belief that there are phenomena of mind or of cell that are not "reducible" to properties and interactions of the parts. They are saying, in essence, that there can be no adequate theory of the interactions.



SUPPORTED ROD

Consider a concrete example. It is relatively easy to bend a thin rod, but much harder to bend this structure made of several such rods. Where does the extra stiffness come from?

The answer, in this case, is that the "new property" is indeed inherited from the parts, because of the arrangement, but in a peculiar way. In the truss, a force at the middle is resisted -- not by bending-forces across the rods -- but by compression and tension forces along the rods.



TRUSS

The resistance of a thin rod to forces along it is much greater than the resistance to forces across it. So the increased strength is indeed "reduced", in the Theory of Static Mechanics, to the interactions of stresses between members ofthe structure. Even the properties of a single rod itself can be be explained in terms of more microscopic interactions of the tensile and compressive forces between its own (!) "parts", when it is strained. By imagining the rod itself to be a truss (a heuristic planning step that helps one to write down the correct differential equation) we can analyse stress-strain relations inside the rod. Thus one obtains such a beautiful and accurate model that there remains no mysterious "Gestalt" problem at all.

This is not to say that special arrangements have no special properties. In some of Buckminster Fuller's work, the dodecahedral sphere is yields a kind of structural stiffness rather different than that in the triangular truss. Here the rigidity does not come directly from that of small or "local" triangular substructures, and it takes a different kind of mathematical analysis to see why it is hard to distort it. Even so, there remains no mysterious "emergent" property here that cannot be deduced from the classical theory of statics.

Of course, our real concern is with problems of intelligence, rather than with engineering mechanics. But many problems that seem at first to be "purely psychological" often turn out to center around just such problems of wholes and parts. And with such an interpretation, we may replace an elusively ill-defined psychological puzzle by a much sharper problem within the theory of computation.

The computer is the example par excellence of mechanisms in which one gets complex results from simple interactions of simple components. In asking how thought-like activity could be embedded in computer programs, scientists for the first time really came to grips with understanding how intelligent behavior could be made to emerge from simple interactions.

The issue seems really to be fundamentally one of assessing the complexity of processes. The content of the gestalt discoveries is that certain psychological phenomena require forms of computation that lie outside the scopes of certain models of the brain -- and outside certain conjectures about the "elementary" units of which behavior is supposed to be composed. So, the whole discussion must be considered in relation to some overt or covert committment about what units of behavior, or of brain-anatomy, or of computational capacity, are supposed to be "atomic".

To illustrate extreme versions of atomism vs. gestaltism one might consider these caricatures:

> Extreme ATOMISM: all behavior can be understood in terms of simple functions of neural paths that run from single receptors, through internuncials, to effectors.
>
> Extreme GESTALTISM: The essence in is the whole pattern. Many simple examples show that the response is made to the whole stimulus and cannot be represented as simple sums or products of simple local stimulations.

Clearly one does not want to set a threshold between these; one wants to classify intermediate varieties of interactions that might be involved, arranged if possible in some natural order of complexity.

Thus in PERCEPTRONS we studied a variety of simple schemas such as these:

> EXTREMELY ATOMIC ALGORITHM: One of the input wires is connected to the output, the others to nothing.
>
> VETO ALGORITHM: If every input says "yes", the output is "yes". If any input says "no", the output is "no".
>
> MAJORITY ALGORITHM: If M or more of N inputs say "yes", output is "yes".
>
> LINEAR SUM ALGORITHM: To each input is assigned a "weight". Add together the weights for just those inputs that say "yes". The output is just this sum.
>
> LINEAR THRESHOLD ALGORITHM: Use the LINEAR SUM algorithm, except, make the output "yes" if the sum is greater than a certain "threshold", otherwise the output is "no".

Exercise: the reader should convince himself that "extremely atomic", "veto", and "majority" are special cases of "linear threshold".

> EQUIVALENT-PAIR ALGORITHM: The input is considered to be grouped in pairs. The output is "yes" only when, for every pair, the two members have the same input value.

The reader should convince himself that this is not a special case of "linear threshold"!

> SYMMETRICAL ALGORITHM: The response is "yes" if the pattern of inputs is symmetrical about some particular center, or about some particular linear axis.

This is a special case of the equivalent-pair algorithm. They are both examples of perceptrons in which the global function can be expressed as a linear threshold function of intermediate functions of two variables. Here the whole is only trivially more than the sum of the parts.

> PERCEPTRON ALGORITHM: First some computationally very simple functions of the inputs are computed, then one applies a linear threshold algorithm to the values of these functions.

Many different classes of perceptrons have been studied; such a class is defined by choosing a meaning for the phrase "very simple function." For example, one might specify that such a function can depend on no more than five of the stimulus points. This would result in what is called an order-five perceptron. All of the examples above had order one or two. The next example has no "order restriction", but the functions are very simple in another sense; they are themselves "order one" or linear-threshold functions.

> GAMBA PERCEPTRON: A number of linear threshold systems have their outputs connected to the inputs of a linear threshold system. Thus we have a linear threshold function of many linear threshold functions.

Virtually nothing is known about the computational capabilities of this latter kind of machine. We believe that it can do little more than can a low order perceptron. (This, in turn, would mean, roughly, that although they could recognize some relations between the points of a picture, they could not handle relations between such relations to any significant extent.) That we cannot understand mathematically the Gamba perceptron very well is, we feel, symptomatic of the early state of development of elementary computational theories.

Which of these are atomic and which gestaltist? Rather than muddle through a philosophical discussion of which cases "really" do more than add the parts, we should try to classify the kinds of mechanisms needed to realize each in certain "hardware" frameworks, chosen for good mathematical reasons. Then for each such framework, we might try to see which admit simple reinforcement mechanisms for learning, which admit efficient descriptive teaching (see section 4), which admit the possibility of the cognitive machinery "figuring out for itself" what are the important aspects of a situation!

To supply such ideas, we have to make theoretical models and systems. One should not expect to handle complex systems until one thoroughly understands the phenomena that may emerge from their simpler subsystems. This is why we focussed so much attention on the behavior of Perceptrons in problems of Computational Geometry. It is important to emphasize that we want to understand such systems for the reasons explained above, rather than as possible mechanisms for practical use. When a mathematical psychologist uses terms like "linear", "independent", or "Markoff Process", etc., he is not (we hope!) proposing that a human memory is one of those things; he is using it as part of a well-developed technical vocabulary for describing the structure of more complicated schemata. But until recently there was a serious shortage of ways to describe more procedural aspects of behavior.

The community of ideas in the area of computer science makes a real change in the range of available concepts. Before this, we had too feeble a family of concepts to support effective theories of intelligence, learning, and development. Neither the finite-state and stimulus-response catalogs of the Behaviorists, the hydraulic and economic analogies of the Freudians, or the holistic insights of the Gestaltists supplied enough technical ingredients to develop such an intricate subject. It needs a substrate of debugged theories and solutions to related but simpler problems. Computer science has brought a flood of such ideas, well defined and experimentally implemented, for thinking about thinking; only a fraction of them have distinguishable representations in traditional psychology:

symbol table	closed subroutine
pure procedure	pushdown list
time-sharing	interrupt
calling sequence	communication cell
functional argument	common storage
memory protection	decision tree
dispatch table	hardware-software trade-off
error message	serial-parallel trade-off
function-call trace	time-memory trade-off
breakpoint	conditional breakpoint
formal language	asynchronous processing
compiler	interpreter
indirect adress	garbage collection
macro language	list structure
property list	block structure
data type	look-ahead
hash coding	look-behind (cache)
micro-program	diagnostic program
format matching	executive program
syntax-direction	operating system

These are only a few ideas from the environment of general "systems programming" and debugging; we have mentioned none of the much larger set of concepts specifically relevant to programming languages , artificial intelligence research, computer hardware and design, or other advanced and specialized areas. All these serve today as tools of a curious and intricate craft, programming. But just as astronomy succeeded astrology, following Kepler's discovery of planetary regularities, the discoveries of these many principles in empirical explorations of intellectual processes in machines should lead to a science, eventually.

Before discussing scene-analysis in detail, we have a few remarks about the nature of problems in this area. In the early days of cybernetics [McCulloch-Pitts 1943, Wiener 1949] it was felt that the hardest problems in apprehending a visual scene were concerned with questions like "why do things look the same when seen from different viewpoints", when their optical images have different sizes and positions.

How does one capture the "abstraction" or "concept" common to all the particular examples. For two-dimensional character-recognition, this kind of problem is usually handled by a two-step process in which the image is first "normalized" to standard position and then "matched" -- by a correlation or filtering process -- to one of a set of standard representatives. In practical engineering applications, the "normalizing" often failed because it could not disarticulate parts of images that touch together, and "matching" often failed because it is hard to make correlation-like processes attend to "important" parts of the figures instead of to ornaments. Even so, such methods work well enough for reasonably standardized symbols.

If, however, one wants the machine to read the full variety of typography that a literate person can, the problem is harder, and if one wants to deal with hand-printing, quite different methods are needed. One is absolutely forced to use exterior knowledge involving the pictures' contexts, in situations like this. [Selfridge, 1955]

Here the distinction between the "H" and the "A" is not geometric at all, but exists only in one's knowledge about the language. An early program that could do this was described in [Bledsoe and Browning 1959]. But we will not stop to review the field of character-recognition, for its technology is quite alien to the problems of three-dimensional scenes. This is because the problems that concern us most, like how to separate objects that overlap, or how to recognize objects that are partially hidden (either by other objects or by occluding parts of their own surfaces), simply do not occur at all in the two-dimensional case. Some more interesting two-dimensional problems require description when geometric matching fails; a conceptual "A" is not simply a particular geometric shape; it is

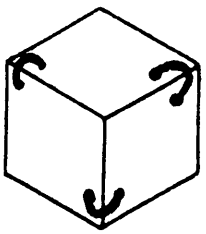
> "Two lines of comparable length that meet at an acute angle, connected near their middles by a third line."

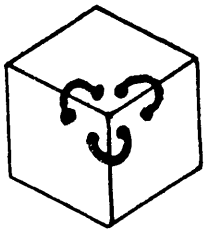
3.1 Programs for finding bodies in scenes

Let us review quickly how Guzman's SEE program works. First a collection of "lower level" programs are made to operate directly on the optical data. Their job is to find geometric features of the picture -- regions, edges and vertices -- so that the scene can be described in a simple way in the program's data-structure. Next, the vertices are classified into "types". The most important kinds are these:

The main goal of the program is to divide the scene into "objects", and its basic method is to group together regions that probably belong to the same object. Each type of vertex is considered to provide some evidence about such groupings, and can be used to create "links" between regions.

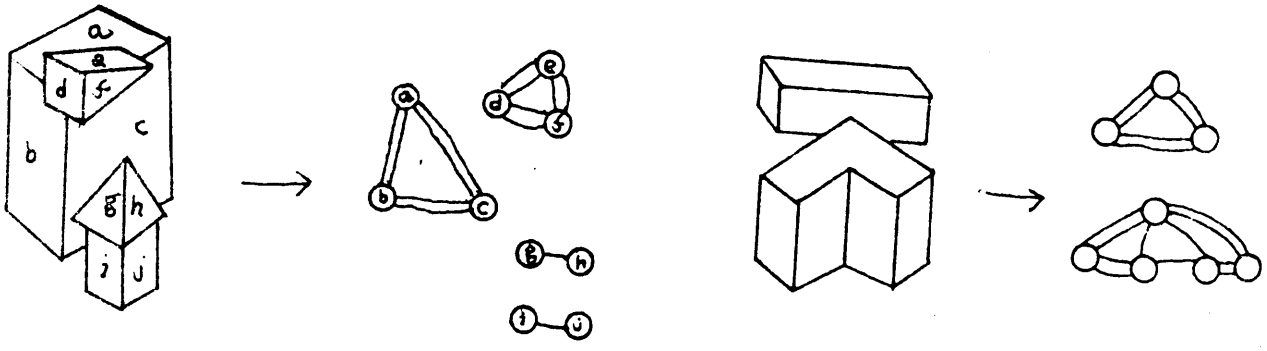
For example, the ARROW type of vertex usually is caused by an exterior corner of an object, where two of its plane surfaces form an edge. So we insert a "link" between the two regions that are bounded by the two smaller angles:



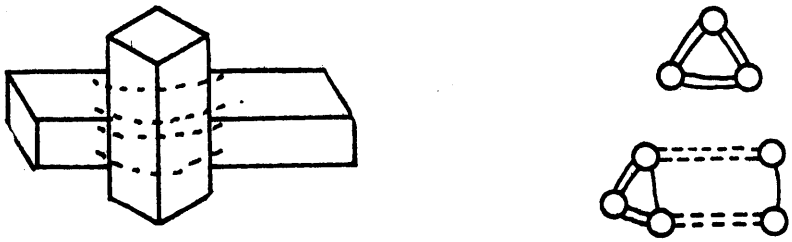


Similarly, the FORK type of vertex, which is usually due to three planes of one object, causes three links between those regions.

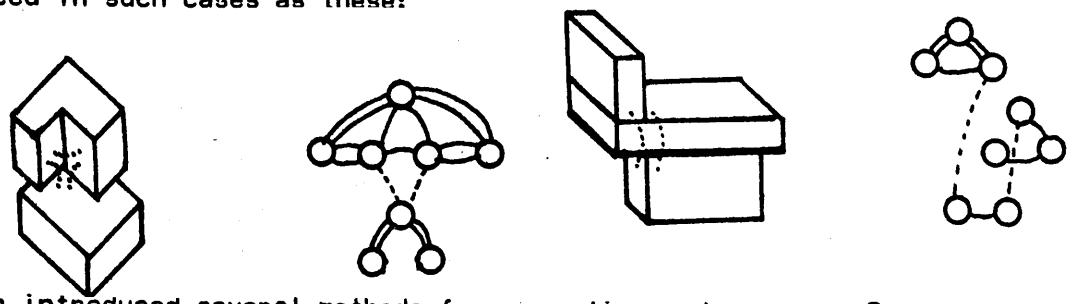
Using these clues, and representing the resulting relations by simple abstract networks, many scenes are "correctly" analyzed into objects.



If two TEE vertices have their stems in the same line then we create two more links: This often does just the right thing for an object whose picture is divided into two separate parts by another object in front.

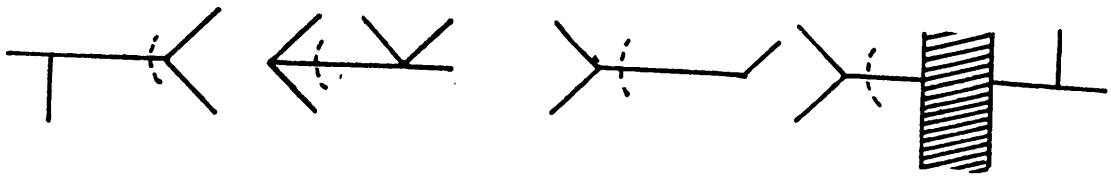


Many scenes are handled correctly by just these simple rules, but many are not. For example, the basic assumption about the FORK linking its three regions is not true of concave corners, and the "matching TEE" assumption may be false by coincidence, so that "false links" may be produced in such cases as these:



Guzman introduced several methods for correcting such errors. One method involves a conservative procedure in which groupings are considered to have different qualities of connectedness. Two high-quality groups that are connected together by only a single link are broken apart -- the link is deleted.

A second error-correction method is more interesting. Here we observe that the TEE vertex really has a special character, quite opposed to that of the FORK and the ARROW. The most usual physical cause of a TEE is that an edge of one object has disappeared under an edge of another object. Hence we should regard the TEE joint as evidence against linking the corresponding regions! Guzman's implementation of this was to recognize certain kinds of configurations as special situations in which the existence of one kind of vertex-type causes inhibition or cancellation of a link that would otherwise be produced by the other vertex-type. That would happen, for example, in these figures:

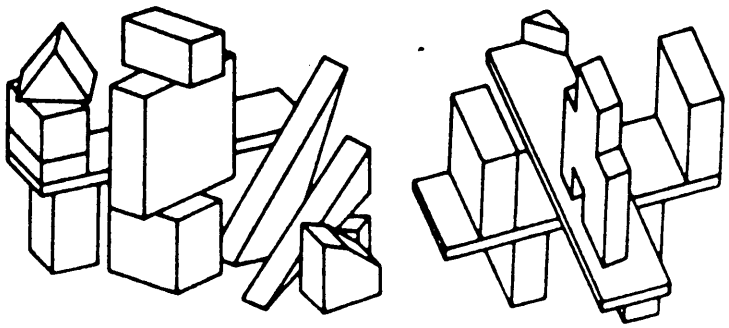


This technique corrects many errors that the more "naive" system makes, especially in objects with concavities. Note that it attempts to compute Connectedness: -- for is not the notion of "object" as we are using it exactly that idea? -- by extremely local methods, while the (better) system with cancellation is less local beacuse of the effects of vertex-types of contiguous or closely-related geometric features.

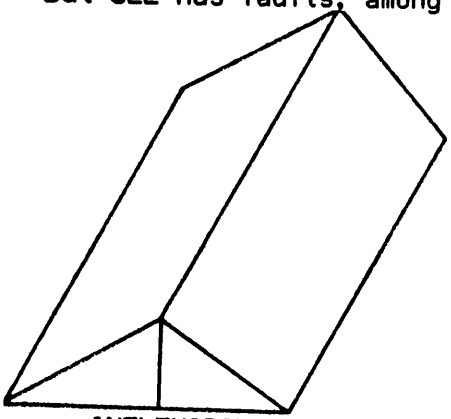
Guzman's method might seem devoid of the normalization and matching operations. Indeed, in a sense it has nothing to do with "recognizing" at all; it is concerned with the separation of bodies rather than with their shapes. But both normalization and matching are more or less inherent in the descriptive language itself, since the very idea of vertex-type is that of a micro-scene which is invariant of orientation, scale, and position. This scheme of Guzman is very much in accord with the Gestaltists' concptual scheme in which the separation of figure from background is considered prior to and more primitive than the perception of form.

The "cancellation" scheme has a more intelligible physical meaning. It has been pointed out by D. Huffman [1970] that each line in a line-drawing may be interpreted as a physical edge formed (we assume) by the intersection of two planes, at least locally. In some cases one can see parts of both planes, but in other cases only one. A T-joint is good evidence that the edge involved is of the latter kind, and once one assigns such an interpretation to an edge, then it follows immediately that the adjacent Guzman links to the alien surface ought to be rejected. Accordingly, Huffman developed a number of procedures for making detailed global interpretations from local edge-region assignments.

We will not give further details of the SEE program here. As an example of its performance, it correctly separates all the objects in this scene.

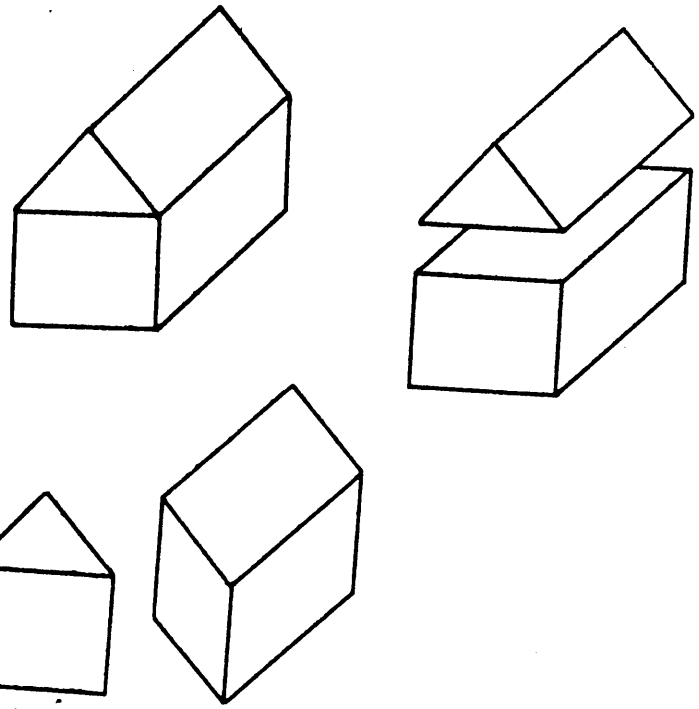


But SEE has faults, among which are:



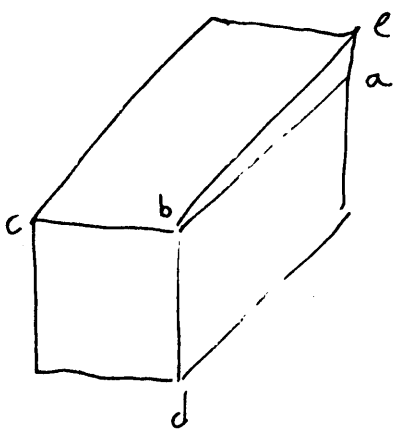
ORDINARY "MISTAKES": Certain simple figures are not handled "correctly." To be sure, all figures are inherently ambigous (any scene with n regions could conceivably arise from a picture of n objects). Our real goal is to find an analysis that makes sense in everyday situations. Normally one would not suppose that this is a single body, but SEE says it is, because all regions get linked together.

INFLEXIBILITY: If its very first proposal is not acceptable, the body-aggregation program ought to be able to respond to complaints from other higher and lower level programs and thus generate some alternative "parsings" of the scene. For example, SEE finds a single body in the top one of these figures, but it should be able to produce the two other alternatives shown below it. (It is interesting how difficult it is for some humans to see the third parsing.)



IGNORANCE: It has no way to use knowledge about common or plausible shapes. While it is a virtue to be able to go so far without using such exterior information, it is a fault to insist on this!

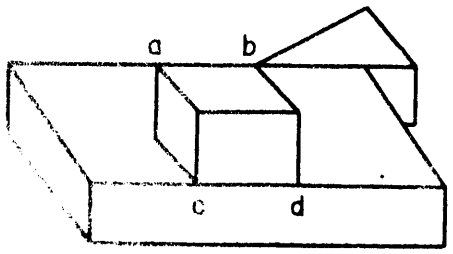
Following Guzman's work, Martin Rattner has described a procedure, called SEEMORE, that can handle some of these problems. [Rattner 1970] While it uses linking heuristics much as did Guzman, SEEMORE puts more emphasis on local evidence that an edge might separate two bodies. These "splitting heuristics" operate initially at certain kinds of vertices, notably TEE-vertices and vertices with more than three edges (which were not much used in earlier programs). When there is more than one plausible alternative, SEEMORE uses other evidence to make tentative choices of how to continue a splitting line, but stores these choices on back-up lists that can later be used to generate alternative parsings.

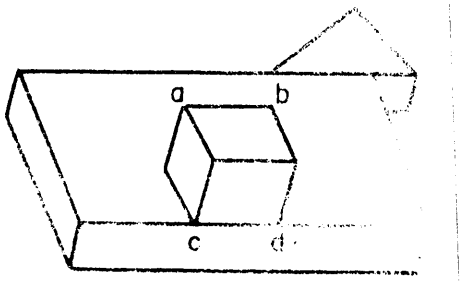


Here is a simple example. In this figure, one might imagine splitting either along the line a-b-c or along the line d-b-e. The central vertex 'b' suggests (locally) either of these; on the other hand, such splits as a-b-d or a-b-e are considered much less likely.

The vertex 'a' strongly suggests a split along a-b, while neither 'c', 'd', nor 'e' have much in their favor. Thus SEEMORE starts a split at 'a' and continues at 'b' toward 'c'. Generally, splits originate at TEE's, propagate through L's and matching TEE's, and avoid the sharpest turns through the multiple-edge vertices.

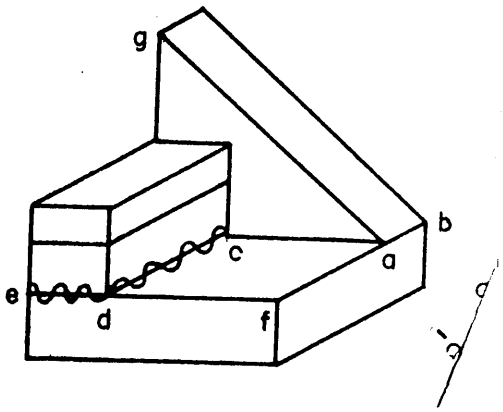
Degenerate situations like this, in which a small change in viewing





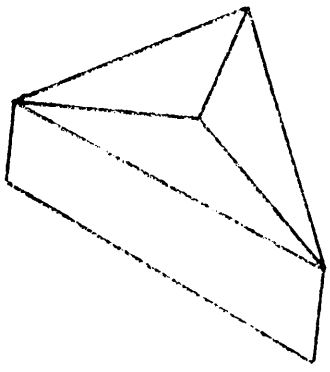
angle produces a different topology, are likely to lead to "incorrect" analyses. Rattner uses a rather conservative linking phase, in which links are placed more cautiously than in SEE, but using similar "inhibiting" rules. Regions that are doubly-linked to one another by these are considered "strongly" bound; then the heuristic rule is to attempt to split around these "nucleii," and to avoid splitting through them.

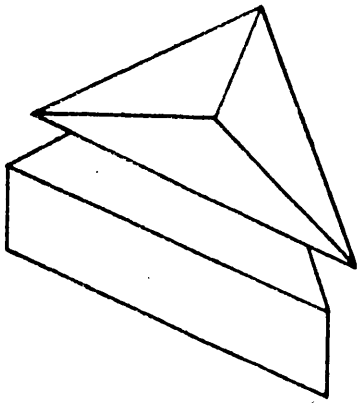
It would be tedious to give full details here, partly because the subject is so specialized, but primarily because the procedure has not been tested and debugged in a wide enough variety of situations. A few examples follow.

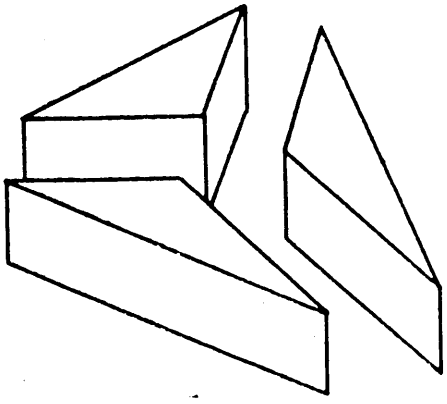


An initial split is made along e-d, extended to d-c. Then, between the possible splits g-a-f and c-a-b, the latter is preferred because it completes the unfinished split ending at 'c'.

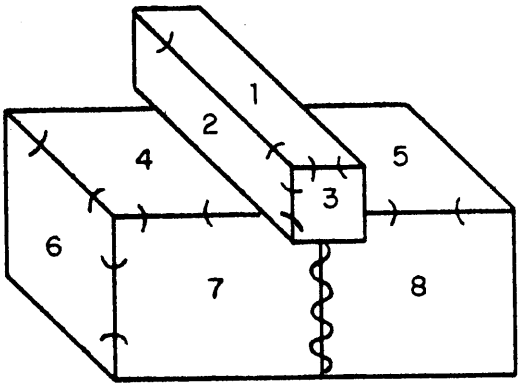
In this situation, B is the procedure's first choice, C its second:



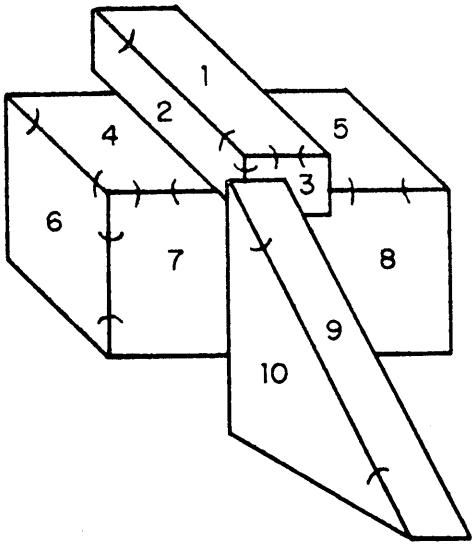




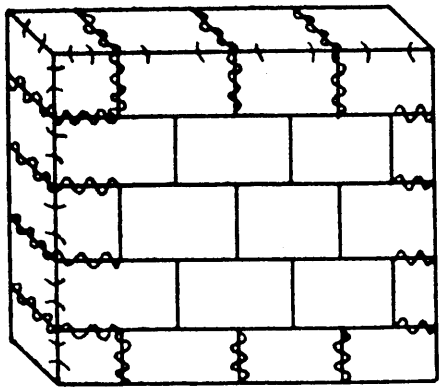
In A below, we get three bodies, (4-6-7), (5-8), and (1-2-3). SEE does not split between regions 7 and 8. In B, one gets the plausible three-body analysis. If there is any complaint, SEEMORE will propose to separate (4-6-7) amd (5-8). In C, all the bricks are properly separated. While SEE would have to put in many spurious links because of the coincidentally matching TEE's, SEEMORE inhibits these on the basis of other splitting evidence.



(A)

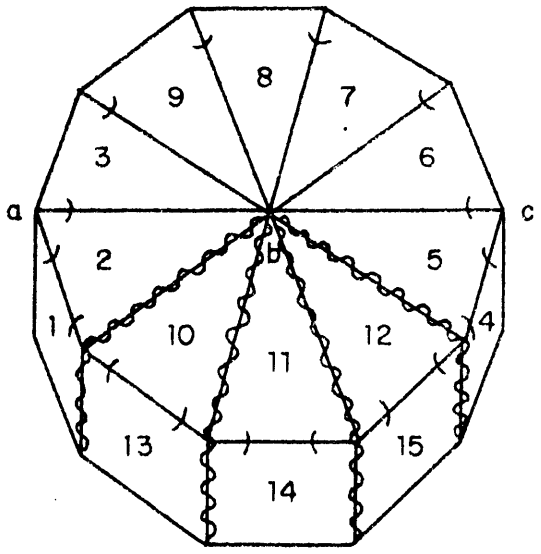


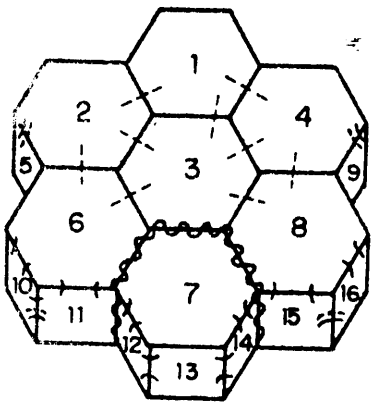
(B)



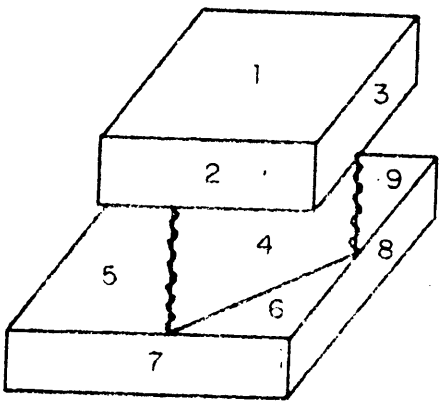
(C)

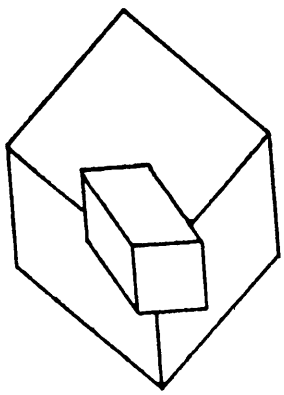
The procedure divides these into the "natural" parts:





But in figure A below it finds three bodies 1-2-3, 5-7-8-9, and 4-6. The latter is perhaps not the first way a person would see it. And the procedure cannot aggregate the outer segments of the larger cube in figure B because its initial grouping process is so conservative. Clearly, such problems eventually must be gathered together in a "commonsense" reasoning system; the multiple T-joints all would meet, if "extended" in such a way as to suggest the proper split, and the program ought to realize this.





Chapter 4. DESCRIPTION AND LEARNING

The concepts we used to analyse ANALOGY and SEEING are just as vital in understanding LEARNING. It was traditional to try to account for learning in terms of such primitives as "conditioned reflex" or "stimulus-response bond". The phenomena of learning become much more intelligible when seen in terms of "description" and "procedure".

> There might seem a world of difference between activities involving permanent changes in behavior -- and the rest of thinking and problem-solving. But even the temporary structures one obviously uses in imagining and understanding have to be set up and maintained for a time. We feel that the differences in degree of permanence are of small importance compared to the problems of deciding what to remember. It is not the details of how recording is done, but the details of how one solves the problem of what to record, that must be understood first.

As we develop this idea, we find ourselves forced to question the whole tradition in which one distinguishes a special sub-set of mental or behavioral processes called "learning". Nothing but disaster can come from looking for three separate theories to explain (for example)

How one learns mathematics,

How one thinks mathematically once he has learned to,

and

What mathematics is, anyway.

We are not alone in trying to replace such subdivisions -- but perhaps more radical and thorough-going. In this chapter we shall argue that many problems about "learning" really are concerned with the problem of finding a description that satisfies some goal. Gestalt psychologists also often emphasized the similarity between solving apparently abstract problems and situations that intuitively feel like simple perception; the same relation that is dimly reflected in ordinary language by expressions like

"I suddenly saw the solution!"

We thoroughly agree about bringing these phenomena together, but we have a very different way of dealing with the newly united couple. We might caricature this difference by saying that the Gestaltists might look for simple and fundamental principles about how perception is organized, and then attempt to show how symbolic reasoning can be seen as following the same principles, while we might construct a complex theory of how knowledge is applied to solve intellectual problems and then attempt to show how the symbolic description that IS what one "sees" is constructed according to similar such processes. Indeed, we think that ideas that have come from the study of symbolic reasoning have done more to elucidate visual perception than ideas about perception have clarified our thoughts about abstract thinking -- but the whole comparison is too dialectical to try to develop technically.

In any case we differ from the Gestaltists more deeply in problems of learning, which they neglected almost entirely -- perhaps because that was the favorite subject of the abominable behaviorists!. Let us now explain why we feel that learning, technically, cannot usefully be separated from other aspects either of perception or of symbolic reasoning. As usual, we present first a caricature; then point to where the extreme positions might be softened.

Learning -- or "Keeping Track"

Everyone would agree that getting to know one's way around a city is "learning". Similarly, we see solving a problem often as getting to know one's way around a "micro-world" in which the problem exists. Think, for example, of what it is like to work on a chess problem (or on a geometry puzzle, or trying to fix something). Here the microworld consists of the network of situations on the chessboards that arise when one moves the pieces. Solving the chess problem consists largely of getting to know the relations between the pieces, and how the moves affect things. One naturally uses words like "explore" in this context. As the exploring goes on, one experiences events in which one suddenly "sees" certain relations. A grouping first seen as three pieces playing different roles is now described in terms of a single relation between the three, such as "pin", "fork", or "defense." The experience of re-description can be as "vivid" as if the pieces involved suddenly changed color or position.

One might object that the difference between getting to know the city and solving the chess problem is that one remembers the city and forgets the chess situation (assuming that one does). Isn't that what brings one into the domain of learning and excludes the other? Only to a degree! The chess analysis has to be remembered long enough, within the rest of the analysis. To take an extreme form of the argument, one would repeat one's first steps forever unless one remembered which positions had been analyzed, what relations were observed, and how their descriptions were summarized. What is stored within problem-solving is as vital to the immediate solution as what is retained afterwards is to the solution of the presumably larger-scale problems one is embedded in throughout life. Of course there is a problem about how long one retains what one learns -- but perhaps that belongs to the theory of forgetting rather than of learning!

In our laboratory the chess program written by R. Greenblatt plays fairly good chess, by amateur tournament standards. But visitors are always disappointed to find that this program does not "learn", in the sense that it carries no permanent change away from the games it plays. They are even more disappointed in our attempts to explain why this does not disturb us very much. We claim that there is indeed an important kind of learning within the program; this is in the position-description summaries that are constructed and used as it analyses the positions it is playing. But because board positions do not often repeat exactly in subsequent games (except for opening positions and end-games) and because the kinds of descriptions the program now uses do not have good qualities for dealing with broader classes of positions, there would be no point in keeping such records permanently.

We do not yet understand how to make the higher-level strategy-oriented descriptions that wouls make sense in the context of learning to improve. When we, ourselves, learn how to construct the right kind of descriptions, then we can make programs construct and remember them, too, and the problem of "learning" will vanish. In the past, our Laboratory avoided experiments with learning systems that seemed theoretically unsound, although we did NOT avoid studying them theoretically. This was because we believed that learning itself was not the real problem; what was needed was more knowledge about the intelligent shaping of description-handling processes. For the same reasons we avoided linguistic exercises such as Mechanical Translation, in favor of studying systems that could deal with limited fragments of meaning, and we avoided "creative" systems based on uninterpreted stochastic processes in favor of analysing the interactions of design goals and constraints. Now we think we know enough to begin such experiments.

In the rest of this chapter we will discuss some systems that do exhibit some non-trivial learning functions. It should be understood from the start that these are not to be thought of as "self-organizing systems". They are equipped with very substantial initial structures; -- they are provided with many built-in "innate ideas".

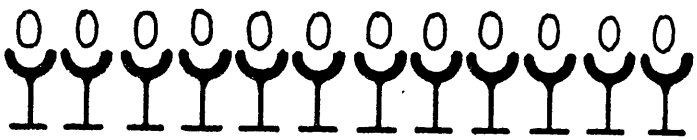
Because of this, some readers might object that although these programs learn, they do not significantly "learn to learn". Is this a serious objection? We do not think so, but the question is really one of degree and we are still much too uncertain about it to take a decisive position. In one view learning to learn would be an extremely advanced problem compared to what we now understand. In another view it is just one more problem about certain kinds of program-writing processes, not strikingly different from the static structural situations we already understand rather well. Our position is intermediate between these, at present.

We think that learning to learn is very much like debugging complex computer programs. To be good at it requires one to know a lot about describing processes and manipulating such descriptions. Unfortunately, work in Artificial Intelligence has not, up to now, been pointed very much in that direction, so today we have little real knowledge about such matters.

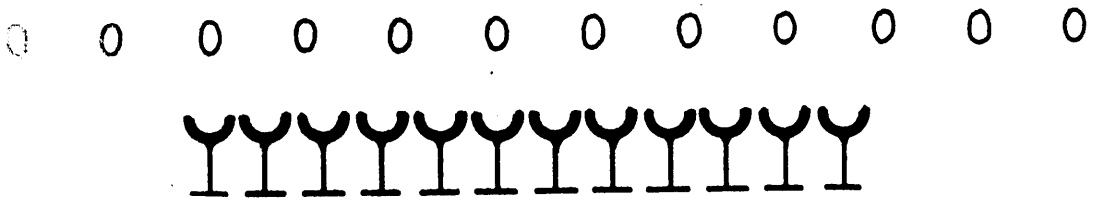
Consequently we are in a poor position to estimate how complex must be the initial endowment of intelligent learners -- ones that could develop as rapidly as human minds rather than requiring evolutionary epocks. We certainly cannot assume from what we know that the "innate structure" required must be very, very complex as compared to present programs. It might be much simpler. Even in the case of humans we have no useful guidelines. There is probably enough potential genetic structure to supply large innate behavioral programs but no one really knows much about this, either, at present. So let us proceed, instead, to discuss our present understanding. We begin with some experiments on natural intelligence.

4.1 An Example of Learning: Piaget's Conservation Experiments

A classical experiment of Jean Piaget shows remarkably repeatable patterns of response of children (in the age range of 4-7 years) to questions about this sort of material:



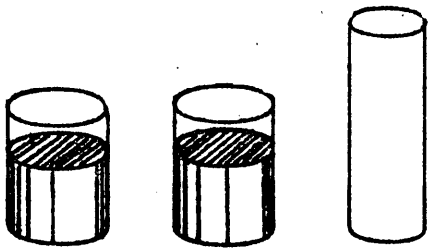
Question: "Are there more eggs or more egg-cups?"
Typical Answer: "No, the same."



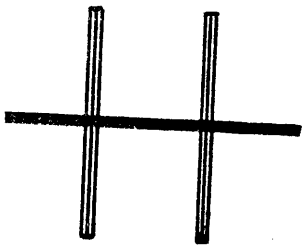
Question: "Are there more eggs or more egg-cups?"
Typical Five Year Old's Answer: "More eggs."
Typical Seven Year Old's Answer: "Of course not!"

Further questioning makes it perfectly clear that the younger child's comparison is based on the greater "spread" or space occupied by the eggs. The older child ignores or rejects this aspect of the situation and is carried along by the "conservationist" argument: before we spread them out there were the same number of eggs and egg-cups; we neither added or subtracted any, so the number must still be the same.

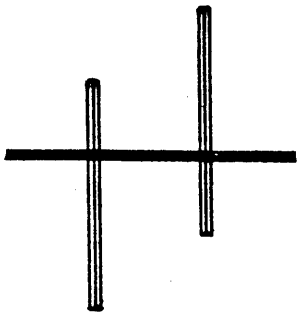
Before constructing a theory of this we describe some other situations that are similar; nothing is more dangerous than to base a theory on just one example and we want the reader to have enough material to participate and, amongst other things, make rival theories. Here is another relatively repeatable experiment. One shows the child three jars.



He agrees that the first two contain the same amount of liquid. Then, before his eyes, we pour the second jar into the third and ask again about the amounts. Usually, the younger child will say that the tall jar contains more; the older child says "Of course they have the same amount. It is the same water so it could not have changed." If we perform the pouring behind a screen, telling him what we are doing without his seeing it, the younger child also may say the amounts are the same, but may change his mind when he sees it.



In this experiment, younger children agree the rods are equal at first, but when displaced as shown at the right the "upper" one is usually said to be longer.



How can we explain the difference between the less and more mature children. We see two problems here from the point of view of learning. First, how is the pre-conservationist view acquired (and executed); then how is it replaced by a conservationist one? To many psychologists only the second seems interesting. This is because it is tempting to explain the earlier response in terms like "the child is carried away by appearances," or "the child is dominated by its perception," that is, instead of logic. The usual interpretation, then, is that the transition requires the development of some sort of reasoning capacity that allows it to "ignore the appearance" in favor of reasoning about "the thing itself".

There are serious problems with this view, we feel. First, the "appearance" theory is too incomplete; the notion of appearance is not structured enough. Second, we know that much younger children are quite secure (in other circumstances) about the properties of "permanent objects"; they are sufficiently surprised by magic that there is no reason to suppose they lack the required "logic". We do not think they lack any really basic or primitive intellectual ingredient; rather, they lack some particular kinds of knowledge and/or procedures that are appropriate here. Our view is most easily explained by proposing a more detailed mini-theory for the performance of the non-conservation child.

Behind the "appearance" theory lies some sort of assumption that the water in the tall jar, the upper one of the rods, and the spread-out eggs appear to be "more" than their counterparts, because of some basic law of perception. We think things are more complicated than that, and postulate that the younger child, when asked to make a quantitative comparison, CHOOSES to describe the things being compared in terms of "how far they reach, preferably upwards or in some other direction if necessary". That this description comes from a choice is clear from the fact that he can reliably tell which is "wider" or "taller", when it is not a question of which is "more". Indeed, if we asked the younger child to describe the situation in detail BEFORE asking which has more, he might say something like this:

(A) "There is a tall, thin column of water in the tall, thin jar and a short, wide column in the short, wide jar"

Actually, a four year old will not say anything of the sort. His syntactic structure will not be so elaborate, but more important, he is unlikely to produce that many descriptive elements in any one description. If we ask him "what is this", he might say any of "high glass", "almost full", high water", "round", etc., depending on what he imagines at the moment as a purpose for the question or the object. In any case, if we ask him for a description AFTER telling him we want to know which has more he will probably say the equivalent of:

(B) "There is a high column of water in the tall jar and a low column of water in the short jar"

To answer the question "which has more" one has to apply some process to the description of the situation. Once we have the second description (B) almost any process would choose the "high column of water". We still need a theory of what symbolic rules delete preferentially the horizontal descriptive elements from the first description (A).

Another possibility is that perhaps the child is misinterpreting "more" ; if he were strongly "motivated" by being thirsty or hungry he might give better answers. The experiments are, however, always careful about this, and one gets similar results if the eggs are replaced by candy actually to be eaten, or the water by a delicious beverage.

In suggesting that the child converts description "A" to description "B" we are proposing an analogy with ANALOGY! Is this too neat? Are we inventing this process for the child, who does not really do anything so simple? Certainly, we are making a mini-theory much simpler than what really happens. But what really happens is, we believe, correspondingly simpler than what most observers of children imagine is happening! The following kind of dialog is typical of what goes on in another situation that Piaget and his colleagues have studied, and illustrates explicitly the same striking kind of transformation of descriptions:

INTERVIEWER: How many animals are there?
CHILD: Five. Three horses and Two cows.
INTERVIEWER: Are there more horses or more animals?
CHILD: More horses. Three horses and two animals.

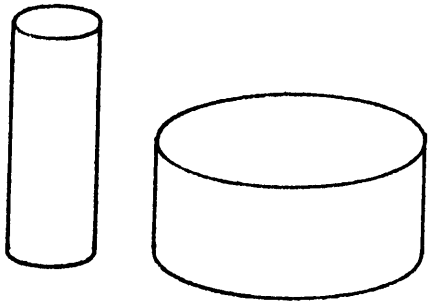
I: Now listen carefully:
ARE THERE MORE HORSES OR MORE ANIMALS?

I: What did I ask you?
C: Are there more horses or more animals?
I: What is the answer?
C: More horses.
I: What was the question again?
C: Are there more horses or more cows?

We explain this phenomenon on a similar basis; again the child has to make a comparison of quantity. He has learned that it is generally correct to do this by counting mutually exclusive classes and the worst thing is to count anything more than once. So he proceeds to describe the situation "correctly" for such purposes, and (in this frame) gets the correct answer.

It is often said that the pre-conservation child gets the answer wrong to "inclusion" questions. No. He gets the answer right. He gets the QUESTION wrong! Of course, inclusion comparisons are never natural, so we can agree with the child that these are silly "trick" questions, anyway.

Returning to judging "amount" by height alone, we must ask what "learning" process could cause a child to acquire this "false" idea? Our mini-theory begins not by trying to explain the particular fact (why the child says this about water or that about eggs) but to look for a general rule for comparing quantities that combines simplicity with widespread utility. Who is bigger; the child or his cousin? Stand back to back! How do you divide a bottle of coke between two glasses? By the level -- and generally this is fine because the glasses are identical. Finally, the child can afford to be wrong some of the time; this rule serves very well for many purposes and it would be hard to find a better one without taking a giant step.

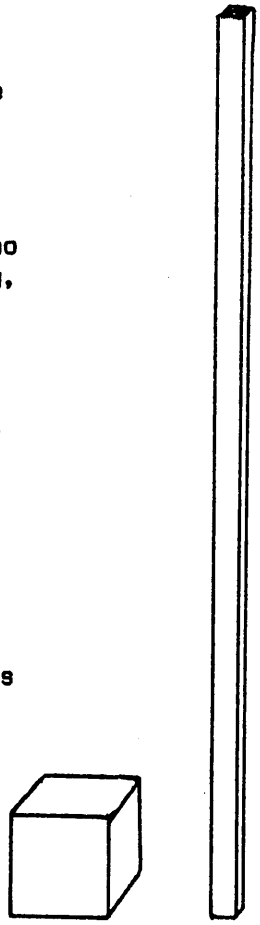


A confirmation of this is given by the children who judge there is more water in the thinner container of this pair.

Although fewer children will say this, the fact that there are ANY who do disproves the "appearance" theory, for one can hardly maintain that an unalterable law of perception is operating here.

Clearly the (heuristic) symbolic rule of vertical extent here overrides "perception" of dimensions.

One could make a case for the "appearance" theory, in the water-jar experiment as follows: The water is MUCH higher where it is high, but only somewhat wider where it is wide. The most plausible kind of comparison algorithm would look first for a unique term or quality upon which to base its decision -- as is easily found in (B). If there is none -- as in (A) -- then a subprocess has to make a "quantitative" comparison. But even this seems less symbolic than quantitative, for if we compare "much higher" with "somewhat thinner", the former will surely win! In any case, even adults can hardly believe that these two solids could have the same volume. So if the child were really faced with the problem of comparing quantitative dimensions, this would be almost impossible for him.



We next have to ask, how was this rule acquired, and how can we explain the transition to conservationist thinking? The simplest theory would assert that the child specifically learns each conservation (and, earlier, each comparison technique) as isolated pieces of knowledge. However, this theory is incomplete because it postulates some agent or specific circumstance responsible for the specific act of learning A more satisfactory kind of theory would let the child himself play the part of the "teaching agent" in the weak theory, and find his own strategies for making descriptions adequate for his problems.

Consider again the original conservation-of-number experiment. Suppose that we wanted to TELL the child how to behave. An authoritarian approach would shout at him: no, no, no, they are equal. But most teachers would prefer the gentler approach of explaining what he is doing wrong. One could say: "Yes, you are right, the eggs take up more space than the egg-cups so you could say that SPATIALLY there are more eggs; but NUMERICALLY there are still as many eggs as egg-cups."

We hope readers are objecting that no child of five will understand this little speech. Indeed, one can go a step further and say that the attempted lesson begs the entire question. The non-conservation child seems to lack a sharp distinction between "numerical" and "spatial". That's his problem! If he knew how to use the distinction well enough he would not need us to teach him about conservation. Our child has already a variety of concepts about quantities; we maintain that his problem is in knowing which to use when (instead of, or combined with others) in describing situations. His real problem is that he does not yet know good enough ways to describe his descriptors! If he learned how to describe his descriptors -- for example, to label some as "spatial" and some as "numerical" -- and if he could use these descriptions of descriptors to choose the appropriate ones, then the specific problem of learning conservations would dissolve away. As it should! For "conservation" is not a single thing, and "it"s development is typically spread out over several years as a child learns to deal with number, mass, volume, and other descriptive concepts.

Assuming a structure for classifying descriptions we can imagine an internal scenario, for the egg experiment, in which many descriptions are considered by a supervising process:

(1) Choose a kind of rule. Choices are
QUANTITATIVE RULES
HISTORICAL RULES
.
.

(2) QUANTITATIVE is chosen. Select a kind. Choices are
SPATIAL
NUMERICAL
.
.

(3) SPATIAL is chosen. Select a kind. Choices are
EXTENT implies more
SPARSENESS implies less
.
.

(4) Try EXTENT. The spread out eggs have more extent.
This means MORE.

(5) Test for coherence with other SPATIAL rules?
Try SPARSENESS. The eggs are sparser.
This means LESS!
An inconsistency. Reject or explain.

(3') Try NUMERICAL. Reject method
Try COUNTING
Too many to count.

(2') Reject choice of quantitative rules! Reject method
Try the next choice, HISTORICAL
.
.

When HISTORICAL is tried, one might first choose IDENTITY. Some eggs were moved, but none added or taken away.

Test for coherence with other HISTORICAL rules. Try REVERSIBILITY. The operation SPREADING-OUT is reversible. This means SAME!

We conclude that HISTORICAL seems consistent! This means SAME!

The same sort of scenario could be constructed for the water experiment; there the counting descriptions cannot be invoked, but instead other quantitative descriptions must be available. In each attempt, the description of the scene takes on a different form: the successful historical form will resemble

> "The water that was in the second jar
> is now in the third jar"

and "of course" it has the same amount as the first jar! Well! This gives the right answer, because he has obtained an adequate description. What kinds of processes must he have in order to do this. We have already proposed that he has a procedure for selecting descriptions; in what kind of environment could this operate? One kind of model would assume that the mature child's description is at first more elaborate, including both geometric and historical elements,

> "The amounts of water in the first and second jars were equal. The water that was in the second jar is now in the third jar. The water in the third jar is higher and thinner than that in the first jar."

The mature child, we might theorize, will eliminate elements from his description until there are no serious conflicts. This will yield a tentative answer, which he can maintain if he can explain away any problems that arise from reconsidering other details. Alternatively, one might imagine a process that begins with a very primitive description and elaborates it. But in any case, the process must have facilities for such functions as:

Choosing among the most plausible methods for answering the question. To apply a method he must bring the description into a useable form. For example, when he chooses a "history" method he suppresses some features of the spatial appearance. This means he must have a good classification of the different kinds of description elements.

The selection of the description involves common-sense knowledge. This, in a word, means that his entire cognitive structure is potentially engaged -- language, goals, logic, even interpersonal situational processes.

If the situation is at all novel, then any committment to "ignore" a class of elements may require a reason or "excuse", for conflicts in the original description that remain unexplained. A standard strategy is "compensation" -- knowing when it is reasonable to propose tradeoff between such pairs as height and width when manipulating fluids.

One cannot balance an arbitrary pair of dimensions, and particular pairs compensate only under suitable conditions. Ideas like "geometric property" are necessary, so that one isn't tempted to trade height with color, for example. What features of histories might correspond to such static properties as "spatial" and "numerical?"

Most important, the directing process in which the history of the situation wins out over the unusable geometric features, must exist and be debugged well enough that it can be relied upon! The child needs to have and trust the higher-order knowledge about which kinds of knowledge should have priority in each situation.

We have intentionally not specified the time scale of this scenario; some of it occurs over long periods, while some in the course of solving a particular problem. Furthermore, these conditions are still incomplete, yet our structure is already quite complicated. But so is the situation! Remember, our child can already carry on an intelligent conversation. This is not a good place to encourage the use of Occam's Razor. The time for that is when one has several good competing theories, not before one has any! It takes the child several years to work out all of this, and a theory that explained it away on too simple a basis might be therefore suspect! We do not, we repeat, want to explain the different conservations either on completely separate bases or by one unifying principle. We want to see it as the outcome of an improvement in the child's procedures for dealing with the variety of descriptions that he comes into possession of.

In the traditional "theories of learning" there was a tendency to ask

> "How does such-and-such a "response" become connected to such-and-such a "stimulus".

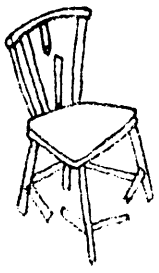
We now see that the proper questions are much more like

> "How can such-and-such a procedure be added to the descriptive or deductive systems"

4.2 LEARNING

A serious complaint about the heuristic programs of the past was their very limited ability to learn. This made them too inflexible to be useful except in very special situations. Over the years many direct attempts to construct "learning programs" led to very indifferent results. There is a close analogy, we feel, between this and the similar situation in the history of constructing psychological theories of learning.

If a child were to learn that 7+5=12 and 39+54=93 and, say, one hundred other such "responses", we would not agree he had learned to add. What is required is that he learn an appropriate procedure and how to apply it to numbers he has never used before. Another side of this "stimulus-response" problem: just as in the Analogy situation, the secret of learning often lies in the discovery of descriptions that emphasize the "essential" aspects of things or events, and omit or subjugate the "accidental" features. It would do us little good to remember that some particular thing happened in exactly a certain situation, since identical conditions never recur.



We do not need, or want, to remember the precise details of a broken chair, but we do want to remember that bad things happen when chairs have broken rungs -- for that is an essential difference between this and a usable chair. Indeed, the greater our knowledge and powers of observation, the more selective must be our choice of descriptions, because of the magnified problem of becoming lost in searching through networks of irrelevant details.

Finally, one hears complaints of the form "You programmed it to do that! It didn't learn it by itself"! There is a spectrum of degrees of autonomy in learning activities, and one wonders what are the distinctive features of importance between a child learning while playing by himself, discovering things under the shrewd guidance of an attentive instructor, prying a theory out of a mediocre textbook, and having it explained directly and concisely by a superb expositor.

It is tempting to try to disentangle this messy web of different phenomena. The appearance of an impossibly refractory problem in science is often the result of fusing fundamentally different problems (each of which may be relatively simple) when there is no common solution to the whole set. We think this is true of the many different ways in which programs can be said to learn. But despite this diversity there are important common themes. Most important of these, we feel, is the need for enough descriptive structure to represent the relation between learning situations and the concepts learned from them. Another theme comes from noticing that the kinds of learning we have found most difficult to simulate are those that involve a large stock of prior knowledge and analytic abilities. This leads us to propose for study very pure forms of the problem of handling divers kinds of knowledge -- prior to worrying about the problems of acquiring such knowledge. To separate out these strands we will consider at various points such not-entirely-separable ideas of "learning" as these:

Learning by development or maturation
Learning without description (by quantitative adaptation)
Learning by building and modifying description
Learning by being taught
Learning by Analogy
Learning by being told
Learning by being programmed
Learning by Understanding

4.3 Learning without description. "Incremental Adaptation".

There is a large literature concerned with clustering methods, scaling, factor analysis, and optimal decision theories, in which one find proposals for programs that "learn" by successive modifications of numerical parameters. An outstanding example of this is seen in one of the well-known programs of A. Samuel, that plays a good game of Checkers. Other examples abound; all perceptron-like "adaptive" machines, all "hill-climbing" optimization programs, most "stochastic learning" models using reinforcement. Some details can be found in the later chapters of our book, PERCEPTRONS.

The conclusions drawn in PERCEPTRONS are too technical to review here in detail, but we can describe the general picture that emerges. Within the classes of concepts that these machine can represent, that is, describe as rather literal "sums" of already programmed "parts" -- the learning abilities are effective and interesting. However, the descriptive powers of these quasi-linear learning schemes have such peculiar and crippling limitations that they can be used only in special ways. For example, we can construct, by special methods, a perceptron that could learn either to recognise squares, or to recognize circles. But the same machine would.probably not be able to learn the class of "circles or squares"! It certainly could not describe (hence learn to recognize) a relational compound like "a circle inside a square".

These limitations are very confining. It is true that such methods can be useful in "decision-making" and diagnostic situations where things are understood so poorly that a "weighted decision" is better than nothing! But we think it might be useful to put this in perspective by assigning it as an example of a new concept of TERMINAL LEARNING. The basic problem with this kind of "learning program" is that once the program has been run, we end up only with numerical values of some parameters. The information in such an array of numbers is so homogeneous and unstructured -- the "weight" of each "factor" depends so much on what other factors are also involved in the process -- that each number itself has no separate meaning. We are convinced that the results of experience, to be useful to "higher level processes", must be summarized in forms that are convertible to structures that have at least some of the characteristics of computer programs -- that is, something like fragments of program or descriptions of ways to modify programs. Without such capabilities, the simple "adaptive" systems can "learn" some things, to be sure, but they cannot learn to learn better! They are confined to sharpening whatever "linear separation" or similar hypotheses they are initially set to evaluate. A terminal learning scheme can often be useful at the final stage of a performance or an application, but it is potentially crippling to use it within a system that may be expected later to develop further.

One could make similar criticisms of another aspect of the adaptive "branch and bound" procedures found in most game-playing and other heuristic programs that follow the "look-ahead and minimax" tradition. Suppose that in analyzing a chess position we discovered that the KB-2 square is vulnerable to a rook-queen fork by moving a knight to that square. The traditional program returns a low numerical value for that position. What it really should do is return a description of why the position is bad. Then the previous plausible-move generator can be given a constructive suggestion: look for moves that add a defense to that square, or threaten one of the attacking pieces, etc. Subsequent exploration will discover more such suggestions. Eventually, these conditions may come to conflict logically, e.g., by requiring a piece to attack two squares that cannot both lie in its range. At this point, a deductive program could see that it is necessary to think back to an earlier position. Similarly, a description of that situation, in turn, could be carried further back, so that eventually the move generator can come to work with a knowledgeable analysis of the strategic problem. Surely this is the sort of thing good players must do, but no programs yet do anything much like it.

This argument, if translated into technical specifications, would say that if a chess program is to "really" analyse positions it must first have descriptive methods to modify or "update" its state of knowledge. Then it needs ways to "understand" this knowledge in the sense of being able to make inferences or deductions that help decide what experiments next to try. Here again, we encounter the problem of "common sense" knowledge since although some of this structure will be specific to chess, much also belongs to more general principles of strategy and planning.

People working on these homogeneous "adaptive learning" schemas (either in heuristic programming or in psychology) are not unaware of this kind of problem. Unfortunately, most approaches to it take the form of attempting to generalize the coefficient-optimizing schema directly to multi-level structures of the same kind, such as n-layer perceptrons. In doing so, one immediately runs into mathematical problems: no one has found suitably attractive generalizations (for n levels) of the kinds of convergence theorems that, at the first level, make perceptrons (for example) seem so tempting. We are inclined to suspect that this difficulty is fundamental -- that there simply do not exist algorithms for finding solutions in such spaces that operate by successive local approximations. Unfortunately we do not know how to prove anything about this or, for that matter, to formulate it in a respectably technical manner.

We could make similar remarks about most of the traditional "theories of learning" studied in Psychology courses. Almost all of these are involved with the equivalent of setting up connections with the equivalent of numerical coefficients between "nodes" all of the same general character. Some of these models have a limited capacity to form "chains of responses", or to cause some classes of events to acquire some control over the establishment of other kinds of connections. But none of these theories, from Pávlov on, seem to have adequate ability to build up processes that can alter in interesting ways the manner in which other kinds of data are handled. These theories are therefore so inadequate, from a modern computation-theory view, that today we find it difficult to discuss them seriously.

Trial and Error

Why, then, have such theories been so persistently pursued? Their followers were certainly not naive about these difficulties. One influence, we think, has been a pervasive misconception about the role of multiple trials, and of "practice", in learning. The supposition that repeated experiences are necessary for permanent learning certainly tempts one to look for "quantitative" models in which each experience has a small but cumulative effect on some quantity, say, "strength-of-connection".

> In the so-called "stimulus-sampling" theories we do see an attempt to show how certain kinds of one-trial learning processes could yield an external appearance of slow improvement. In this kind of theory, a response can become connected with many different combinations of stimulus features or elements as a result of a sampling processes. In each learning event a new combination can be tried and tested. This is certainly closer to the the direction we are pointing. However, we are less interested in why it takes so many trials to train an animal to perform a simple sequence of acts, and more interested in why a child can learn what a word means (in many instances) with only a single never-repeated explanation.

What is the basis for the multiple-trial belief? When a person is "memorizing" something he may repeat it over and over. When he practices a piece of music he plays it over and over. When we want him to learn to add we give him thousands of "exercises". When he learns tennis he hits thousands of balls.

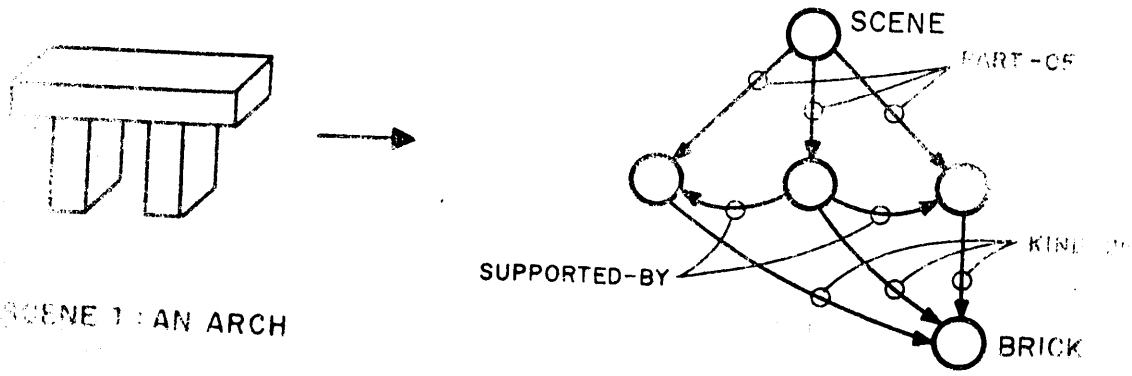
Consider two extreme views of this. In the NUMERICAL theory he moves, in each trial, a little way toward the goal, strengthening the desired and weakening the undesired components of the behavior. In the SYMBOLIC view, in each trial there is a qualitative change in the structure of the activity -- in its program. Many small changes are involved in debugging a new program, especially if one is not good at debugging!. It is not a matter of strengthening components already weakly present so much as proposing and testing new ones.

The external appearance of slow improvement, in the SYMBOLIC view, is an illusion due to our lack of discernment. Even practicing scales, we would conjecture, involves distinct changes in one's strategies or plans for linking the many motor acts to already existing sequential process-schema in different ways, or altering the internal structures of those schemas. The improvement comes from definite, albeit many, moments of conscious or unconscious analysis, conjecture, and structural experiment. "Thoughtless" trials are essentially wasted.

To be sure, this is an extreme view. There are, no doubt, physiological aspects of motor and other learning which really do require some repetition and/or persistence for reliable performance. Our point is that the extent of this is really quite unknown and one should not make it the main focus of theory-making, because that path may never lead to insight into the important structural aspects of the problem. In motor-skill learning, for example, it is quite possible one needs much less practice than is popularly supposed. It takes a child perhaps fifteen minutes to learn to walk on stilts. But if you tell him to be sure to keep pulling them up, it takes only five minutes. Could we develop new linguistic skills so that we could explain the whole thing? We might conjecture that the "natural athlete" has no magical, global, coordination faculty but only (or should we say "only"!) has worked out for himself an unusually expressive abstract scheme for manipulating representations of physical activities.

4.4 Learning by building descriptions

We can illustrate much more powerful concepts of learning in the context of a procedure developed by P. Winston to learn to recognize simple kinds of structures from examples. Like the SEE program of Guzman (which it uses as a sub-process) it works in the environment of childrens' building blocks. When presented with a scene, it first observes relations between features and regions, then groups these to find proposed structures and objects, and then attempts to identify them (using description-matching methods and the results of earlier learning experiences). Thus, the simple scene on the left is described by a network of abstract objects, relations, and relations between relations.

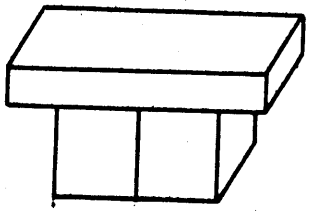


SCENE 1 : AN ARCH

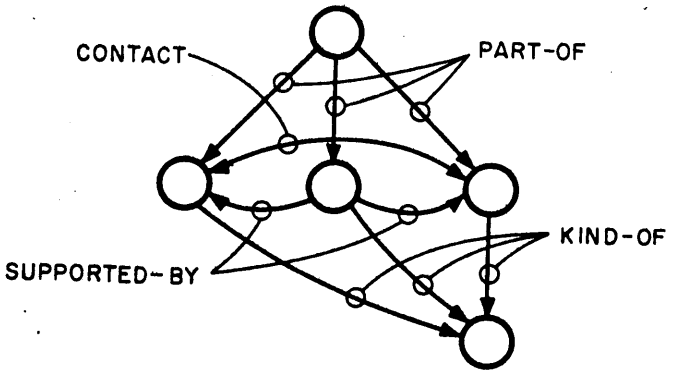
In this diagram, the heavy circles represent particular physical objects, the other circles represent other kinds of concepts, and the labels on the arrows represent relations. The program is equipped from the start to recognize certain spatial relations such as contact, support, and some other properties of relative position. We tell the machine that this is (an example of) an ARCH, and it stores the description-network away under that title.

Note that since these properties describe only relative spatial relations, the very same network serves to describe both of these figures, which are v sually quite different but geometrically the same.

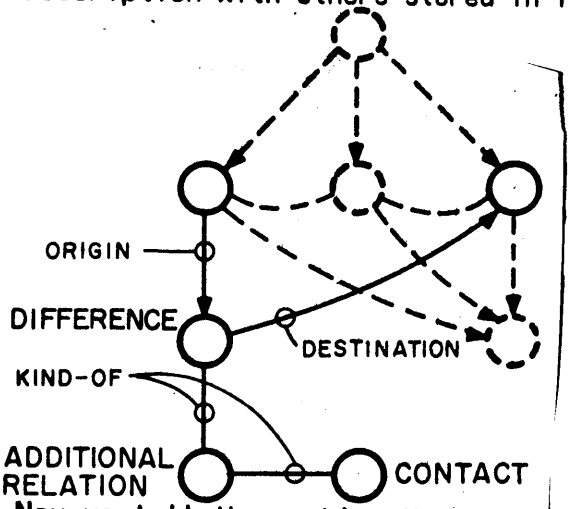
Next we present SCENE 2, to the left below, and the machine constructs the network shown to its right.



SCENE 2 : NOT AN ARCH

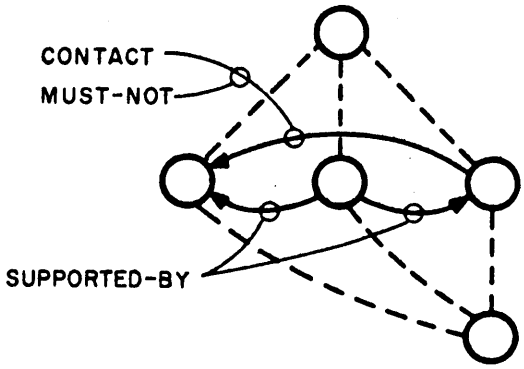


This differs from the network of scene 1 in only a few respects. If the program is asked what this structure "is," it will compare this description with others stored in its memory.

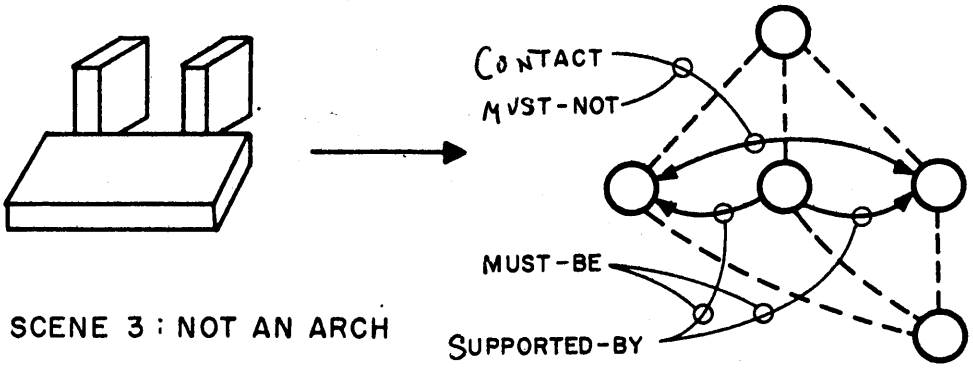


It has already networks for tables, towers, and a few other structures but, as one might expect, the structure it finds most similar is the ARCH description stored just a moment ago. So it tentatively identifes this as an arch. In doing this, it also builds a descriptive network that describes the difference between scene 1 and scene 2, and the difference is represented somewhat like this.

Now we tell the machine that scene 2 is NOT an example of an ARCH. It must therefore modify its description of "ARCH" so that structure 2 will no longer match the description, hence will no longer be "seen" as an ARCH. The method is to add a "rejection pointer" for the contact relation.

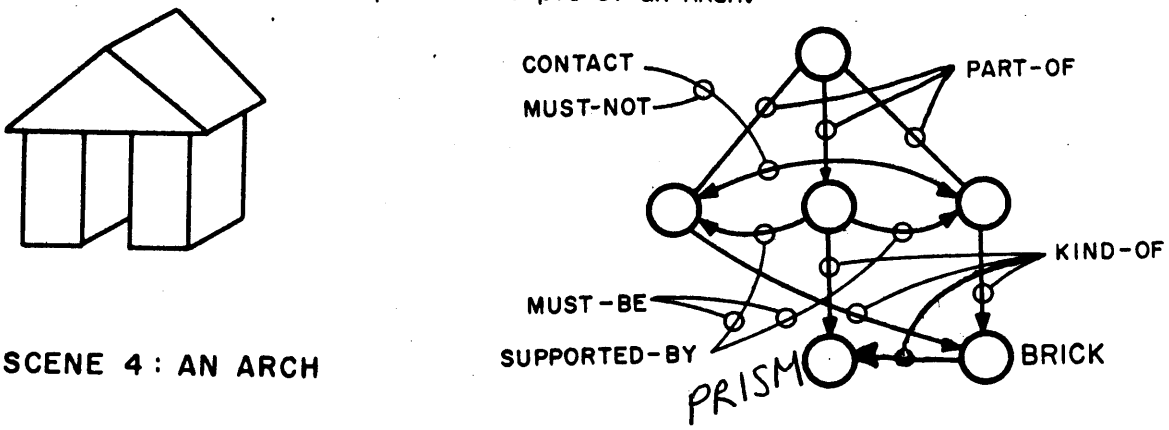


Now for the next example: we present scene 3 and assert that this, too, is not a ARCH. The most prominent difference, in this case, is that the new structure lacks the support relations



SCENE 3 : NOT AN ARCH

and the program for modifying "ARCH" now adds an "enforcement pointer" to the support relations. Finally, we present another example, scene 4, and assert that this is an acceptable example of an ARCH.



SCENE 4 : AN ARCH

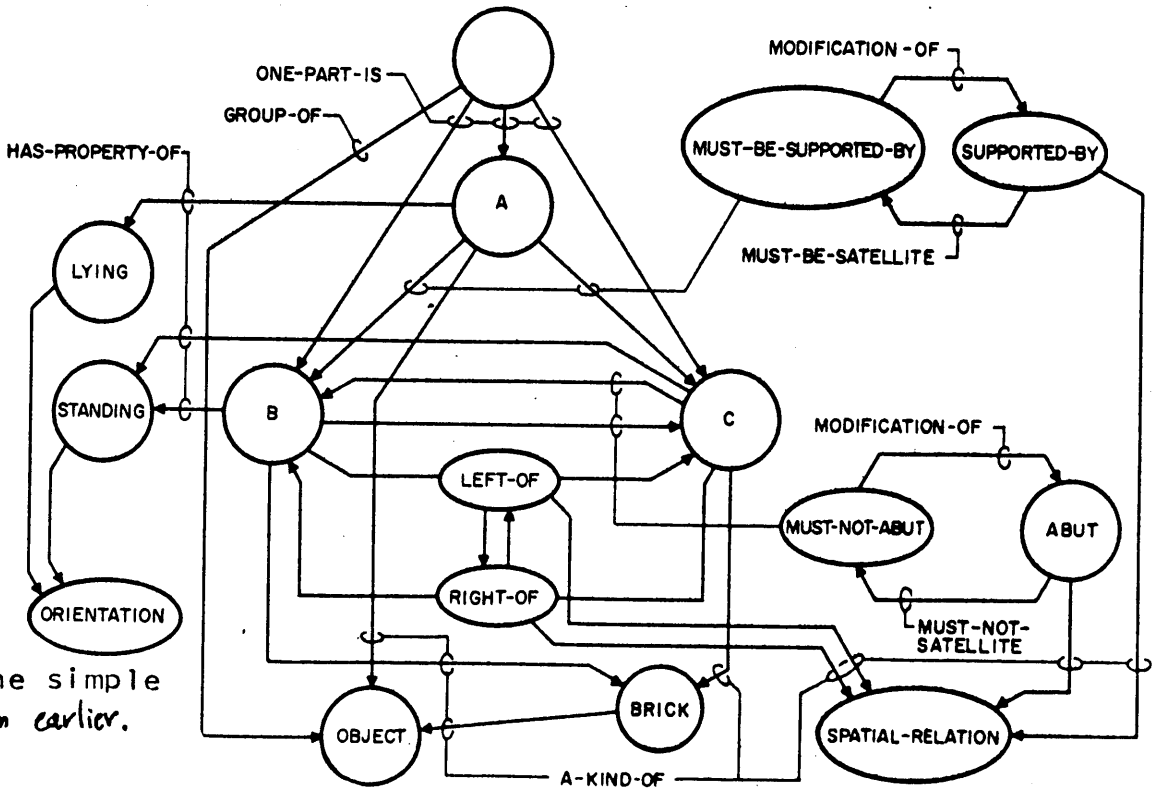
The most important difference, now, is the shape of the top block. The machine has to modify the description of "ARCH" so that the top block can be either a brick or a wedge. One strategy for this would be simply to invent a new class of objects -- "brick-or-wedge." This would be extremely "conservative", as a generalization or explanation. Winston's strategy is to look in memory for the smallest class that contains both bricks and wedges. In the machine's 'present state the only existing such classes are "prism" and "object" -- the latter is the class of all bodies, and includes the "prism" category, so the new description will say that the top object is a kind of prism. If we replaced the wedge by a pyramid, and told it that this, too, is an arch, it would have to change the top object-description to "object," because this is the smallest class containing "brick" and "pyramid." Now we can summarize the program's conclusion: an arch is

> "A structure in which a prismatic body is supported by two upright blocks that do not touch one another."

We have just seen how the program learns to identify correctly the membership of Scenes 1-4 as to whether they are ARCHes or not. As a consequence, it will probably "generalize" automatically to decide that

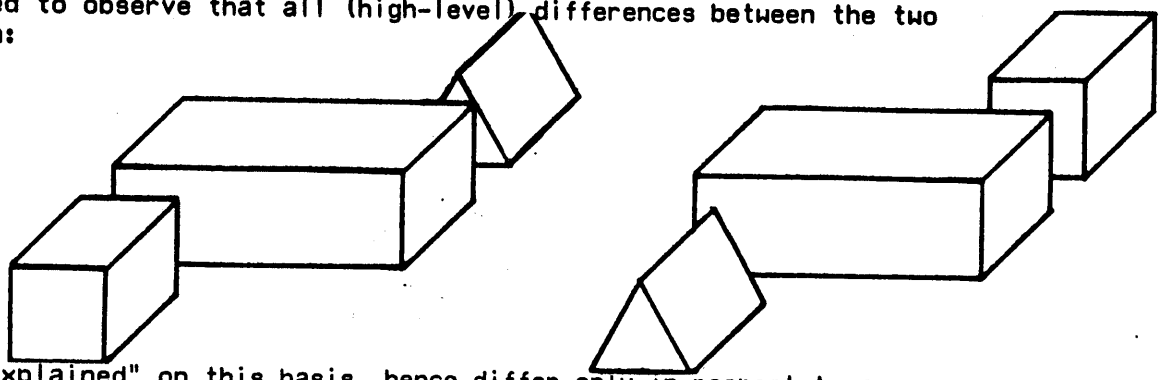
are also arches, because there are no "must-be-a..." enforcement pointers to either the supports or the top. Of course this judgement really depends on the machine's entire experience, i.e., on what concepts are already learned, and upon details of the comparison programs.

We have suppressed many interesting details of the behavior of Winston's program, especially about how it decides which differences are "most important". For example, the final form of the network for "ARCH" is more like:



than like the simple schemata shown earlier.

While on the subject, it should be noticed that within the network are represented relations between relations., as well as objects, properties and simple relations. There are important advantages to this when it comes to construction of the difference-descriptions. If the comparison program can be told that the difference between "IN-FRONT-OF" and "BEHIND," as well as that between "LEFT-OF" and "RIGHT-OF," can both be described in terms of "vertical axis symmetry," then it can be programmed to observe that all (high-level) differences between the two scenes in:



can be "explained" on this basis, hence differ only in respect to a vertical axis rotation. This is an example of a beautifully abstract form of description manipulation that, psychologically, would traditionally be attributed to something much more like an imaginary graphical rotation of the scene -- (as though there were no critically complicated problems in that reconstruction). In his thesis, Winston has only initiated such studies, and we know little about how far one can go with these methods. How much more structure would one need, to be able to learn, from examples, such concepts as symmetry? How difficult will it be to adapt such a system to learning new procedures, instead of structures? At first this might seem a huge step, but the ideas in the next section, on describing groups and repetitive structures, make the gap seem to become smaller.

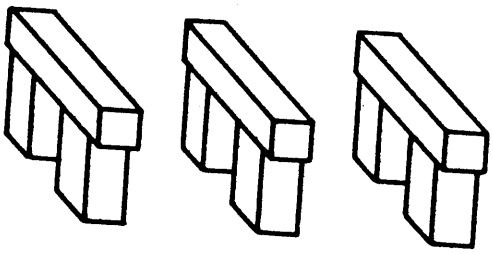
We shall see that the advantages of having a description for a "concept" (rather than just a competence) are absolutely crucial for further progress. These advantages include:

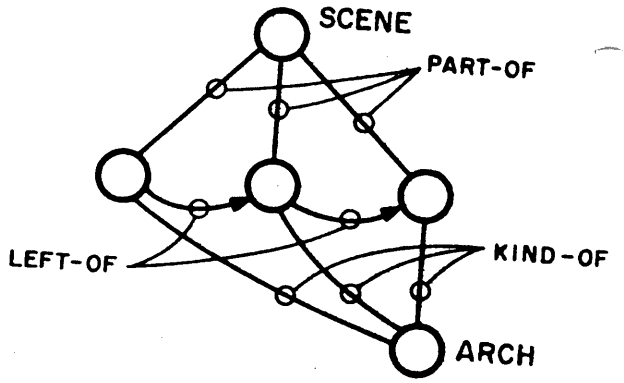
The ability to compare and contrast descriptions (as we shall see in section 4.6)

The ability to make deductions involving the concept, to adapt it to new situations.

Combining several descriptions to make new concepts.

An example of the latter: Every structural "concept" that Winston's program acquires is automatically incorporated within its own internal descriptive mechanisms. Thus, if the machine were presented with the nine-block scene below, before learning a concept of ARCH, it would have produce an impossibly complex and almost useless network of relations between the nine blocks. But after learning ARCH, it will now describe it in a much more intelligent way:





because its descriptive mechanisms proceed from local to global aggregates using as much available knowledge as it can apply. In doing this we encounter, now on a higher level, grouping problems very much like those we saw in our sketch of Guzman's SEE program, and in many cases one can adopt analogous strategies.

4.4 Learning by being taught

Imagine a child playing with a toy car and his blocks. He wants to build an interesting structure to play with. If the user of Winston's program were present, he could teach the child how to make an arch by the process just described, for it is not hard to convert the above description into a procedure for building arches. In fact, in chapter 5, we shall give a sketch of exactly how this can be done! This is precisely what Winograd's program does when it translates from the semantic analysis of an object-describing noun-phrase into a robot program for building with blocks! See chapter 5.

It is not necessary for the child to have a teacher, however. In the course of "playing" he can try experiments with the blocks and the car, and he can recognize "success" in either of these cases, among others:

a) He knows how to recognize an "ARCH" once it is built -- but does not know how to describe or to build it.

b) He has a functional play-goal: construct a road-problem for himself that is not too easy and not too hard -- such as an obstacle that requires two hands to overcome, but cannot be negotiated trivially with one hand.

In case (a) he he knows how to tell which structures are in the class. In case (b), while experimenting he will indeed find that Scene 1 is good, Scene 2 is impossible, Scene 3 is too easy, and Scene 4 (discovered as the simplest variant of the successful Scene 1) is also good. Here we get the same overall effect -- through the same mechanism -- yet in humanistic terms the behavior would be described much more naturally in terms of "exploratory," or "play," or "undirected" activity. The final result, if described in structural terms, is again

> "a structure in which an object is supported by
> two upright bricks that do not touch one another."

This is certainly not a perfect logical equivalent of the adult's idea of an arch; nor does it contain explicitly the idea of a surrounded passage or hole. Still, for the playing child's purposes, it would represent perhaps an important step toward formulation and acquisition of such concepts.

Again we have left alone some very important loose ends. We have concealed in the catch-all expressions "play" or "exploration" some supremely important conditions that must be fulfilled -- and at early stages of child development they won't be, and the things that are learned during "play" will be different!

> The child must already be equipped with procedures that have a decent chance of generating plausible structures.
>
> To do this, he must be able to describe to some extent why an experiment is unsatisfactory. If he cannot get his car between the supports, he must be able to think of moving the supports apart. This is not very hard, since pushing against the obstacle will sometimes do this.
>
> Since most experiments not carefully planned lead to useless structures, he has to have some ability to reconstruct a usable version of earlier and better situations after a disaster.

Without the teacher, it is unlikely that he will get good results after just four trials! He must have enough persistence in his goal-structure to carry through. To do this consistently would presuppose a good assessment of the problem's difficulty. Of course, if this is missing, he will find something else to do; not all play is productive!

Winston's program seems to be a reasonable model for kinds of behavior that would be plausible in, if not typical of, a child. The "concept" the program will develop, after seeing a sequence of examples, will depend very much on the examples chosen, on the order in which they are presented, and of course on the set of concepts the program has acquired previously. In many cases the experimenter may not get the result he wants; presenting examples in the wrong order could get the program (or child) irreparably off the track, and he might have to back up -- or perhaps restart at an earlier stage. We cannot expect our concept-learning programs to be foolproof any more than a teacher can expect his instructional technique always to work. The teacher always risks failure until he acquires correct insights into what has happened in the students' mind.

Of course there are many small but important details of how the program decides what to do at each step, which differences to give highest priority, which parts of the description networks should be matched, what explanations it should assign to the differences that are noticed.

Thus, in building with blocks, the relations "support" and "contact" ought to dominate properties of color, particular shapes and even other spatial relations like "in front of" or "to the right of."

In a different realm of activity, a different set of priorities might be essential, lest learning be slow or simply wrong. So, one can conclude that we must also develop intermediate structures in "learning to learn; a prerequisite to a child's (or machine's) mastery of mechanical structures will be some preparation in acquiring, grouping, and interrelating the more elementary descriptive structures to be used in assembling, comparing and modifying the representations to be used in the performance-level learning itself. This is exactly the conclusion we reached, in 4.1, about the requirements implicit in "maturation".

4.6 Analogy, again

Now we can return to our very first topic, solving problems involving analogies. In section 1.1 we proposed that the key idea would lie in finding ways to describe changes in descriptions. But this is exactly what happens in the program we have just described. When asked to describe a new scene situation, Winston's program makes use of the other descriptions it remembers, so that it can describe the scene in terms of already-learned concepts. Although we have not explained in detail how this is done, it is important to mention that the result of comparing two descriptions, in this system, is itself a description! Basically, the comparison works this way:

1. The two descriptions are "matched together",using various heuristic rules to decide which nodes probably correspond.

2. We create a new network, whose nodes are associated with pairs of nodes from the two descriptions that were matched. This is the skeleton of the comparison-description.

3. We associate with each node of this skeleton, a "comparison note" describing the correspondence. If the descriptions immediately local to two "corresponding" nodes are the same, the comparison-note is trivial. But if there are differences, (e.g., if one is a brick and the other a wedge) the "comparison note" describes this difference. Since these descriptive elements have the same format as by the original scene descriptions, one can operate upon them with the same programs. In particular, two difference-descriptions can be compared as handily as any other pair of descriptions.

Now we can apply this idea to the analogy problem. The machine must select that scene X (from a small collection of alternatives) which best completes the statement

A is to B as C is to X

That is, one must find how B relates to A and find an X that relates to C in the same way. Using the terminology

Diff[A:B]

to denote the difference-description-network resulting from comparing A with B, we simply compare the structures resulting from:

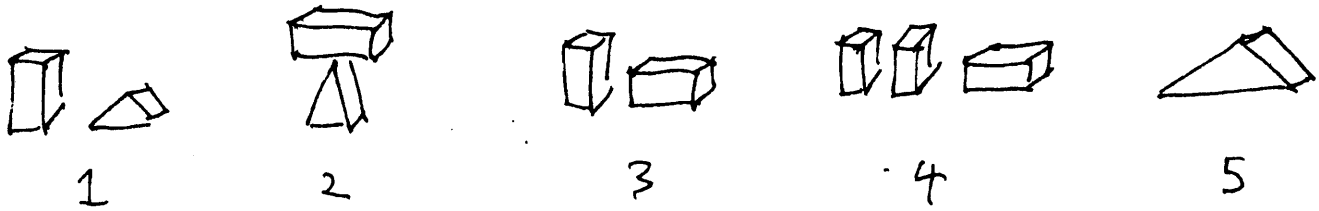
Diff[Diff[A:B] : Diff[C:X1]],

Diff[Diff[A:B] : Diff[C:X2]],

Diff[Diff[A:B] : Diff[C:X3]], etc.

Each of these summarizes the discrepancies within the "analogical explanations" for each corresponding possible answer. So to make the decision, we have to choose the "best" or "simplest" of these. We will not give details of how this is done; it is described in Chapter 7 of Winston's thesis. But note that some such device was needed already for the basic ability to identify a presented scene most closely with one of the descrptive models in memory. Thus the program must incorporate, in its comparison mechanism, conventions and priorities about such matters as whether the difference between Right and Left is to be considered simpler than the difference between Right and Above.

In this example

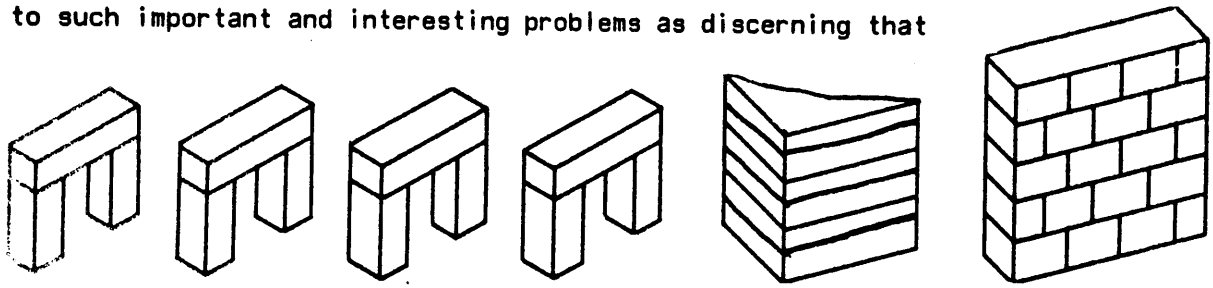


the machine chooses THREE as its answer, ONE as its second choice. In the slightly altered problem

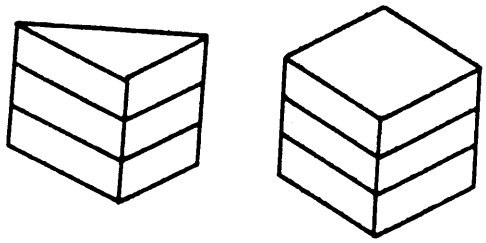
It chooses FOUR as its answer.

4.7 Grouping and Induction

The problem of recognizing or discerning grouping or clusterings of related things is another recurrent concern not only in Psychology, but also in statistics, artificial intelligence, theory of inductive inference; indeed, of science and art in general. Most studies of "clustering" have centered around attempts to adapt numerical methods from the theory of multivariate statistics to group data into subsets that minimize some formula which compares selected inter- and intra-group measures of relatedneess. But such theories are not easily adaptable to such important and interesting problems as discerning that

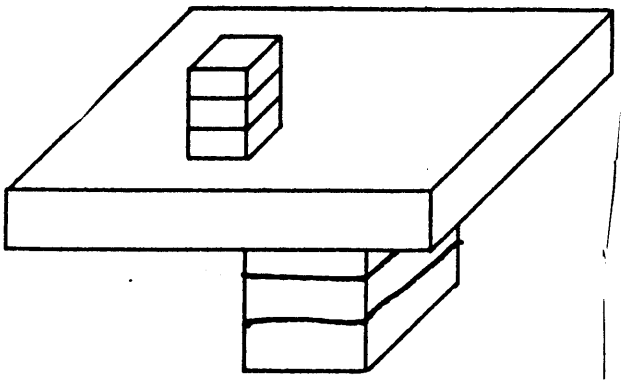


shows, not 12 + 6 + 20 = 38 objects, but "a row of arches, a tower of cubes, and a brick wall." More subtly, how do we "know" that one of these is three wedges while the other is three blocks? Visually, the lower objects in each tower are the same. These problems, too, can be treated by the same general methodology used in our approach to Analogy and to Learning of structures in scene-analysis.



> On many occasions we have been asked why the A.I. Laboratory is so concerned with special problems like machine vision, rather than more general approaches and problems about intelligence. In the early stages of a new science one proceeds best by gaining a very deep and thorough understanding of a few particular problems; that way one discovers important phenomena, difficulties, and insights, without which one risks fruitless periods of speculations and generalities. If the reader can see the present discussion in terms of general problems about induction and learning, the fruitfulness of the approach should speak for itself; we cannot imagine anyone believing the usefulness of these ideas is in any important way confined to description of visual or mechanical structures!

Take the groupings in the preceding figures and ask: "What qualities of the scene-descriptions characterize the intuitively acceptable groups." In some groups, like those shown above, it seems clear that the important feature is a CHAIN, say, of supported-by or in-front-of relations. In other cases it seems obvious that several objects show a common relationship to another. But no simple rules work in all situations.

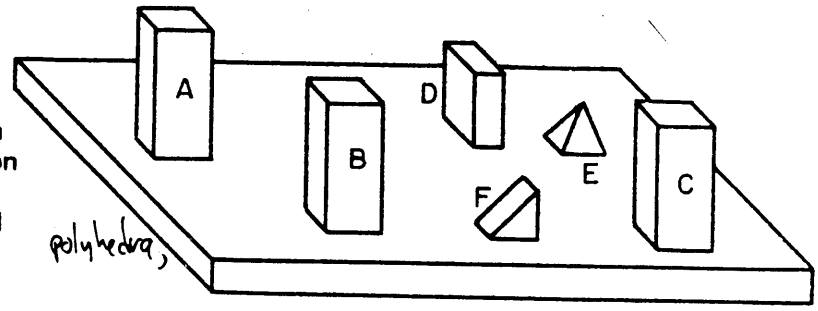


In this scene one does not usually see a single group or tower of seven blocks. Whether it is appropriate to describe this as "a seven-block stack," or as "a three-block stack supporting a plate that in turn supports a three-block stack," or as yet something else, depends on one's current purposes, orientations, or specifically on what grouping criteria are currently activated for whatever reason.

In some situations the discrepancies in the individual properties of the blocks should cause the grouping procedure to separate out the three-block stacks in spite of the fact that the suppport-relation chain continues through all seven blocks.

We next summarize some experiments along this line, again reporting results from P. Winston's dissertation. In Winston's grouping program, a generous hypothesis is followed by a series of criticisms and modifications.

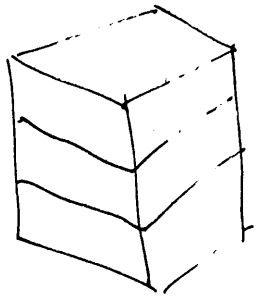
For example, when several objects have the same or very nearly the same description, they are immediately taken as candidates for a group. The blocks on this table are typical. All are bricks, all are standing, and all are supported by the board.

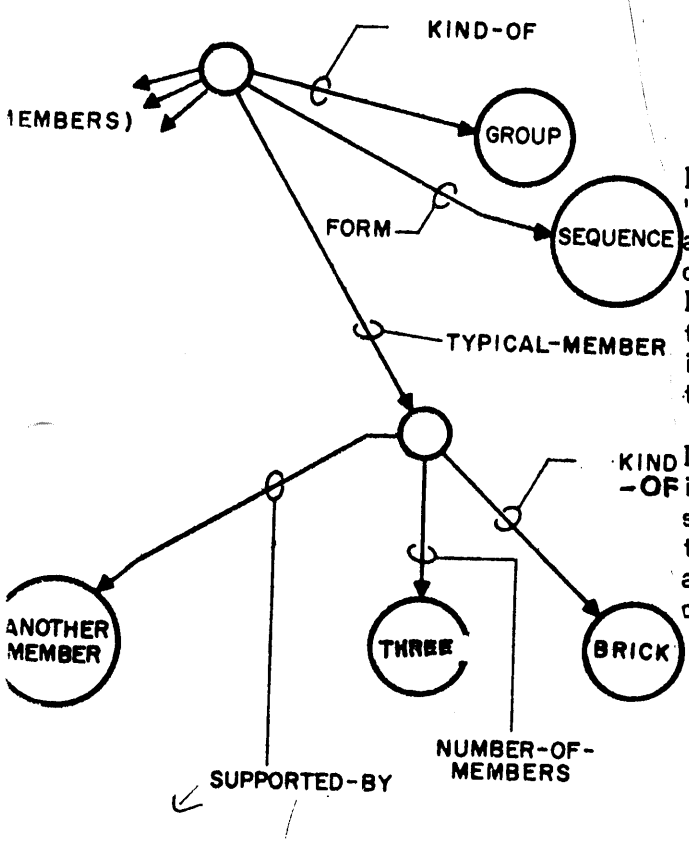


This proposal is then examined to eliminate objects which seem atypical, until a fairly homogeneous set remains. To do this, a program lists all relationships exhibited by more than half of the candidates in the set.

When the procedure operates, the first pass through the loop rejects E and F, mainly on the basis of shape. (Size is not considered in this pass because the six objects are too heterogenous for "size" to be put on the common-relationships list.) In a second pass, however, more than half the remaining objects share the "medium" size property, and block D is rejected, mainly because it does not share this property. So, finally, the procedure accepts only A B and C into the group. Obviously this is appropriate for some goals, but not others.

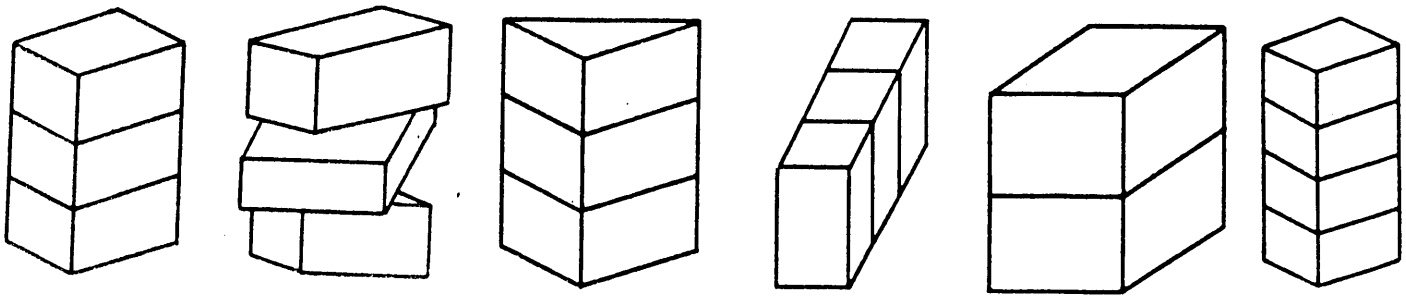
When grouping concepts are injected into the description framework, there can be unexpected and exciting consequences for other problems of induction. The figure below shows the network representation obtained when the grouping process operates on the description of this 3-block column.





Into the description is introduced a "typical member" to which is attributed the common properties discerned by the grouping procedure. In this case, chaining was used to form the group and the description includes the fact that there were three elements in the chain.

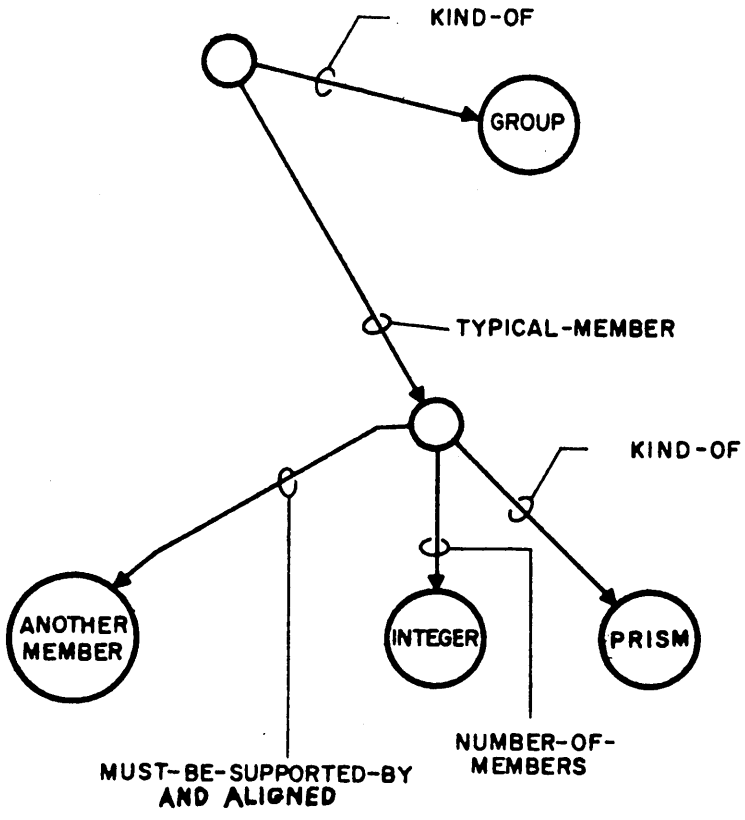
In a learning experiment, the program is presented with the depicted sequence of scenes shown below and is told that the first, third, and sixth are instances of "column" while the others are not.



COLUMN NOT A COLUMN COLUMN NOT A COLUMN NOT A COLUMN COLUMN

The second example causes the enforcement of a new pointer, "ALIGNED," a concept already available to the program that refers to the neat parallel alignment of edges. The third example tells the system that the typical member can be a wedge or a brick; the smallest common generalization here is "PRISM" so now a "column" can be any neatly piled stack of prisms. The fourth example changes "supported-by" to "must-be-supported-by"; the fifth, which is not seen as a group because it has only two elements, changes "one-part-is-a group" to "one-part-must-be a group".

The sixth and final example is of particular interest with respect to traditional induction questions. Comparison of it with the current concept of "column" yields a difference- description whose highest-priority feature is the occurrence of "FOUR" instead of "THREE," in the number-of-members property of the main group. What is the smallest class that contains both "THREE" and "FOUR?" In the program's present state, the only available superset is "INTEGER." Thus we obtain this description of "column", which permits a column to have any number of elements!



Is this too rash a generalization to make from so few examples? The answer depends on too many other things for the question to make much sense. If the program had already some concept of "small integer," it could call upon that. On a higher level we could imagine a program that supervised the application of any generalization about integers, and attaches an auxiliary "warning" pointer label to conclusions based on marginally weak evidence. We are still far from knowing how to design a powerful yet subtle and sensitive inductive learning program, but the schemata developed in Winston's work should take us a substantial part of the way.

Finally, we note that in describing a sequential group in terms of atypical member and its relations with the adjacent members of the chain, we have come to something not too unlike that in programming languages that use "loops," entry, and exit conditions. Again, a structure developed in the context of visual scene-analysis suggests points of contact with more widely applicable notions.

5.0 Knowledge and Generality

We now turn to another set of questions connected with our long-range goal of understanding "general intelligence". An intelligent person, even a young child, is vastly more versatile than the "toy" programs we have described. He can do many things; each program can do only one kind of thing. When one of our programs fails to do what we want, we may be able to change it, but this almost always requires major revisions and redesign. An intelligent human is much more autonomous. He can often solve a new kind of problem himself, or find how to proceed by asking someone else or by reading a book.

One might try to explain this by supposing that we have "better thinking processes" than do our programs. But it is premature, we think, to propose a sharp boundary between any of these:

Having knowledge about how to solve a problem,

Having a procedure that can solve the problem,

Knowing a procedure that can solve the problem!

In any case, we think that much of what a person can do is picked up from his culture in various ways, and the "secrets" of how knowledge is organized lie largely outside the individual. Therefore, we have to find adequate models of how knowledge systems work, how they are acquired by individuals, and how they interact both in the culture and within the individuals.

How can we build programs that need not be rebuilt whenever the problems we want to solve are slightly changed? One wants something less like ordinary computer "programming" and more like "telling" someone how to do something, by informal explanations and examples.

In effect, we want larger effects while specifying less. We do not want to be bothered with "trivial" details. The missing information has to be supplied from the machine's internal knowledge. This in turn requires the machine itself to solve the kinds of easy problems we expect people to handle routinely --- even unconsciously --- in everyday life. The machine must have both the kinds of information and the kinds of reasoning abilities that we associate with the expression "common sense".

There are differences of opinion about such questions, and we digress to discuss the situation. Artificial Intelligence, as a field of inquiry has been passing through a serious crisis of identity. As we see it, the problem stems from the tendency for the pursuit of technical methods to become detached from their original goals so that they follow a developmental pattern of their own. This is not necessarily a bad thing; many productive areas of research were born of such splits. Every discipline has had to deal with such situations and it has happened often in the study of human intelligence. Nevertheless, if one is interested in the particular goal of building a science of Intelligence, one has to be concerned with the use of resources both on the local scale of conserving one's own time and energy and on a global scale of watching to see whether the scientific community seems to be directing itself effectively. We suspect that there is now such a problem in connection with the studies of Mechanical Theorem Proving.

5.1 Uniform procedures vs. Heuristic Knowledge

As a first approximation to formulating the issues, consider a typical research project working on "automatic theorem proving". Schematically, the project has the form of a large computer program which can accept a body of knowledge or "data base," such as a set of axioms for group theory, or a set of statements about pencils being at desks, desks being in houses, and so on. Given this, the program is asked to prove or disprove various assertions. What normally happens is that if the problem is sufficiently simple, and if the body of knowledge is sufficiently restricted in size, or in content or in formulation, the program does a presentable job. But as the restrictions are relaxed it grinds to an exponential stop of one sort or another.

There are two kinds of strategy for how to improve the program. Although no one actually holds either policy in its extreme form and although we encounter theoretical difficulties when we try to formalize them, it nevertheless is useful to identify their extreme forms

The POWER strategy seeks a generalized increase in computational power. It may look toward new kinds of computers ("parallel" or "fuzzy" or "associative" or whatever) or it may look toward extensions of deductive generality, or information retrieval, or search algorithms -- things like better "resolution" methods, better methods for exploring trees and nets, hash-coded triplets, etc. In each case the improvement sought is intended to be "uniform" --- independent of the particular data base.

The KNOWLEDGE strategy sees progress as coming from better ways to express, recognize, and use diverse and particular forms of knowledge. This theory sees the problem as epistemological rather than as a matter of computational power or mathematical generality. It supposes, for example, that when a scientist solves a new problem, he engages a highly organized structure of especially appropriate facts, models, analogies, planning mechanisms, self-discpline procedures, etc., etc. To be sure, he also engages "general" problem-solving schemata but it is by no means obvious that very smart people are that way directly because of the superior power of their general methods -- as compared with average people. Indirectly, perhaps, but that is another matter: a very intelligent person might be that way because of specific local features of his knowledge-organizing knowledge rather than because of global qualities of his "thinking" which, except for the effects of his self-applied knowledge, might be little different from a child's.

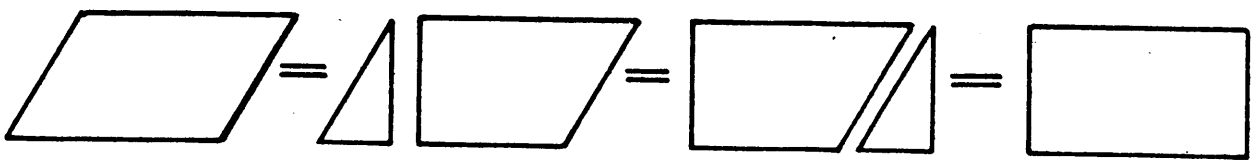
This distinction between procedural power and organization of knowledge is surely a caricature of a more sophisticated kind of "trade-off" that we do not yet know how to discuss. A smart person is not that way, surely, either because he has luckily got a lot of his information well organized or because he has a very efficient deductive scheme. His intelligence is surely more dynamic in that he has (somehow) acquired a body of procedures that guide the organization of more knowledge and the formation of new procedures, to permit bootstrapping. In particular, he learns many ways to keep his "general" methods from making elaborate but irrelevant deductions and inferences.

5.1.1 Successive Approximations and Plans

The mechanical theorem-proving programs fail unless provided with carefully formulated diets of data; either if given to little knowledge and asked advanced theorems, or given too much knowledge and asked easy questions. In any case, the contrast with a good mathematician's behavior is striking; the programs seem to have no "global" strategies. If a human mathematician is asked to find the volume of some object of unusual shape he will probably try to use some heuristic technique like:

1. cutting it into a sum of familiar shapes; or
2. enclosing it "tightly" in a familiar shape and
   try to find the difference-volume; or
3. transform, metrically, the space so that the shape
   becomes more familiar;
4. etc.

Thus, one would transform:



Now, in his final "proof" the heuristic principle that was used will not appear explicitly, even though its use was crucial. The three kinds of information in; The knowledge exhibited in the proof;

The knowledge used to find the proof;

The knowledge required to "understand" or explain the proof so that one can put it to other uses,

are not necessarily the same in extent or in content. The "Theorem Prover" systems have not been oriented toward making it easy to employ the second and the third kinds of knowledge. We have just given an example of how the second type of knowledge can be used.

The third kind of knowledge is exemplified by the following story about an engineer or physicist analyzing a physical system. First, he will make up a fairy-tale:

> "The system has perfectly rigid bodies, that can be treated as purely geometric. There is no friction, and the forces obey Hooke's law."

Then he solves his equations. He finds the system offers infinite resistance to disturbance at a certain frequency. He has used a standard plan; call it ULTRASIMPLE; and it produced an absurdity. But he does not reject this absurdity, completely! Instead, he says: "I know this phenomenon! It tells me that the "real" system has an interesting resonance near this frequency". Next, he calls upon some of his higher-order knowledge about the behavior of plan ULTRASIMPLE. Accordingly, this tells him next to call upon another plan, LINEAR, to help make a new model which includes certain damping and coupling terms.

Next, he studies this system near the interesting frequency that was uncovered by plan ULTRASIMPLE. He knows that his new model is probably very bad at other, far-away, frequencies at which he will get false phenomena because of the unaltered assumptions about rigidity; he has reason to believe these harmless in the frequency band now being studied. Then he solves the new second-order equations. This time he might obtain a pair of finite, close-together resonances of opposite phase. That "explains" the singularity in the simpler model. We abandoned one simple "micro-world" and adopted another, slightly more complicated and better adapted to the better-understood situation. This too may serve only temporarily and then be replaced by a more specialized set of assumptions for studying how nonlinearities affect the fine-structure of the resonances; a new plan, NONLINEAR or INELASTIC or THIRD-ORDER or DISCRETE, or whatever his third-type knowledge suggests.

One cannot overemphasize the importance of this kind of scenario both in technical and in everyday thinking. We are absolutely dependent on having simple but highly-developed models of many phenomena. Each model -- or "micro-world" as we shall call it -- is very schematic; in either our first-order or second-order models, we talk about a fairyland in which things are so simplified that almost every statement about them would be literally false if asserted about the real world. Nevertheless, we feel they are so important that we plan to assign a large portion of our effort to developing a collection of these micro-worlds and finding how to embed their suggestive and predictive powers in larger systems without being misled by their incompatibility with literal truth. We see this problem -- of using schematic heuristic knowledge -- as a central problem in Artificial Intelligence.

5.2 Micro-worlds and Understanding

In order to study such problems, we would like to have collections of knowledge for several "micro-worlds", ultimately to learn how to knit them together. Especially, we would like to make such a system able to extend its own knowledge base by understanding the kinds of information found in books. One might begin by studying the problems one encounters in trying to understand the stories given to young children in schoolbooks. Any six-year-old understands much more about each of such crucial and various things as

time	space	planning	explaining
causing	doing	preventing	allowing
failing	knowing	intending	wanting
owning	giving	breaking	hurrying

than do any of our current heuristic programs. Eugene Charniak, a graduate student, is now well along in developing some such models, and part of the following discussion is based on his experiences.

Although we might describe this project as concerned with "Understanding Narrative", -- of comprehending a story as a sequence of statements as read from a book -- that image does not quite do justice to the generality of the task. One has the same kinds of problems in:

making sense of a sequence of events one has seen or otherwise experienced (what caused what?)

watching something being built (why was that done first?)

understanding a mathematical proof (what was the real point, what were mere technical details?)

Many mental activities usually considered to be non-sequential have similar qualities, as in seeing a scene: why is there a shadow here? -- What is that? -- Oh, it must be the bracket for that shelf.

In any case, we do not yet know enought about this problem of common sense. One can fill a small book just describing the commonsense knowledge needed to solve an ordinary problem like how to get to the airport, or how to change a tire. Each new problem area fills a new catalogue. Eventually, no doubt, after one accumulates enough knowledge, many new problems can be understood with just a few additional pieces of information. But we have no right to expect this to happen before the system contains the kind of breadth of knowledge a young person attains in his elementary school years!

We do not believe that this knowledge can be dumped into a massive data base without organization, nor do we see how embedding it in a uniformly structured network would do much good. We see competence as emerging from processes in which some kinds of knowledge direct the application of other kinds in which retrieval is not primarily the result of linked associations but rather is computed by heuristic and logical processes that embed specific knowledge about what kinds of information are usually appropriate to the particular goal that is current.

We already know some effective ways to structure logically deep but epistemologically narrow bodies of knowledge, as the result of research on special purpose heuristic programs like MACSYMA, DENDRAL, CHESS, or the Vision SystemTo get experience with broader, if shallower, systems we plan to build up small models of real world situations; each should be a small but complete heuristic problem solving system, organized so that its functions are openly represented in forms that can be understood not only by programmers but also by other programs. Then the simple-minded solutions proposed by these mini-theories may be used as plans for more sophisticated systems, and their programs can be used as starting points for learning programs that intend to improve them.

In the next section we will describe a micro-world whose subject matter has a close relation to the vision world already described. Its objects are geometric solids such as rectangular blocks, wedges, pyramids, and the like. They are moved and assembled into structures by ACTIONS, which are taken on the basis of deductions about such properties as shape, spatial relations, support, etc. These interact with a base of knowledge that is partly permanent and partly contingent on external commands and recent events.

5.3 Winograd's BLOCKS World

Note: Sections 5.3 through 5.6 are largely adapted from Terry Winograd's Thesis, but he is not responsible for the oversimplifications and reinterpretations.

For developing and demonstrating his ideas about understanding natural language, Terry Winograd [ref.] needed a micro-world in which to carry on a discourse containing statements, questions and commands. In this world we pretend we are talking to a very simple type of robot, like the ones being developed in AI projects at Stanford and MIT. The robot has an arm and an eye. It can look at a scene containing toy objects and can move them with its hand. Winograd did not try to use an actual robot or to simulate it in great physical detail. His "robot" exists only as a display on the CRT scope attached to the computer.

A subject for such a discourse needs a certain amount of structure to support interesting description and manipulation problems. The BLOCKS WORLD has OBJECTS, RELATIONS (and properties) of the objects, ACTIONS that can be performed, and GOALS --- descriptions of states of the world that one might want to acheive.

5.3.1 OBJECTS

In Winograd's model, the robot (named :SHRDLU) has a hand (:HAND) which manipulates objects on a table (:TABLE) that has on it a box (:BOX). The rest of the physical objects are toys --- mainly blocks and pyramids We give them the names :B1, :B2, :B3, etc. Any symbol beginning with ":" represents a specific object.

Built into this world are some concepts we will use to describe these objects and their properties. We represent them in a tree:

```
                ^TABLE
                ^BOX        ^BLOCK
^PHYSOB---^MANIP----^BALL
^ROBOT      ^HAND       ^PYRAMID
^PERSON     ^STACK
^PROPERTY----------^COLOR
                    ^SHAPE
```

The symbol PHYSOB stands for "physical object" and MANIP for "manipulable object" (i.e. something the robot can pick up). Using the concept IS to mean "has as its basic description," we can write assertions like

(IS :SHRDLU ROBOT) (IS :HAND HAND) (IS :B5 PYRAMID)

For other, less basic properties we can write attribute-value statements like (MANIP :B5) and (PHYSOB :TABLE). Shape and color are handled with simple assertions like (COLOR :BOX WHITE) and (SHAPE :B5 POINTED). The possible shapes are ROUND, POINTED, and RECTANGULAR, and the colors are BLACK, RED, WHITE, GREEN and BLUE. The property names themselves can be treated as objects, so we can make such assertions as (IS BLUE COLOR) and (IS RECTANGULAR SHAPE).

Size and location are more complex, as they depend on the way we choose to represent physical space. We adopted a standard three-dimensional coordinate system and make the simplifying assumption that objects are not allowed to rotate, and therefore always keep their orientation aligned with the coordinate axes. We can represent the position of an object by giving the coordinates of its front lower left-hand corner, and its size by giving three dimensions, as in (AT :B5 (400 600 200)), and (SIZE :B5 (100 100 300)).

5.3.2. Relations

Since we are interested in building structures with the objects around in the scene, one of the most important relations is SUPPORT. The initial data base contains assertions about all of the support relations in the initial scene, like (SUPPORT :B1 :B2). Every time an object is moved, a PLANNER "antecedent theorem" removes the old assertion about what was supporting it, and puts in the correct new one. See 5.3.3. An "antecedent theorem" can be regarded as a sort of demon that watches for some sort of event to happen, and then takes a suitably programmed action. The Blocks World uses a notion of "support" in which an object is supported by whatever is directly below its center of gravity, at the level of its bottom face. Therefore, one object can support several others, but there is only one thing supporting it. Of course, this is an extreme simplification since it does not recognize that a simple bridge is supported.

The assertion (CLEARTOP X) will be in the data base if and only if there is no assertion (SUPPORT X Y) for any object Y. Whenever an object is moved, a procedure checks to see if the CLEARTOP status of any object has changed, and if so the necessary erasures and assertions are made.

Information about what is contained in the box is also kept current by an antecedent theorem concerned with the property CONTAIN. The property GRASPING is used to indicate what object (if any) the robot's hand is currently grasping.

Another relation is the PART relation between an object and a stack. We can give a name to a stack, such as :S1, and assert (PART :B2 :S1). Relations using the symbols RIGHT, BEHIND and ABOVE represent the difference in coordinate axes for X, Y and Z respectively. The symbol ON is used to represent the transitive closure of SUPPORT. That is, Z is ON A if A supports B, B supports C,...supports Z.

The measurements of LENGTH, WIDTH and HEIGHT are contained in the SIZE assertions and (HEIGHT X) evaluates to the height of whatever object the variable X is bound to. If SIZE is used in this way, it returns a measure of "overall size" to be used for comparisons like "bigger." Currently, it returns the sum of the X, Y and Z coordinates, but it could be easily changed to be more in accord with human psychology. In order to compare measurements, we have the relation MORE. The sentence ":B1 is shorter than :B2" is equivalent to the assertion (MORE HEIGHT :B2 :B1).

OWN relates a person to any object. Knowledge about what the human user owns is gathered from his statements. The semantic programs can use statements about owning to generate further PLANNER theorems which are used to answer questions about what :FRIEND (the human operator) owns and make deductions needed to carry out commands involving owning. The current system contains only token examples of such properties unrelated to the microworld of blocks.

5.3.3 Actions

The only events that can take place in our world are actions taken by the robot in moving its hand and manipulating objects. At the most basic level, the only actions which "really" occur are !MOVETO!, !GRASP! and !UNGRASP!. These are the actual commands sent to the display routines and, theoretically, to a physical robot system.

> To explain how the actions are programmed, in Winograd's system, we need to know a little about the PLANNER language of Carl Hewitt. Briefly, PLANNER has several ways for handling information of the form "A implies B", customarily called "theorems". In one form, the "consequent" form, it is interpreted roughly as "If you want something of the form B, make A a subgoal". In another, the "antecedent" form, it means "if something of the form A occurs, then deduce B and add it to the data base". Still another form of theorem can erase statements, such as support assertions that ought to be excised automatically when one of the participating objects is moved.

The result of calling a consequent theorem to achieve a goal requiring motion, like (PUTON :B3 :B4), is a plan -- a list of instructions using the three elementary functions. !MOVETO! moves the hand and whatever it is grasping to a specified position. !GRASP! sets an indicator that the grasped object is to be moved along with the hand, and !UNGRASP! unsets it. The robot grasps by moving its hand directly over the center of the object on its top surface, and turning on a "magnet." It can do this to any manipulable object, but can only grasp one thing at a time. Using these elementary actions, we can build a hierarchy of actions, including goals that may involve a whole sequence of deductions and actions, like STACKUP which causes the construction of a whole stack of blocks).

Inside the system are another set of "conceptual actions" MOVEHAND, GRASP and UNGRASP, and corresponding consequent theorems to achieve them. There is a significant difference between these and the functions listed above. Calling the function !MOVETO! actually causes the hand to move. On the other hand, when PLANNER evaluates a statement like:

(GOAL (MOVEHAND (600 200 300)) (USE tc-MOVEHAND))

nothing is actually moved. Translation: If your goal is to move the hand to (600, 200, 300), use the advice in the tc-MOVEHAND theorem to achieve this goal. The "USE" clause is a feature in PLANNER to allow the insertion of advice on how to achieve goals, etc., in any assertion or theorem. Here, the tc-MOVEHAND theorem creates a plan to do the motion, but if this move would cause us to be unable to achieve a goal at some later point, the PLANNER backup mechanism will automatically erase it from the plan. The robot plans its entire sequence of actions before actually moving anything, trying if necessary all of the recommended means it has to achieve its goal. We do not have space to explain PLANNER's backup system in complete detail; it is described in Hewitt's thesis [ref.], and the following sections show roughly how it provides automatic tree searching when necessary, under the control of the "USE" recommendations attached to the theorems in the data base.

These theorems also do some checking to see if we are trying to do something impossible. For example, MOVEHAND makes sure the action would not place one block where there is already an other, and UNGRASP fails unless something will support the object it wants to let go of.

These are the basic objects, relations and actions in the blocks world. But a micro-world also needs concepts about intentions, processes, strategies, etc. We next describe the strategies programmed into the system for obeying commands about complex goals and for answering questions about its performance and about its intentions.

5.3.4 Carrying Out Commands

Some theorems, like tc-GRASP, are complex, as they can cause a series of actions. The following program gives simplified definitions of various PLANNER theorems. Using these definitions, we will be able to follow the system through a complex action in detail.

```
tc-CLEARTOP X
            GO      (COND ((GOAL (SUPPORT X _Y))
                           (GOAL (GET-RID-OF Y) (USE tc-GET-RID-OF))
                           (GO GO))
                          ((ASSERT (CLEARTOP X))))

tc-GET-RID-OF X
                  (OR
                   (GOAL (PUTON X :TABLE)(USE tc-PUTON))
                   (GOAL (PUTON X Y)(USE tc-PUTON)))

tc-GRASP X
                  (GOAL(MANIP X))
                  (COND ((GOAL (GRASPING X)))
                          ((GOAL (GRASPING _Y))
                           (GOAL (GET-RID-OF Y) (USE tc-GET-RID-OF))))
                          (T))
                  (GOAL (CLEARTOP X) (USE tc-CLEARTOP))
                  (SETQ _Y (TOPCENTER X))
                  (GOAL (MOVEHAND Y)
                          (USE tc-MOVEHAND))
                  (ASSERT (GRASPING X))

tc-PUT X
                  (CLEAR Y (SIZE X) X)
                  (SUPPORT Y (SIZE X) X)
                  (GOAL (GRASP X) (USE tc-GRASP))
                  (SETQ _Z (TCENT Y (SIZE X)))
                  (GOAL (MOVEHAND Z)  (USE tc-MOVEHAND))
                  (GOAL (UNGRASP)  (USE tc-UNGRASP)))

tc-PUTON X Y
                  (NOT EQ X Y))
                  (GOAL (FINDSPACE Y $E (SIZE X) X _Z)
                          (USE tc-FINDSPACE tc-MAKESPACE))
                  (GOAL (PUT X Z)  (USE tc-PUT))
```

Let us trace, for example, the meaning of PUTON. The first clause

(PUTON X Y)

is the "pattern" of the goal. X and Y are variables to be matched. If the goal has this form, then these variables are bound to what they matched and

(NOT (EQ X Y))

checks for the (impossible) situation of trying to put a block on itself. If this "failure" occurs then the current goal will be abandoned. This means thatPLANNER will back up --- reconstruct the situation at the most recent previous variable-binding decision. For example, in this case, the system must have been looking for a place to put the block X, and stupidly decided to put it on X! Now it must make another choice, and presumably this time Y will be bound to a different, more sensible location. So this time tc-PUTON will pass the (NOT (EQ X Y)) test and go on to the next step, which is to create a subgoal:

(GOAL (FINDSPACE Y $E (SIZE X) X _Z)
(USE tc-FINDSPACE tc-MAKESPACE))

which says to try to find a space on Y big enough for X, ignoring space currently occupied (possibly) by X. The location resulting from success of this goal is then bound to Z. Again, if the goal fails, we would back up, but the program makes two recommendations for how to find such a place. tc-FINDSPACE says to try to find a space already there; if this fails then tc-MAKESPACE says to try to make such a space.

(GOAL (PUT X Z) (USE tc-PUT)))

Assuming that this succeeds, then try to use tc-PUT to actually put X in that location Z.

With this explanation we can follow what happens if PLANNER tries the goal:

(GOAL (GRASP :B1) (USE tc-GRASP))

The theorem tc-GRASP checks to make sure :B1 is a graspable object by looking in the data base for (MANIP :B1). If the hand is already grasping the object, it has nothing more to do. If not, it must first get the hand to the object. This may involve complications -- the hand may already be holding something, or there may be objects sitting on top of the one it wants to grasp. In the first case, it must get rid of whatever is in the hand, using the the command GET-RID-OF.

The easiest way to get rid of something is to set it on the table, so tc-GET-RID-OF creates the goal (PUTON X :TABLE), where the variable X is bound to the object the hand is holding. Then tc-PUTON must in turn find a big enough empty place to set down its burden, using the command FINDSPACE, which performs the necessary calculations, using information about the sizes and locations of all the objects. tc-PUTON then creates a goal using PUT, which calculates where the hand must be moved to get the object into the desired place, then calls MOVEHAND to actually plan the move. If we look at the logical structure of our active goals at this point, assuming that we want to grasp :B1, but were already grasping :B2, we see:

```
(GRASP :B1)
      (GET-RID-OF :B2)
        (PUTON :B2 :TABLE)
              (PUT :B2 (453 201 0))
                (MOVEHAND (553 301 100))
```

After moving, tc-PUTON calls UNGRASP, and we have achieved the first part of our original goal -- emptying the hand. Now we must clear off the block we want to grasp. tc-GRASP sets up the goal:

```
(GOAL (CLEARTOP :B2) (USE tc-CLEARTOP))
```

This is a good example of the double use of PLANNER goals to both search the data base and carry out actions. If the assertion (CLEARTOP :B1) is present, it satisfies this goal immediately without calling the theorem. However if :B1 is not already clear, this GOAL statement calls tc-CLEARTOP which takes the necessary actions. Then tc-CLEARTOP will try to GET-RID-OF the objects on top of :B1. This will in turn use PUTON, which uses PUT. But tc-PUT may have more to do this time, since the hand is not already grasping the object it has to move. It therefore sets up a goal to GRASP the object, recursively calling tc-GRASP again.

And so on! To answer questions about the past, the BLOCKS programs remember parts of their subgoal tree by creating objects called events. The system does not remember small, specific steps like MOVEHAND, but only larger goals like PUTON and STACKUP. The time of events is measured by a clock which starts at 0. It is incremented by 1 every time any motion occurs, creating a new event that combines the original goal statement with an arbitrary name, the starting time, ending time, and "reason" for each event. The reason is the name of the event nearest up in the subgoal tree which is being remembered. (The reason for goals called by the linguistic part of the system is "because you asked me to").

A second kind of memory keeps track of the actual physical motions of objects, noting each time one is moved, and recording its name and the location it went to. This list can be used to establish where any object was at any past time.

When we want to pick up block :B1, we can say: (GOAL(PICKUP :B1)), and it is interpreted as a command. We can also ask "Did you pick up :B1?", since when the robot picked it up, an assertion like (PICKUP E2 :B1) was stored in the data base. If PLANNER evaluates PICKUP X :B1)) it will find the assertion, binding the variable X to the event name E2. Since the property list of E2 gives its starting and ending times, and its reason, this is sufficient information to answer most questions.

5.4 Sample Dialog with Winograd's Language Understanding System

To make a useful micro-world one needs not only objects, properties and actions, but also goals, processes, and simplified versions of general concepts like time, space, and intention. The Blocks World was designed by Terry Winograd to serve as a subject domain for the development of a language understanding system. This system is radically more ambitious than earlier attempts to deal with natural language in several respects. It is as much, or more, concerned with the problem of representing the meanings in a discourse than with the grammatical structure; Winograd's thesis asserts that the interactions between meanings and sysntactic structures are so intimate that it is a bad idea to try to draw a definite boundary between the procedures that "parse" a sentence and the procedures that are to be used to draw logical inferences about the meanings carried in the discourse.

Winograd's system is described in his thesis at several levels of detail and it is impossible to do justice to it in any short summary. Our strategy will be first to present an example of its performance in the blocks world context, and then to discuss some of the mechanisms that are involved in attaining this performance. In the following dialog the system carries out actions and answers questions about a simple scene containing a table, hand, and box, and several blocks and pyramids, as shown in Fig. The lines in lower case were typed in by a person, and the lines in upper case are the computer's response. Indented lines are comments describing some of the features being exhibited.

The dialog was carried out in real time with the response of the "robot" displayed on a CRT screen. Figures are included to show the state of the scene at various points. It is important to read the dialog with some skepticism! While the apparent linguistic fluency is to an extent genuine, the vocabulary and semantic base is very thin, and the conversation works because the human operator stays closely within the boundaries of the system's understanding.

1. pick up a big red block.
------- OK.

The system answers "OK" when it carries out a command. In order to pick up the red block, it had to clear it off by finding a space for the green one and moving the green one away.

2. grasp the pyramid.
------- I DON'T UNDERSTAND WHICH PYRAMID YOU MEAN.

There are three pyramids in the scene, so the program does nothing.

3. find a block which is taller than the one you are holding and put it into the box.

The system has to figure out what is meant by "one" and "it", as well as using deduction to decide which block it is holding.

------ BY "IT", I ASSUME YOU MEAN THE BLOCK WHICH IS TALLER THAN THE ONE I AM HOLDING. (Printed out to keep us informed.)
------- OK.

The hand has to set down the red block to pick up the taller blue one. Figure 4 shows the result of the system's manipulations so far.

4. what does the box contain?
------- THE BLUE PYRAMID AND THE BLUE BLOCK.

5. what is the pyramid supported by?
Earlier the system responded to "the pyramid" by saying "I don't understand which pyramid you mean."
------- THE BOX. Now, it assumes that we mean the one just mentioned.

6. how many blocks are not in the box?
Negatives such as "not" and "no" are handled whether they appear in clauses or noun groups (such as "no boxes").
----- FOUR OF THEM.

7. is at least one of them narrower than the one which i told you to pick up?
The phrase "at least one of them" must be analyzed to refer back to the previous question, and "the one which" must also refer to blocks. Noun groups can have complicated numbers such as "at least one", "more than three", "exactly seven", etc. The memory of past events is used to deduce which block is "the one which I told you to pick up". Notice the complicated grammatical structure in which "the one" is actually the object of a clause embedded two levels deeper.
------- YES, THE RED CUBE.

8. is it supported? Passives can be used without a "by ..." phrase.
------- YES, BY THE TABLE.

9. can the table pick up blocks?
------- NO. The semantic rules make this a nonsensical question which therefore must be false.

10. can a pyramid be supported by a block?
------- YES. The deductive system finds an actual example, so it knows this is possible.

11. can a pyramid support a pyramid?
------- I DON'T KNOW.
The question makes sense, but there is no example in the scene, so there is no easy way to tell without being able to examine the robot's programs.

12. stack up two pyramids.
------- I CAN'T. The robot tries and fails.

13. the blue pyramid is mine.
======= I UNDERSTAND. This data is added to the data base.

14. i own blocks which are not red, but i don't own anything which supports a pyramid.
===== I UNDERSTAND. Two new theorems are created for proving things about "owning".

15. do i own the box?
======= NO. The box supports whatever it contains, including a pyramid.

16. do i own anything in the box?
======= YES, TWO THINGS: THE BLUE BLOCK AND THE BLUE PYRAMID.

17. will you please stack up both of the red blocks and either a green cube or a pyramid?
======= OK. Logical connectives such as "and" "or" "either" etc. are handled in both the grammar and semantics. A command is executed even if it looks like a question. The command is carried out by the following series of steps: It puts a green cube on the large red block (note that it chooses the green cube with nothing on it), then removes the small pyramid from the little red cube, so that it can use that cube to complete the stack as specified, resulting in the situation of Figure 5.

18. which cube is sitting on the table?
======= THE LARGE GREEN ONE WHICH SUPPORTS THE RED PYRAMID.
Ordinarily the system would be satisfied with calling this "a big green cube", but since we asked "which cube" it has to be more specific in naming it. The use of substitute nouns (such as "one") and pronouns in answers makes the discourse more natural.

19. is there a large block behind a pyramid?
======= YES, THREE OF THEM: A LARGE RED ONE,
A LARGE GREEN CUBE AND THE BLUE ONE.

20. put a small one onto the green cube which supports a pyramid.
"A small one" must contrast with the earlier phrase "a large block", so this means "a small block".
======= OK.

21. put the littlest pyramid on top of it.
Words like "littlest" are not in the dictionary but
======= OK. are interpreted from the root forms like "little".

22. how many things are on top of green cubes?
======= I'M NOT SURE WHAT YOU MEAN BY "ON TOP OF" IN THE PHRASE "ON TOP OF GREEN CUBES ". DO YOU MEAN:
1 - DIRECTLY ON THE SURFACE 2 - ANWHERE ON TOP OF ?
When there are several interpretations and none is clearly better, the system has to ask the user for more information. The paraphrases were included with the dictionary definition of "on", and the request for more information occurs automatically when the heuristics for resolving ambiguities aren't sufficient. A request like this can also occur when the meaning of a pronoun such as "it" or "they" is unclear.
2 ====== THREE OF THEM.

23. had you touched any pyramid before you put the green one on the little cube?
======= YES, THE GREEN ONE.
The system deduces from its memory what event is being referred to, and can relate other events to it in time. The analysis includes complex tenses of verb groups, and there are special facilities for keeping track of the various forms of irregular verbs like "have".

24. when did you pick it up?
======= WHILE I WAS STACKING UP THE RED CUBE, A LARGE RED BLOCK AND A LARGE GREEN CUBE. A time is described by naming the main goal which was being worked on.

25. why? The system remembers immediately previous sentences to understand questions like this and the following ones.

======= TO GET RID OF IT.
By keeping track of selected parts of the original subgoal tree, the system has some understanding of its own motives.

26. why did you do that?
======= TO CLEAN OFF THE RED CUBE.
"That" refers to the action mentioned in the answer.

27. why did you clear off that cube?
Here, "that" means the one just referred to.
======= TO PUT IT ON A LARGE GREEN CUBE.

28. why did you do that?
======= TO STACK UP THE RED CUBE , A LARGE RED BLOCK AND A LARGE GREEN CUBE.

29. why did you do that?
======= BECAUSE YOU ASKED ME TO.
It remembers what it did, not how the request was worded.

30. how did you do it? "It" refers to the action mentioned in the last question, rather than the answer.
======= BY PUTTING A LARGE RED BLOCK ON THE TABLE ; THEN LETTING GO OF IT ; THEN PUTTING A LARGE GREEN CUBE ON IT ; THEN LETTING GO OF THAT CUBE ; THEN PUTTING THE RED CUBE ON THAT CUBE ; THEN LETTING GO OF THAT CUBE.
This is a description of the top-level subgoals used to achieve the goal.

31. how many objects did you touch while you were doing it?
Time references can refer to a time span, using words like "while".
======= FOUR OF THEM.

32. what did the red cube support before you started to clean it off?
======= THE GREEN PYRAMID.
By remembering what has been moved, it is possible to deduce things about earlier states of the scene. We can use complex nestings of clauses like "before you started to clean it off".

33. there were five blocks to the left of the box then.
"Then" must refer to the time mentioned in the last sentence.
======= NO, ONLY FOUR OF THEM: THE RED CUBE, TWO LARGE GREEN CUBES AND A LARGE RED BLOCK.
Earlier in the dialog, information about "owning" was accepted at face value. But the system knows that it has complete information about about the basic properties of objects, including location. It checks any statements about location or other such properties to see whether it agrees or disagrees.

34. put the blue pyramid on the block in the box.
This is syntactically ambiguous, but is understood without ambiguity since the parsing corresponding to "put the blue pyramid which is on the block in the box." does not make sense in the current scene, while "put the blue pyramid on the block which is in the box" does.
======= OK.

35. is there anything which is bigger than every pyramid but is not as wide as the thing that supports it?
The system understands quantifiers such as "every", "any", "some", etc., and structures such as "bigger than", "as wide as", etc. Here "it" refers to a noun group within the same sentence but two levels up in
======= YES, THE BLUE BLOCK. the parsing tree.

36. does a steeple SORRY, I DON'T KNOW THE WORD "STEEPLE".
The system interrupts immediately when it sees a word it can't figure out.

37. a "steeple" is a stack which contains two green cubes and a pyramid.
===== I UNDERSTAND. The quote marks indicate we are defining a new word.

38. are there any steeples now?
======= NO.

39. build one. The new definition is used as a description to recognize steeples.

======= OK. The description given in the definition is enough to deduce a plan for building new objects as well as recognizing them. The word "one" means "a steeple", and the plan is carried out. First, one of the large green cubes is cleared off by removing the red cube and two pyramids. Then the two green cubes and the tall red pyramid are stacked up to fit the description.

40. call the biggest block "superblock".
======= OK. We can also give objects names.

41. have you picked up superblock since we began?
======= YES.

42. why did you drop it?
======= BY "IT", I ASSUME YOU MEAN THE SUPERBLOCK.
======= I CAN'T EXPLAIN A NON-EXISTENT EVENT.
There is no record in memory of any block being dropped.

43. is there anything to the right of the red pyramid?
======= YES, FOUR THINGS: THE BOX , THE RED CUBE, THE BLUE PYRAMID AND THE BLOCK NAMED SUPERBLOCK. Once an object has a name, the name is used to help identify it in responses.

44. thank you.
======= YOU'RE WELCOME!
Idioms can be defined without modifying the grammar.

5.5.1 Syntax and meaning

What concepts are involved in the dialog example? From the performance, it is clear that the system embodies a certain amount of knowledge about blocks, about building, about goals, time, space, words, semantics, syntax, idioms, and common sense about questions, logical inference, tolerance of false assertions, inconsistencies, failures to acheive goals, etc. Each such problem could perhaps be handled, in any particular situation, by appropriate tricks, special case detectors, reduction to standardized schematic situations, etc., but such a system would become more and more limited, unwieldy, and finally incomprehensible and incapable of extension, as situations appear in which special cases interact. In fear of this, perhaps, construction of theories involving meaning has generally been put aside or postponed in favor of attempts to construct syntactic rules that would generate exactly the "grammatical" sentences of the language. In the work of Chomsky and others it seemed at first that this might work out, but as one attempted more and more realistically comprehensive models, these too turned out to require a great body of special methods, and led to systems that were unwieldy, hard to extend, and finally incomprehensibly complex, just as was feared from the semantic approach. Perhaps, then, the attempt to split syntax completely from semantics actually makes matters worse, and one might do better by facing squarely the problems of meaning!

Such a proposal which once seemed much more difficult than syntactic analysis, now seems easier partly because the latter turned out to be so difficult and partly because advances in heuristic programming made meaning so much less mysterious. It now appears that even a modest semantic complement can greatly simplify understanding syntax.

To emphasize why purely syntactic methods cannot tell us how to parse a sentence and why meaning must be studied, consider the following two sentences:

> The city councilmen refused to give the women a permit for a demonstration because they feared violence.
>
> The city councilmen refused to give the women a permit for a demonstration because they advocated revolution.

If we have to make a choice of who "they" means, for example, to find the gender if we were translating into French, we need the information and reasoning power to realize that city councilmen are usually staunch defenders of law and order, but are hardly likely to be revolutionaries. In traditional syntactic analysis one avoids this problem by announcing both parsings, but if we are interested in understanding how the language is to be used, we have to be able to make the choice. So in addition to a grammar of a language, our program needs all sorts of knowledge about the subject it is discussing, and the ability to use reasoning to combine facts in the right ways. To understand a sentence one has to combine grammar, semantics, and reasoning in a very intimate way, calling on each part to help with the others.

> In earlier computer programs for understanding language, such attempts as were made to use such information took the form of lists of rules, patterns, and formulas. In Winograd's system, knowledge is expressed as PROGRAMS in special languages designed to gain the flexibility and power of programs while retaining much of the regularity of traditional simpler rule forms. Since each piece of knowledge can be a procedure, it can call on any other type of knowledge. Thus the "parser" can call semantic programs to see if the phrase it is proposing makes sense, and the semantic programs can call on the deductive programs to see whether that meaning of a phrase makes sense in the current real-world context, as when the choice of a pronoun's assignment depends on the preceding discourse or on detailed knowledge of the subject matter.

While Winograd's system can be described as divided into three parts --- syntax, semantics, and inference --- it is the richness of interplay permitted between these that makes it an advance over previous language-understanding programming attempts. In the following sections we will describe enough of these three "sections" to see how the whole system can handle just the first line in the sample dialog:

pick up a big red block

To fit the type of syntactic analysis he chose to use,Winograd developed a programming language (named PROGRAMMAR) that differs from other parsers in that the grammar is written in the form of a collection of programs. The grammar itself, as we shall explain, is highly suited for semantic analysis since from the start it views the "rules of grammar" as connected with the decisions one makes about conveying meaning rather than about putting words into acceptable orderings.

At the other end of the system we have the knowledge and the reasoning power of a problem-solver system, written in the PLANNER language, to give the system detailed knowledge about its universe --- in this case the BLOCKS WORLD we described in section 5.3. This makes it possible for the system to discuss not only physical happenings but also the robot's own goals and actions.

Interposed between these is the semantic system which contains processes that deduce, from the syntactic constructions, and from the programs that define the meanings of words and other constructions in terms of PLANNER programs, new procedures for the deductive system to use in answering questions, obeying commands, and acquiring new knowledge in the course of the dialog. This system is described in section 5.6. The full system contains some token knowledge also about communication between persons, so that if we say: "There is a block on a green table. What color is it?" the system will assume that "it" refers to the block (rather than the table) since one would not normally ask a question whose answer one knows.

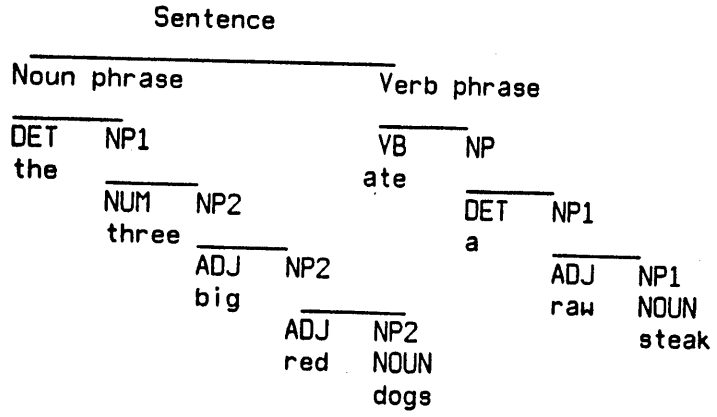
5.5.2 Systemic Grammar

> The following sections might seem unusually detailed for a progress report. But we feel that this system represents a major advance and should be presented in enough detail to see really how it works.

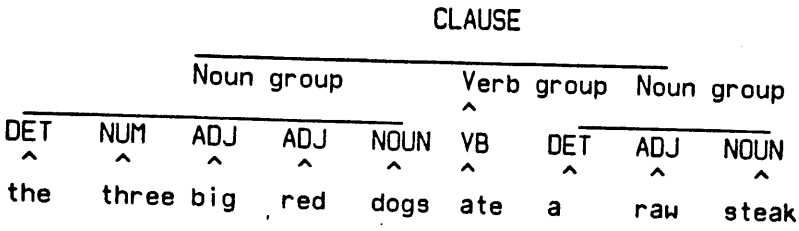
The decision to consider syntax as a proper study devoid of semantics is a basic tenet of most current linguistic theories. Language is viewed as a way of organizing strings of abstract symbols, and tries to explain linguistic competence in terms of symbol-manipulating rules. But although this approach has worked rather well in accounting for which sentences can be formed, it has been unable to shed much light on the basic problem: how does a sentence convey meaning beyond the meanings of individual words? Meanings of words depend on other parts of the discourse and intentions depend on one's general orientation and state of knowledge. We can attack the problem in the usual way, by constructing a "mini-theory" as a first approximation, then apply it to see what problems remain.

The structure of a sentence can be viewed as the result of a series of grammatical choices made in generating it. This is not a novel idea in itself; it underlies the most standard notion of generative grammar. But it is not so usual to proceed on to say: the speaker encodes meaning into the sentence by these choices, through choosing to build the sentence with certain "features"; the problem of the hearer is to recognize the presence of those features and interpret their meaning. Of course, we use "feature" to include elements of structural description as well as simple lexicographic terms.

Winograd's system is based on a theory called Systemic Grammar (Halliday, 1967, 1970) in which these choices of features are primary Instead of placing emphasis on a "deep structure" tree, it describes the way different features interact and depend on each other. In other forms of grammar, syntactic structures are usually represented as a binary tree, with many levels of branching and few branches at any node. For example, the sentence "The three big red dogs ate a raw steak." would be parsed with something like this:



Systemic grammar pays more attention to the way language is organized into units, each of which has a special role in conveying meaning. In English we can distinguish three basic ranks of units, the CLAUSE, the GROUP, and the WORD. In systemic grammar, the same sentence might be viewed as having this structure.



In this analysis, the WORD is the basic building block. There are word classes like "adjective" , "noun", "verb". The word "dogs" is the same basic vocabulary item as "dog", but has the feature "plural" instead of "singular". "Took", "take", "taken", "taking", etc., are all the same basic word, but with differing features such as "past participle" , "infinitive" , "-ing" , etc.

The next unit above the WORD is the GROUP. Noun groups (NG) describe objects, verb groups (VG) carry complex messages about the time and modal (logical) status of an event or relationship, preposition groups (PREPG) describe certain simple relationships, while adjective groups (ADJG) convey other kinds of relationships and descriptions of objects.

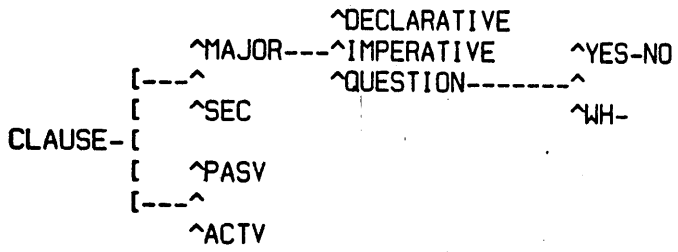
Each GROUP can have "slots" for the words of which it is composed. As we shall see, a NG has slots for "determiner" (DET), "numbers" (NUM), "adjectives" (ADJ), "classifiers" (CLASF), and a NOUN. Each group can also exhibit features, just as a word can. A NG can be "singular" (NS) or "plural" (NPL), "definite" (DEF) as in "the three dogs" or "indefinite" (INDEF) as in "a steak", and so forth. A VG can be "negative" (NEG) or not, can be MODAL (as in "could have seen"), and can have a complex tense.

The CLAUSE is the most complex and diverse unit of the language, and is used to express relationships and events, involving time, place, manner and many other aspects of meaning. It can be a QUESTION, a DECLARATIVE, or an IMPERATIVE, it can be "passive" or "active" , it can be a YES-NO question or a WH- question (like "Why...?" or "Which...?"). Our second parsing tree showed how a clause may be composed of groups, which are in turn made up of words. Also, groups often contain other groups; for example, "the call of the wild" is a NG, which contains the PREPG "of the wild" which in turn contains the NG "the wild". Clauses can be parts of other clauses, as in "Join the Navy to see the world.", and can be used as parts of groups in many different ways, as in the NG "the man who came to dinner" or the PREPG "by leaving the country".

If the units can appear anywhere in the tree, what is the advantage of grouping constituents into "units" instead of having a detailed structure like the one shown in our first parsing tree? The answer is that each unit has associated with it a set of meaning-carrying features, related by definite logical structures. The choice between YES-NO and WH- is meaningless unless the clause is a QUESTION, but if it is a QUESTION, the choice must be made.

Similarly, the choice between QUESTION, IMPERATIVE, and DECLARATIVE is mandatory for a MAJOR clause (one which could stand alone as a sentence) but is not possible for a "secondary" (SEC) clause, such as "the country which possesses the bomb." The choice between PASV --- "the ball was attended by John" --- and ACTV --- "John attended the ball" --- is on a totally different dimension, since it can be made regardless of which of these other features are present.

A set of mutually exclusive features like QUESTION, DECLARATIVE, and IMPERATIVE) is called a system, and will be diagrammed by connecting them with a vertical bar. Each system has an entry condition which can be an arbitrary boolean condition on the presence of other features. For example, in the diagram below, one of the systems has the feature MAJOR as its entry condition, since only MAJOR clauses make the choice between DECLARATIVE, IMPERATIVE, and QUESTION. We can diagram some of our CLAUSE features as:



The choice between SEC and MAJOR and the choice between PASV and ACTV both depend directly on the presence of CLAUSE. This type of relationship will be indicated by a bracket in place of a vertical bar.

In addition, a syntactic "unit" can have different functions as a part of a larger unit. A transitive clause must have units to fill the functions of SUBJECT and OBJECT, and a WH- question has to have a constituent to play the role of "question element" like "which dog" in "Which dog stole the show?".

In most current theories, there is no explicit mention of these features and functions in the syntactic rules, but the rules are designed in such a way that every sentence will in fact be one of the three types listed above, and every WH- question will in fact have a question element. The difficulty is that there is no attempt in such a grammar to distinguish meaning-conveying features such as these from the many other features we could note about a sentence, and which are also implied by the rules.

5.5.3 The NOUN GROUP

We illustrate these ideas by presenting the structure of the NOUN GROUP in some detail, closely following the presentation in Winograd's thesis.

Here is the structure of the typical NG, using a "*" to indicate that the same element can occur more than once. Most of these "slots" are optional, and may or may not be filled in any particular NG.

```
                ^   ^   ^   ^     ^     ^   ^
DET ORD NUM ADJ* CLASF* NOUN Q*
```

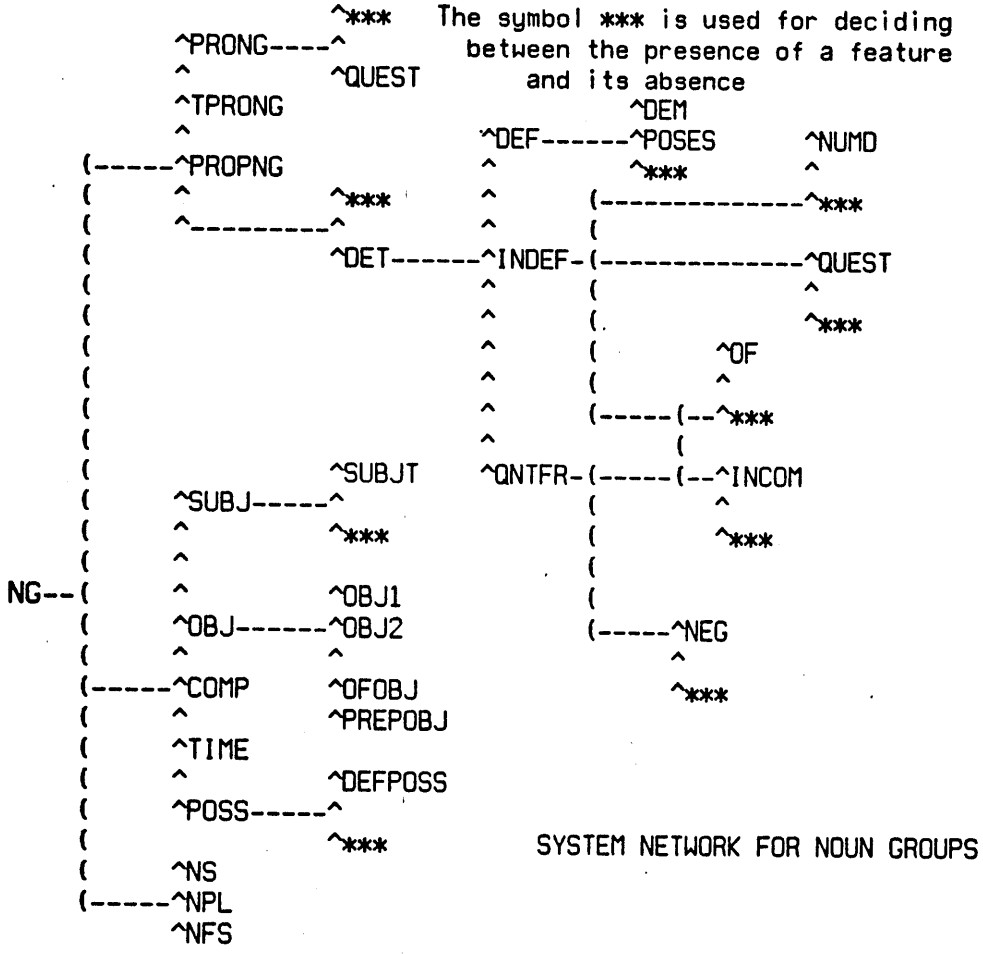
The most important ingredient is the NOUN, which gives the basic information about the object or objects being referred to by the NG. Immediately preceding the NOUN, there are an arbitrary number of "classifiers", like "plant life" or "water meter cover adjustment screw". The same class of words can serve as CLASF and NOUN, in English, and our dictionary gives the meaning of words according to their word class, because nouns often have a special meaning when used as a CLASF.

Preceding the classifiers we have adjectives (ADJ) such as "big beautiful soft red" Adjectives can be used as the complement of a BE CLAUSE, but classifiers cannot. We can say "red hair", or "horse hair", or "That hair is red.", but we cannot say "That hair is horse.", since "horse" is a CLASF, not an ADJ. Adjectives can also take on the comparative and superlative forms ("red, redder, and reddest"), while classifiers cannot ("horse, horser, and horsest"?). Immediately following the NOUN we can have various qualifiers (Q), which can be a PREPG like "the man in the moon" or an ADJG like "a night darker than doom" or a CLAUSE RSQ like "the woman who conducts the orchestra".

The first few elements in the NG work together to give its logical description -- whether it refers to a single object, a class of objects, a group of objects, etc. The determiner (DET) is the normal start for a NG, and can be a word such as "a", or "that", or a possessive. It may be followed by an "ordinal" (ORD), as "first, second, third, etc. or a few others such as "last" and "next". These are the only words that can appear between a DET like "the" and a number, as in "the next three days". Finally there is a number (NUM), like "one", "two", etc. or a more complex construction such as "at least three", or "more than a thousand". It is possible for a NG to have all slots filled, as in:

```
DET ORD  NUM ADJ ADJ CLASF CLASF NOUN   Q(PREPG)   Q(CLAUSE)
the first three old red city fire hydrants without covers you can find
```

With these basic components in mind, let us look at the system network for NG.



At the top of the diagram are some special cases which do not have the structure described above. A NG made up of a pronoun is a PRONG. It can be either a question, like "who" or "what", or a non-question (the unmarked case) like "I", "them", "it", etc. The feature TPRONG marks a NG whose head is a special TPRON, like "something", "everything", "anything", which can enter into a peculiar construction in which an adjective can follow the head, as in "anything green which is bigger than the moon". It has its own special syntax. The feature PROPNG marks an NG made up of proper nouns, such as "Oklahoma", or "The Union Of Soviet Socialist Republics."

The rest of the noun groups are the normal type, discussed above. The DET can be definite (like "the" or "that", indefinite like "a" or "an", or a quantifier (QNTFR) like "some", "every", or "no". The definite determiners can be either demonstrative ("this", "that", etc.) or the word "the" (the unmarked case), or a possessive NG. The NG "the farmer's son" has the NG "the farmer" as its determiner, and has the feature POSES to indicate this.

An INDEF NG can have a number as a determiner: "five gold rings", or "at least a dozen eggs", in which case it has the feature NUMDET, or it can use an INDEF determiner, such as "a". In either case it has the choice of being a question. The question form of a NUMDET is "how many", while for other cases it is "which" or "what".

Finally, an NG can be determined by a quantifier (QNTFR). Although quantifiers could be subclassified along various lines, we do so in the semantics rather than the syntax. The only classifications used syntactically are between singular and plural, and between negative and non-negative.

If a NG is either NUMD or QNTFR, it can be of a special type marked OF, as in "all of your dreams," but can also choose to be incomplete, leaving out the NOUN, as in "Give me three" or "I want none". There is a correspondence between the cases which can take the feature OF, and those which can be INCOM. We cannot say either "the of them" or "Give me the". Possessives are an exception, we can say "Give me Juan's" but not "Juan's of them", and are handled separately .

The middle part of the NG Network describes the different possible functions a NG can serve. In the CLAUSE, one can use an NG as a SUBJ, COMP, or objects OBJ of various types. In addition, it can serve as the object of a PREPG (PREPOBJ), in: "the rape of the lock". If it is the object of "of" in an OF NG, it is called an OFOBJ: "none of your tricks". A NG can also be used to indicate TIME, as in: "Yesterday the world ended" or "The day she left, all work stopped".

Finally, a NG can be the possessive determiner for another NG. In: "the cook's kettle" the NG "the cook" has the feature POSS, indicating that it is the determiner for the NG "the cook's kettle", which has the feature POSES.

When a PRONG is used as a POSS, it must use a special possessive pronoun, like "my", "your", etc. We can use a POSS in an incomplete NG, like "Show me yours or "John's is covered with mud". There is a special class of pronouns used in these noun groups (labelled DEFPOSS), such as "yours", "mine", etc.

Continuing to the last part of the NG Network, we see features of person and number. These are used to match the noun to the verb (if the NG is the subject) and the determiner, to avoid combinations like "these kangaroo" or "the women wins". In the case of a PRONG, there are special pronouns for first, second, and third person, singular and plural. The feature NFS occurs only with the first-person singular pronouns ("I", "me", "my", "mine"), and no distinction is made between other persons, since they have no effect on the parsing. A singular pronoun or other singular NG is marked with the feature NS. The pronoun "you" is always treated as if it were plural and no distinction is made between "we", "you", "they", or any plural (NPL) NG as far as the grammar is concerned. Of course there is a semantic difference.

5.5.4 The Parser in Action

With this sketch of some of the ingredients, we canl now follow the parser through an example to get a feeling for the way the grammar works, and the way it interacts with the different features described above. Consider the first sentence of our sample dialog

"Pick up a big red block."

The system begins trying to parse a sentence, which means looking for a MAJOR CLAUSE. It activates the grammar by calling the CLAUSE program with an initial feature list of (CLAUSE MAJOR).

The CLAUSE program looks at the first word, to decide what unit the CLAUSE begins with. If it sees an adverb, it assumes the sentence begins with a single-word modifier; if it sees a preposition, it looks for an initial PREPG. If it sees a BINDER, it calls the CLAUSE program to look for a BOUND CLAUSE. In English (and possibly all languages) the first word of a construction often gives a very good clue as to what that construction will be. In this case, "pick" is a verb, and indicates that we may have an IMPERATIVE CLAUSE. The program starts the VG program with the initial VG feature list (VG IMPER), looking for a VG of this type. This must either begin with some form of the verb "do", or with the main verb itself. Since the next word is not "do", it checks the next word in the input (in this case still the first word) to see whether it is the infinitive form of a verb. If so, it is to be attached to the parsing tree, and given the additional feature MVB (main verb). The current structure can be diagrammed as

```
(CLAUSE MAJOR)
  (VG IMPER)
   (VB MVB INF TRANS VPRT ------------------- pick
```

TRANS and VPRT came from the definition of the word "pick" when we called the function PARSE for a word.

When the VG program succeeds, CLAUSE takes over again. Since it has found the right kind of VG for an imperative CLAUSE, it puts the feature IMPER on the CLAUSE feature list. It then checks to see whether the MVB has the feature VPRT, indicating it is a special kind of verb which takes a particle. It discovers that "pick" is such a verb, and next checks to see if the next word "up" is a PRT, which it is. It then checks in the dictionary and finds out that the combination "pick up" is defined, so it calls (PARSE PRT) to add "up" to the parsing tree. We might have let the VG program do the work of looking for a PRT, but it would have run into difficulties with sentences like "Pick the red block up." in which the PRT is displaced. By letting the CLAUSE program do the looking, the problem is simplified.

As soon as it has parsed the PRT, the CLAUSE program marks the feature PRT on its own feature list. It then looks at the dictionary entry for "pick up" to see what transitivity features are there. It is transitive, which indicates that we should look for one object -- OBJ1. The dictionary entry shows that the object must be either a NG or a WHRS clause (which would begin with a relative pronoun, like "Pick up what I told you to." Since the next word is "a", this is not the case, so the CLAUSE program looks for an object by calling (PARSE NG OBJ OBJ1), asking the NG program to find a NG which can serve as an OBJ1. The structure is now

```
(CLAUSE MAJOR IMPER PRT)
  (VG IMPER)
   (VB MVB INF TRANS PRT) ------------------- pick
  (PRT) ------------------------------------- up
  (NG OBJ OBJ1)
```

The NG program notices that the upcoming word is a determiner, "a". It calls (PARSE DET) to add it to the parsing tree, then transfers the relevant features from the DET to the entire NG. It also adds the feature DET to the NG to indicate that it has a determiner. The feature list for the NG is now:

(NG OBJ OBJ1 DET INDEF NS)

since "a" is a singular indefinite determiner. The NG program then notices the feature INDEF, and decides not to look for a number or an ordinal --- we can't say "a next three blocks" -- or for the OF construction -- "a of them" is impossible. It goes on immediately to look for an adjective by calling (PARSE ADJ). When this succeeds with the next word "big", a simple program loop returns to the (PARSE ADJ) statement, which succeeds again with "red". On the next trip it fails, and sends the program on to look for a classifier, since "block" isn't an ADJ. But "block" isn't a CLASF either in our dictionary, so the NG program goes on to look for a NOUN, by calling (PARSE NOUN). This succeeds with the NOUN "block", which is singular, and the program checks to see if it agrees with the number features already present from the determiner (to eliminate illegal combinations like "these boy"). In this case, both are singular (NS), so the program is satisfied. Ordinarily it would go on to look for qualifiers, but in this case there is nothing left in the sentence. Since we have found all of the basic constituents we need for a NG, the NG program should return success. If we had run out after the determiner, it would have checked for an incomplete NG, while if we had run out after an ADJ it would have entered a backup program which would check to see whether it had misinterpreted a NOUN as an ADJ.

In this case, the NG program returns, and the CLAUSE program notices that the sentence has ended. Since a TRANS verb needs only one object, and that object has been found, the CLAUSE program marks the feature TRANS, and returns, ending the parsing. In actual use, a semantic program would be called now to understand and execute the command -- in fact, semantic programs would have been called at various points throughout the process. The final result is:

```
(CLAUSE MAJOR IMPER PRT TRANS)
  (VG IMPER)
   (VB MVB INF TRANS VPRT)-------------------- pick
  (PRT) ---------------------------------------- up
  (NG OBJ OBJ1 DET INDEF NS)
   (DET INDEF NS) ---------------------------- a
   (ADJ) ------------------------------------- big
   (ADJ) ------------------------------------- red
   (NOUN NS) --------------------------------- block
```

5.6 Semantic structures

In 5.5 we described some of the operation of the systemic grammar parsing program. For the semantic system we again will use the Noun Group as an example, to present the general idea. As one hears or reads linguistic sequences, one extracts meanings and uses them to modify one's model of the world or in some other way to organize one's behavior. In Winograd's system, the meanings are usually represented by procedures written in the PLANNER language. There are a number of ways in which these procedures are used to build up meanings by cooperation between the systemic-grammar analyzer and other processes called "semantic specialists".

One of the most obvious semantic functions of expressions is to describe objects, and the "noun group" is most commonly used for this. It contains a noun which indicates the kind of object, adjectives and classifiers which describe further properties of the object; and a complex system of quantifiers and determiners describing its logical status -- whether it is a particular object, a class of objects, a particular set of objects, or even an unspecified set containing a specified number of objects ("three bananas"), etc. The syntactic structure already discussed provides a systematic framework for such descriptions. One might object that this is too rigid and that there are other ways to describe objects. Indeed, but this one handles a wide range of ordinary cases and Winograd's PROGRAMMAR system supplies an unprecedented flexibility for introducing other methods and even complex heuristic programs for dealing with other situations.

The semantic system is built around a dozen or so programs, "semantic specialists" which are experts at interpreting particular syntactic structures. These are called by PROGRAMMAR when the parsing system believes that a certain structure, say, a noun group, has been parsed. They look at both the syntactic structures and the meanings of the words (which are also represented by programs), and build up PLANNER theorems which can be used either by the deductive mechanisms (for performing actions in, or for answering questions about, the Blocks World) or by the syntactic system itself to decide whether the proposed noun group is meaningful.

A Noun Group like "a red cube" can be described as:

```
(GOAL (IS X BLOCK))
(EQDIM X)
(GOAL (COLOR X RED))
```

The variable "X" represents the object, and this description says that the object X should be a block, it should have equal dimensions, and it should be red. A phrase such as "a red cube which supports three pyramids but is not contained in a box" would be built up from the descriptions for the various objects, and would end up as

```
(GOAL (IS X BLOCK))
(EQDIM X)
(GOAL (COLOR X RED))
(FIND 3 X2  (GOAL (IS X2 PYRAMID))
            (GOAL (SUPPORT X X2)))
(NOT (PROG X3
     (GOAL (IS X3 BOX))
     (GOAL (CONTAIN X3 X))))
```

This "meaning" is a procedure. A larger deductive system could use it

to find such an object;

to say whether one exists;

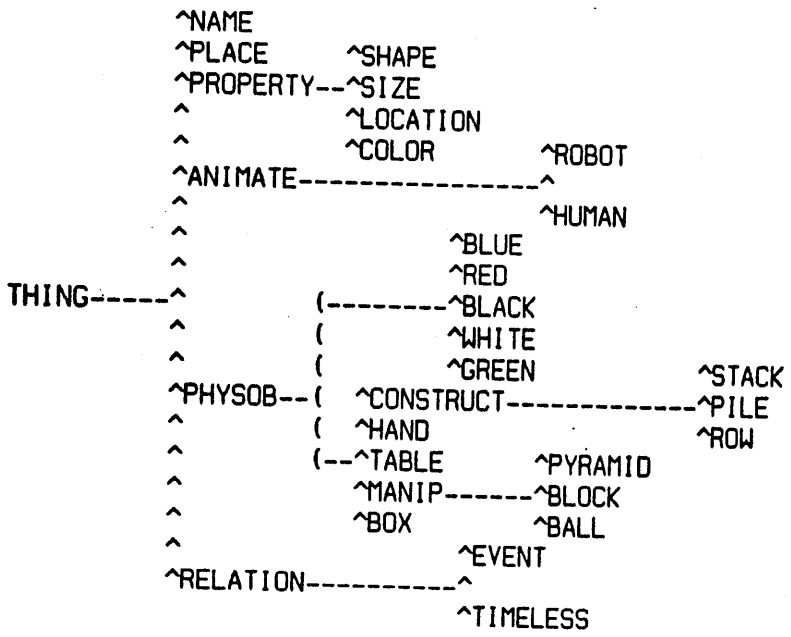
to list relations in which it does, or could, participate;

to answer more abstract questions about whether such an object could exist or (as in the BLOCKS program) to plan a sequence of actions that will cause it to exist.

Furthermore, the "theorem" that embodies the meaning could be used within the parsing process itself, for if the deductive system finds that there could be no such object then the alleged noun group would be suspect and one could search for an alternative parsing. One could imagine a much more sophisticated system that would suspend this strategy if the discourse concerns a subject, like language itself, in which normally unacceptable expressions are sometimes permitted.

How do the semantic specialists build this structure? Consider the simple expression "a red cube". First the noun group is parsed, then the PLANNER description is built up backwards by the specialists, starting with the noun, and continuing in right-to-left order through the classifiers and adjectives.

Part of the definition for a noun uses semantic markers to filter out meaningless interpretations of a phrase. The BLOCKS world uses this tree of semantic markers:



Again, vertical bars represent exclusive choices, while horizontal lines represent logical dependency. "PHYSOB" means "physical object", and "MANIP" means "manipulable object". The first specialist, SMNG1, finds that the definition of the noun "cube" is:

(NMEANS (CUBE) ((IS X BLOCK)(EQDIM X)))

which says that a cube is a block with equal dimensions. NMEANS is the name of a function for dealing with nouns, which accepts a list of different meanings for a word. In this case, there is only one meaning. When NMEANS is executed, it puts information onto the semantic structure which is being built for the object. It takes care of finding out what markers are implied by the tree, here (THING PHYSOB MANIP BLOCK), deciding which predicates need to be in a GOAL statement (like IS), and which are LISP predicates (like EQDIM). It also can decide on recommendation lists to put onto the PLANNER goals, to guide deductions.

Next, SMNG1 calls the definition for the adjective "red".

(NMEANS((PHYSOB)((COLOR X RED))))

This definition indicates that the property applies only to physical objects.

There is no absolute definition for "big" or "little"; a "big flea" is still not much competition for a "little elephant". The meaning of the adjective is relative to the noun it modifies, and it may also be relative to the adjectives following it as well, as in a "big toy elephant." As the system analyzes the NG from right to left, the meaning of each adjective is added to the description already built up for the head and modifiers to the right. Since each definition is a program, it can just as well examine the description (both the semantic markers and the PLANNER description), and produce an appropriate meaning relative to the object being described. This may be an absolute measurement (e.g. a "big elephant" is more than 12 feet tall) or a relative PLANNER description of the form "the number of objects fitting the description and smaller than the one being described is more than the number of suitable objects bigger than it is".

In adding the meaning of "red" to the semantic structure, the specialist must make a choice in ordering the PLANNER expressions. In the robot's tiny world, this isn't of much importance, but if the data base took phrases like "a man in this room", we certainly would be better off looking around the room first to see what was a man, than looking through all the men in the world to see if one was in the room. To make this choice we allow each predicate (like IS or COLOR) to have associated with it a program which knows how to evaluate its "priority" in any given environment. The program might be as simple as a single number, or it might be a complex heuristic program which takes into account the current state of the world and the discourse.

Here is the structure which would be built up by the program.

```
        (GOAL (IS X BLOCK))
        (GOAL (COLOR X RED))
        (EQDIM X) - - - - - - - - - - -PLANNER description
    (BLOCK MANIP PHYSOB THING ) - - - - - -markers
    (MANIP PHYSOB THING ) - - - - - - - - -systems
    (NS INDEF)- - - - - - - - - - - - - - -determiner
```

Let us now take a slightly more complicated NG, "a red cube which supports a pyramid," We can only summarize what happens here. First, the NG parsing program finds the determiner ("a"), adjective ("red"), and noun ("cube"). $MNG1 then creates the structure described in the previous section. Then, after further analysis a CLAUSE specialist is called to deal with "which supports a pyramid" and it constructs a corresponding planner theorem, for the meaning of "a pyramid". Next the definition of the verb "support" is called, and used to build up an assertion that the subject and object are related by SUPPORT.

The clause is now finished, and the specialist on relative clauses ($MRSQ) is called to take the PLANNER descriptions of the objects involved in the relation, along with the relation itself, and put the information onto the PLANNER description of the object to which the clause is being related. The result, for the description of "a red cube which supports a pyramid" is

```
        (GOAL (IS X BLOCK))
        (GOAL (COLOR X RED))
        (EQDIM X)
        (GOAL (IS X2 PYRAMID))
        (GOAL (SUPPORT X X2))- - - PLANNER description
    (BLOCK MANIP PHYSOB THING) - - - - markers
    (MANIP PHYSOB THING) - - - - - - - systems
    (NS INDEF) - - - - - - - - - - - - determiner
```

Relationships have the full capability to use semantic markers just as objects do, and at an early stage of construction, a relation structure contains a PLANNER description, markers, and systems in forms identical to those for object structures (this is to share some of the programs, such as those which check for conflicts between markers). We can classify different types of events and relationships (for example, those which are changeable, those which involve physical motion, etc.) and use the markers to help filter out interpretations of clause modifiers. For example, the modifying PREPG "without the shopping list" in

"He left the house without the shopping list"

has a different interpretation from "without a hammer" in

"He built the house without a hammer."

If we had a classification of activities which included those involving motion and those using tools, we could choose the correct interpretation.