Physical Object Representation and Generalization:

A Survey of Programs for Semantic-Based Natural Language Processing

Kenneth Wasserman
Department of Computer Science, Columbia University, New York, NY 10027

Abstract

This article surveys a portion of the field of natural language processing. The main areas considered are those dealing with representation schemes, particularly work on physical object representation, and generalization processes driven by natural language understanding The emphasis of this article is on conceptual representation of objects based on the semantic interpretation of natural language input. Six programs serve as case studies for guiding the course of the article. Within the framework of describing each of these programs, several other programs, ideas, and theories that are relevant to the program in focus are presented.

RECENT ADVANCES in natural language processing [NLP] have generated considerable interest within the Artificial Intelligence [AI] and Cognitive Science communities. Within NLP, researchers are trying to produce intelligent computer systems that can read, understand, and respond to various human-oriented texts. Terrorism stories, airline flight schedules, and how to fill ice cube trays are all domains that have been used for NLP programs.

In order to understand these texts and others, some way of representing information is needed. A complete understanding of human-oriented prose requires the ability to combine the meanings of many readings in an intelligent manner. Learning through the process of generalization is one such mechanism. The integration of representation and generalization in the domain of NLP is the subject of this article.

Physical object understanding is an area in which a variety of representation schemes and generalization methods have been used. In past years, researchers have devised various representation systems for objects that range from very simple PART-OF relations to complex, visually-oriented techniques. Many of these systems are driven from natural language input. Thus, physical object understanding systems serve as a good focal point for our discussion.

The need to integrate representation with generalization comes about when one is faced with the problem of understanding how several objects and/or events compare with each other. For example, a particular representation system might be able to encode that a chair has a seat, a back, and legs. Furthermore, assume that this system has represented within itself several different chairs that all have these three basic parts. Now suppose that this system finds out (reads) about a bench that has just a seat and legs. In order to recognize that the bench is just like a chair only without a back, the representation system needs the ability to make generalizations. Here the generalization would be, "an object to sit on must have a seat and legs." One could argue that a complete representation of chairs and benches requires knowledge of their common parts. Thus, generalization is intertwined with representation. The generalization process is, of course, more than just a way of structuring knowledge. Generalization is one very important aspect of learning.

Recent work in NLP has recognized the interaction between representation and generalization and has started to integrate them into a unified approach to understanding. The need to integrate these heretofore separate areas is particularly obvious in systems that are intended to read and process a large number of texts. As a matter of convenience, this article will refer to representation, generalization, and their interrelation as *representation/generalization.*

This article surveys a portion of the field of NLP. The main areas considered are those dealing with representation schemes, particularly work on physical object representation and generalization processes driven by natural language understanding. A historical account of how research has proceeded in these areas is given with emphasis on the past few years, during which the field of NLP has grown tremendously. Somewhat stronger consideration is given to work done in representation than in learning (generalization). This is simply due to the overwhelming amount of research that has been done in conceptual representation. Early work in learn-

I would especially like to thank Michael Lebowitz for his many helpful suggestions and for reading over earlier drafts of this article I also greatly appreciated comments from Kathy McKeown and Rod Farrow that were helpful in organizing and refining this article. The development of RESEARCHER at Columbia is supported in part by the Defense Advanced Research Projects Agency under contract N00039-84-C-0165

ing did not deal with complex representations of events or objects, so there was little need to integrate generalization with representation. Therefore, much of the material in this article will appear to be divided into two distinct groups: representation and generalization.

We have chosen to present the work in representation/generalization by following the chronological progression of computer programs written for NLP. The reasons for doing so are twofold. Most researchers in cognitive science with a computer science background at some point embody their ideas in a program as a vehicle to test them on real-world problems. Thus, NLP programs written to date generally span the body of research done in this field. The second reason to discuss these programs is that they incorporate ideas from outside the field of AI. Any single functioning NLP program must in some way incorporate concepts that many researchers outside of computer science grapple with. A focus on programs still allows us to report work done by cognitive scientists who lack a computer science leaning, as well as those researchers who are program-oriented. By following the chronological progression of these programs, we can get a feel for where current NLP research came from and where it is headed.

The six programs that will guide the course of this article are: SHRDLU (Winograd, 1972), MARGIE (Schank, 1975), GUS (Bobrow *et al.,* 1977), OPUS (Lehnert and Burstein, 1979), IPP (Lebowitz, 1980) and RESEARCHER (Lebowitz, 1983a). Within the framework of describing each of these programs, several other programs, ideas, and theories that are relevant to the program in focus will be presented.

The first program, SHRDLU, provides a context for discussing a very important technique used in representation systems: semantic networks. Some rudimentary learning techniques were also explored in conjunction with this program and they are mentioned in this section.

Conceptual Dependency [CD] (Schank, 1972) forms the backbone of MARGIE. CD and other similar systems offer language-independent means for representing knowledge derived from natural language input. Other related linguistic theories are also mentioned while describing MARGIE.

GUS was one of the first NLP programs to employ Marvin Minsky's frame idea (Minsky, 1975) for representing knowledge. KRL (Bobrow and Winograd, 1977a), a language built concurrently with GUS and designed to provide an environment for developing frame-based systems, is also treated in this section.

The next two programs presented, OPUS and IPP, are recent developments dealing with physical object representation and generalization-based memory, respectively. OPUS uses Object Primitives, an extension to CD, to represent real-world objects. IPP employs Memory Organizational Packets [MOPs] (Schank, 1980; Schank, 1982) to encode action-oriented events in a system that makes generalizations about terrorism stories.

RESEARCHER continues in the vein of IPP and applies similar concepts of generalization-based memory to the domain of understanding physical objects. It integrates a robust physical object representation scheme with an advanced generalization method in an NLP system designed to read, understand, and remember patent abstracts. As such, it also demonstrates how hierarchically structured objects can be generalized about as part of understanding.

The OPUS, IPP, and RESEARCHER programs, as well as several other ones discussed within their contexts, represent the state of the art in NLP, as far as physical object representation/generalization are concerned.

SHRDLU—Representation Using Semantic Nets

We start by considering a system concerned with problems similar to the ones faced by many researchers working on representation/generalization. Representing physical objects and understanding natural language about them is what SHRDLU (Winograd, 1972) was all about.

In the early 1960's work in NLP centered on computationally intensive programs that applied a small set of general, usually syntactic[1] rules to some input text, in order to achieve a desired result. These programs are typified by those that tried to do machine translation of one natural language into another. As is well known, these attempts were unsuccessful (Tennant, 1981). Several years later, as researchers realized that more specialized rules were needed and computers became more capable, NLP programs changed in nature. The result was that programs could employ many specific rules for processing purposes and/or include large amounts of data for representational uses. This, of course, brought about the problem of what kinds of rules to use and how to control them.

SHRDLU was one of the first of this new wave of NLP programs. It was a fully integrated program that dealt with a very specific domain, the blocks world. As implemented, the computer created a simple setting containing images of cubes, pyramids, and the like on a video display, along with an imaginary arm that could move these objects around. Within this world, SHRDLU allowed the user to request rearrangements of the blocks, ask questions about the state of the world, and converse about what was possible within this world.

What made SHRDLU a truly landmark program was the way it accomplished its goals. Three major components made up the system: a syntactic parser based on an Augmented Transition Network [ATN] (Thorne *et al.,* 1968; Woods, 1970), a semantic processor used to interpret word meanings, and a logical deductive segment that figured out how to perform the user's requests and answer questions about what is possible in blocks-world. The functioning of the various components of SHRDLU proceeded as follows: The ATN-based syntactic parser would figure out what possible meanings the input text might have; next the semantic

[1] *Syntactic* is used to mean the simple subject, verb, object ordering of a sentence Whole or even partial grammars were not used in early machine translation attempts Most sentences were translated on a word-by-word basis

procedures would pick one of these meanings based on its knowledge of the state of the blocks-world; finally the logical deductive components would create a plan for fulfilling the user's request.

Another early program to make use of an ATN parser was LUNAR (Woods *et al.,* 1972). This program functioned as a question answering front-end to a database about moon rocks. LUNAR's vocabulary and parsing capabilities far exceeded SHRDLU's; however its data representation was the same that the underlying database had, and as such was not particularly interesting from a cognitive point of view. On the other hand, SHRDLU's data representation was very interesting and, at the time, was in the forefront of AI research.

SHRDLU maintained its knowledge in both procedural and declarative formats. The declarative knowledge was represented in the form of a *semantic network.* Semantic nets, as they are commonly called, were first described in (Quillian, 1968). They are arbitrarily complex networks in which nodes represent actions, ideas or, in the case of SHRDLU, physical objects. Arcs connecting nodes represent relations among them. For example, if there is a pyramid on top of a block, where the pyramid is represented by a single node and so is the block, then an arc connecting them would represent the relation SUPPORTED-BY An IS-A link (arc) is what is used to represent the concept that one node is an instance of another. For example, a dog IS-A mammal. All the properties that a mammal might have can be inherited by a dog. Thus, if the network had the fact that a mammal breathes air encoded in it, then it would be assumed that a dog also breathes air. Any relation the programmer chooses can be represented by arcs in semantic nets. Aside from static physical relations, like SUPPORTED-BY, and classification relations, like IS-A, more emphatic relations, like MUST-BE-SUPPORTED-BY and CAN-NOT-BE-A, are possible. Thus, a mammal CAN-NOT-BE-A reptile. The deductive reasoning procedures in SHRDLU make use of these relations.

Much has been written about semantic nets (see Woods, 1975 for example). They have been (and perhaps still are) the dominant knowledge representation system used in NLP, if not in all of AI. SHRDLU exemplified the best points about semantic networks. The simple node-arc formalism provides for easy representation of associations. They are useful at encoding static factual knowledge and are versatile in that they permit a wide range of data to be encrypted. Because of the limited domain of knowledge needed to understand the blocks-world, few of the difficulties and limitations of this scheme surfaced (Wilks, 1974), which is one of the reasons why SHRDLU was so successful. Among the shortcomings of classical semantic nets are: no universally accepted meanings for links; difficulty in representing time dependent-knowledge; problems resulting from the need to organize and manipulate a large network. Nevertheless, semantic nets are a very useful tool for knowledge representation.

One of the consequences of picking a good representation system is that some seemingly difficult problems become relatively easy to solve. By using semantic nets to represent the physical objects in a blocks-world, learning about simple object structures can be carried out. Of particular interest is the work Winston (1977) did with a program [ARCH] to learn concepts, such as the form of an arch. An arch can be represented by a three-node semantic net. After presenting the ARCH program with a correct example of an arch, subsequent three-node nets are inspected by the computer along with external input declaring each example to be correct, nearly correct, or incorrect. From these data, the program generalizes what it means for a structure (semantic net representation) to be an arch, and updates the semantic net. Specifically, the program compares the training examples it is given and extracts the information common to the correct examples that does not contradict what has been learned from the incorrect examples. Winston's work demonstrated the usefulness of generalization, particularly in the context of NLP. The objects generalized were fairly simple compared to the type used in later programs, such as RESEARCHER.

In SHRDLU, semantic networks were sufficient to capture simple relations among block-like objects. A complex physical object with many sub-parts could be represented by a simple semantic network, but it would become an unwieldy computational object to manipulate. For example, representing an automobile would be rather messy using this scheme. Furthermore, the fact that a car is usually thought of as one object is lost to a conventional semantic net representation because all nodes have an equal status. Thus, the car's tire could seem as important as the whole car.

One way to overcome the inability of most semantic net representation systems to deal effectively with large networks of data is to chunk information into regions within the network and treat these chunks as if they were individual nodes. Thus, a large semantic net with 10,000 nodes could logically be reduced to a network of, say, 200 chunks in which each of the 200 chunks would contain sub-networks of a small size. This *partitioning* of a network was proposed by Gary Hendrix (Hendrix, 1979).

Several advantages over simple semantic nets are apparent in his scheme. By separating low-level knowledge from high-level knowledge, the encoding process can represent more varied information. For example, the color, shape, and size of an object could be linked together within a partition and the partition itself could have links to other nodes or partitions (*e.g.,* indicating higher-level facts about the object's purpose).

This hierarchical partitioning results in smaller numbers of objects at any one level that need to be manipulated. Furthermore, partitions are useful for grouping objects so that they can be quantified. That is, a section of a semantic net can be designated so that all its members have some particular property while no objects outside it do. *Frames* (Minsky, 1975) are another way of solving many of the same problems as partitioned semantic nets.

Summary. The SHRDLU program was a milestone in NLP research. It made extensive use of semantic networks as a means of representing knowledge about a blocks-world. By using a syntactic parser, it could perform the commands requested by users and answer questions posed in English. Few limitations of the program were apparent because of the very limited domain in which it dealt.

Semantic networks have proved to be an extremely useful knowledge representation technique. They were used in SHRDLU to represent simple physical objects, but can be used to encode practically anything. Although they are very versatile, they have some important limitations, including the lack of standardized meanings for links and difficulty in manipulation of large network structures. The use of partitioned semantic nets generally solves the large network problem by breaking it into groups of small sections.

The structure of semantic nets allows them to be used for generalization. Links that allow for inheritance of properties from higher level nodes in the network, are the key to carrying out simple learning from examples.

MARGIE—Conceptual Dependency and Other Linguistic Theories

Syntactic parsing worked well in the blocks-world domain, but a deeper understanding of language is called for when using representation/generalization schemes that encode complex data. This section describes one approach to representing the meanings of components that are presented via a natural language.

While researchers in psychology, like Quillian, and in computer science, like Winograd, were working out representational issues using semantic nets and the like, linguists were making great strides forward in a relatively new field called computational linguistics. This branch of linguistics is mainly concerned with using computers to simulate NLP. One way of breaking down computational linguistics is into syntax, semantics, and pragmatics.

Syntax, in a computational linguistic environment, implies the study of sentence analysis and generation from a purely structural viewpoint. Noam Chomsky's theories of generative grammars (Chomsky, 1965) and his classification hierarchy of formal languages were the modern starting points in this subfield. In addition to Chomsky's work, there has been a fairly large effort in describing and building syntactic parsers. Examples of the research in this area are ATNs (Augmented Transition Networks) (Thorne *et al.*, 1968; Woods, 1970), which form the basis of several powerful computer parsers, including the one used in SHRDLU.

Chomsky is credited with revolutionizing linguistic theory. However, he has aroused many critics who point out his failure to deal with semantic and pragmatic issues in language comprehension. Semantics is generally understood to be the study of language meanings, while pragmatics concerns itself with connecting meaning to real-world experiences. Although these definitions are easy to state, in practice, the distinctions between semantics, pragmatics, and syntax are often blurred.

Following the demise of early attempts to do machine translation among natural languages, many computational linguists began focusing their attention on problems of semantics. The early NLP programs were strictly syntactic in nature. Many researchers felt that these programs, were incapable of doing an adequate job of understanding, necessary to perform machine translation or paraphrasing.[2] Semantics seemed to offer a way to improve greatly upon the performance of these programs. Writing programs that could understand the meanings of the words that they were reading became one new theme of NLP research.

One such program, MARGIE (Schank, 1975), was created with several objectives, including the paraphrasing of single sentences, while serving as a test bed for a new theory of semantic representation called Conceptual Dependency [CD] (Schank, 1972). Roger Schank, the principal designer of CD, set out to synthesize some recent work in linguistics and psychology into a consistent and useful theory that would lend itself to computerization. CD is a language-independent, primitive-based representation scheme for NLP. It is primarily based on the ideas of both *semantic primitives* and *case grammars* which will be discussed below. MARGIE was the first attempt at testing this theory in a computer environment.

MARGIE functioned in two similar modes. In paraphrase mode, MARGIE would read English sentences and parse them into an internal CD representation. In this form various inferencing systems would produce other CD-forms. The last stage of this mode would generate an output sentence based on the CD-forms. The inferencing mode of MARGIE worked in a similar manner. However, instead of producing a complete paraphrase of the original sentence, MARGIE would output a series of statements concerning what inferences it made about the meaning of the input text.

To get an idea of what MARGIE's capabilities were, consider the following examples, taken from (Schank, 1975):

In paraphrase mode the input text "John advised Mary to drink the wine" would produce the output:

```
John told Mary that drinking the wine would benefit her
```

This shows that MARGIE must know something about the meaning of the verb "advise." In fact, CD provides the program with a method for classifying all action-based verbs [ACTS]. Although verb classification is not directly applicable to physical object representation, CD provides a paradigm for developing primitive-based understanding schemes. Before a description of CD is presented, consider how MARGIE worked in the inferencing mode

The input: "John gave Mary an aspirin." would cause MARGIE to display the following inferences it had made (among others):

1. John believes that Mary wants an aspirin
2. Mary is sick

[2] It should be noted that early NLP programming attempts did not do an adequate job of syntactic processing. Since then there have been major advances in the ability to use syntax as the basis of NLP systems Many of today's NLP programs rely on syntax, often mixed with other processing techniques, and perform quite well

3. Mary wants to feel better
4. Mary will ingest the aspirin

These examples illustrate that CD must also be capable of representing the meaning of causal connectives. That is, inference (1) (and other beliefs) causes inferences (2) and (3) to be made, which explain the stated action of John giving Mary the aspirin. MARGIE must also have encoded within itself the knowledge that aspirin is usually ingested, in order to make inference (4).

CD works on the theory that all actions (verbs) can be reduced in meaning to combinations of a small group of primitive ACTS. For each ACT represented, there are a fixed number of arguments that accompany it. That is, an actor, recipient, object or other possible case slots must be filled for each ACT. Thus, for example, "John gave Mary an aspirin" would have the representation:

```
(ATRANS)
ACTOR:    John
FROM      John
TO        Mary
OBJECT    aspirin
```

ATRANS, one of the primitive ACTS, is used to represent the meaning of the verb "gave" and indicates Abstract TRANSfer (of possession) of an object. Other verbs, such as "take," are also represented by ATRANS, but have their case slots filled differently.

CD is capable of representing a wide range of actions and situations. In addition to the basic ACTS, both mental and physical states of a being or an object can be encoded. The fact that an event may enable, disable, cause, or generally affect a state is also representable within CD. Using these connectives, it is possible to represent the meaning of a series of sentences that constitute a story with one complex CD structure.

Schank's theory of Conceptual Dependency was not completely new to the field of linguistics. Two main areas of research contributed to its synthesis. The first was the development and study of case grammars (Fillmore, 1968). Case grammars were a byproduct of both classical linguistics and Chomsky's transformational grammar. They reflect classical linguistics in the sense that they identify the various parts of a sentence such as the main verb phrase and noun phrases. However, it is not the surface structure of the sentence that is extracted, but rather the meaning. Thus, regardless of the formal structure of the sentence, the "case frame" extracted by using case grammars will be the same for sentences employing the same main verb. Structurally, the case frame looks very much like what was presented in the CD examples (above) with actor (or agent), object, instrument, and a few other slots available. Case grammars classify verbs by what slots (cases) must accompany a particular verb. Thus, for example, if the verbs *open* and *throw* require the same slots OBJECT, AGENT, and INSTRUMENT for their case frames then they would be grouped together. CD goes beyond case frames, by defining a system of primitives and rules to manipulate them that captures the meaning of a sentence, rather than having a case frame for every verb.

The second building block of CD comes from both linguistic and psychological research. Semantic primitives are generally defined to be the lowest level of symbolism in a representation system. In practice, an understanding/representation system uses semantic primitives as a way of classifying some group such as actions or physical objects. CD is an example of a non-hierarchical classification scheme using semantic primitives.

The use of semantic primitives in a representation scheme can also be of help in processing. That is, inference rules can be grouped according to which primitive classes they apply to. This allows a processing system to determine easily what inference rules should be tried, which reduces search time. For example, the ATRANS ACT in CD can have the rule if the FROM slot filler is not specified, then fill it with the ACTOR slot value, attached to it. Other ACTs may not need such a rule and they need not have one since rules can be specifically bound to a given semantic group.

Some recent psychological research (*e.g.,* Rosch *et al.,* 1976), has investigated the existence of fundamental classes of physical objects. They give a fair amount of evidence which shows that natural categories of objects exist that people use while perceiving physical objects in the real world. Other work by George Miller (Miller, 1975) has given strong support to the thesis that verbs can be categorized as well. In one study he found that English has over 200 words that have the semantic component "to move." These studies show that humans make considerable use of categorization as a way of perceiving and understanding input from the real world. Furthermore, they suggest that fundamental meanings in natural language might be tied to real-world objects and/or events.

The concept of categorization is related to the idea of semantic primitives. Categorization is a hierarchical way of grouping entities so that some organization is apparent. Biological taxonomy is an example of such a categorization system. Semantic primitives strive to reduce real-world knowledge into meaningful groups, usually in a non-hierarchical structure. Thus, categorization and semantic primitives are both ways of helping people and/or machines perceive data from the real world.

Yorick Wilks has developed a system that he calls *preference semantics* (Wilks, 1973), which also uses semantic primitives. Preference semantics is a system whereby the meanings of some words help to disambiguate the meanings of other words while parsing input text. Each word that his system can understand consists of a dictionary entry that classifies the word into one of five major categories. Within the definitions are data that include how to interpret other words read in the same context. Thus, for example, the sentence "John grasped the idea" is understood by using information encoded in the definitions of each word and inferring that if John is grasping a non physical object then the meaning of "grasp" must be "understand." Wilks also built a program (Wilks, 1975) that uses preference semantics to do translation of English text into French. This was accomplished by making use of the fact that preference semantics distinguishes different word senses. Thus, when a given word sense was detected in the English input, its equivalent meaning in French was stored for use in output generation.

Other NLP systems that use representation mechanisms similar to Wilks's program and MARGIE are The Word Expert Parser (Small, 1980), a system much like preference semantics that is totally dictionary-based; SAM (Cullingford, 1978; Schank and Abelson, 1977), a program that uses CD representations built into higher level knowledge structures called scripts; and PAM (Schank and Abelson, 1977; Wilensky, 1978), a high-level representation system that understands stories in terms of plan-based schemes. SAM and PAM share an English language parser called ELI (Riesbeck and Schank, 1976). Both programs are a continuation of Schank's work; they are more advanced than MARGIE in that they understand stories in terms of real-world events. That is, scripts are used to group events into logical units, such as the chain of activities that occur in a restaurant setting. Plans are used to satisfy goals and explain events by specifying a sequence of actions that are needed to achieve a desired result.

Summary. MARGIE was basically a way of testing CD. Later programs like SAM and PAM used CD as the basis for limited natural language understanding systems. CD has proved itself as a robust representation scheme that is particularly well suited to action-oriented events. It has the expressiveness necessary to capture causality accurately and the conciseness to avoid ambiguity. However, it has several drawbacks. The use of a small set of primitives results in the loss of some meaning in certain contexts. Furthermore, static factual knowledge (*e.g.*, physical object descriptions) is almost completely neglected by most CD implementations.

The main reason for studying CD and similar systems is that they have demonstrated the usefulness of primitive-based, semantic representation systems for use in NLP. Case frames, suitably modified for physical object relations, and semantic primitives seem to offer powerful tools for formulating a theory of object representation. Furthermore, the formalism of case frames is quite helpful for performing generalization, as will be seen when IPP is discussed.

GUS—Frame-based Representation Schemes

Semantic networks offer a plausible formalism for physical object representation systems, but have several problems. The solution seems to be the partitioning of a network into groups of nodes that are logically compatible Hendrix introduced partitioned semantic networks as one possible scheme; another scheme was used as the basis of GUS (Bobrow *et al.*, 1977).

SHRDLU and MARGIE were very useful experimental programs but they did not have much application to real-world situations. GUS was designed to provide information on airline flight schedules. Although GUS was still an experimental program, and dealt with only a small number of airline flights, it represented a move in the AI community toward using natural language input/output modules (front-ends) for databases. GUS was one of the first programs to make explicit use of Minsky's frame concept.

GUS's domain of discourse was very limited; in fact, it only knew about airline flights scheduled for cities within California. It played the role of a travel agent during a conversation with a user. An initial database was extracted from the Official Airline Guide. With this data in a suitable frame format, and a parsed user request, GUS reasoned out a correct and appropriate response.

Frames are conceptual objects that are used as an organizational mechanism for grouping pieces of knowledge into logically consistent blocks. They are most easily thought of as an extension of semantic networks where each node is a comparatively large structure that contains enough information to describe an item adequately at some level of detail. While a node in a semantic net usually is simply the name of an item, a frame can possess information about how to classify an item, how to use it, what attributes it has, and virtually anything else that might be useful to know about an event or object. Furthermore, the knowledge encoded in a frame need not be static (declarative): it may be dynamic (procedural), or it can be a combination of these (Winograd, 1975). For example, if an airline reservation system used a frame to represent each date a plane reservation was made on, it might have slots in the frame as follows:

```
YEAR
MONTH
DAY-OF-MONTH
DAY-OF-WEEK
```

The information filling the YEAR, MONTH, and DAY-OF-MONTH slots might be filled with static data (probably single numbers). The DAY-OF-WEEK slot might contain procedural knowledge as follows:

```
(IF YEAR and MONTH and DAY-OF-MONTH are filled
THEN [FIGURE-WEEKDAY])
```

GUS ran by using information encoded within several different frames to guide its operation. For example, at the start of a conversation, GUS would try to find the data needed to satisfy the requests of a prototypical dialog frame. The attempt at filling in slots would lead to the need to fill in lower level frames before the dialog frame would be complete. Thus the date frame might have to have its slots filled in before it could be included as part of the dialog frame. By having a sequence of prototype frames to follow, GUS achieved its goal of acting like a travel agent.

The term slots refers to the "important elements" (Winograd, 1975) in a frame. Slot fillers can be thought of as references to other frames, which is what Minsky originally proposed. In any particular application of a frame system, a considerable amount of thought must be given to how many slots should be used and what they should contain. A guiding principle for frame slot selection is, "A frame is a specialist in a small domain" (Kuipers, 1975).

One very important aspect of the use of frames as a knowledge representation scheme is the default filling of slot values for instantiated frames from stereotypical frames. An instantiated frame is simply one that has its slots filled, at least partially. Default values for frame slots can be easily set up by placing them in a stereotype frame and programming a system so that if no value for a particular slot is specified, then it is inferred from the stereotype. Generally, this default processing seems to make sense. For example, if the YEAR was not explicitly given in the date frame (shown above) then it would be reasonable to assume that the value of the slot should be the current year (as most airline reservations are not booked too far in advance). However if the DAY-OF-MONTH was not given, it would obviously be a mistake to assume some value from a stereotype (assuming that only a few reservations are made on any given day).

In order to use frames effectively as a representation system several other operations, aside from default processing, are essential. These include matching one frame against another, allowing for inheritance of properties from higher level frames, type checking the values that can fill a slot in order to ensure that only certain ones are accepted, and general abilities to manipulate a connected network of frames. KRL (Bobrow and Winograd, 1977a), a language that was developed specifically to allow for knowledge representation in the form of frames, includes facilities for the aforementioned functions and others. Many of these functions, particularly matching and inheritance, are of importance for use in systems that perform some sort of generalization about their knowledge.

Although GUS was not a particularly intelligent or robust system, it was a great asset in the refinement of some of Minsky's ideas about frames. It also served as a model for other programs written in KRL, such as COIL (by Lehnert (Bobrow and Winograd, 1977b)), an NLP program that concerns itself with drawing inferences about physical objects. Other NLP systems that are also strongly framed based include: Ms. Malaprop (Charniak, 1977), a program that reads stories about painting; SAM (Cullingford, 1978) and PAM (Wilensky, 1978), discussed earlier; IPP (Lebowitz, 1980) and RESEARCHER (Lebowitz, 1983a), described in detail in later chapters.

Many other very high-level representation languages for AI exist. KLONE (Brachman, 1979) and FRL (Roberts and Goldstein, 1977) are two systems similar in purpose to KRL.

KLONE is both a language (embedded in LISP) and a methodology for organizing partitioned semantic networks. Objects represented in KLONE are structured much like they are in a frame-based scheme. However, KLONE's structural formalism also provides a way of establishing inheritance hierarchies. A distinction is made between stereotypical objects and instantiated ones. Thus, the properties of an object can be attached either to a stereotype for that object or to the object itself. Because of the hierarchical nature of KLONE, complex, but well organized inheritance dependencies can be established. By using a limited set of possible links, the semantics of the network are clearly defined. The meanings of the allowed links have been chosen so that consistency and accuracy prevail in the final representation.

FRL is much like KLONE, but instead of imposing restrictions on the semantics of links, it forces the network of frames to be hierarchically connected. That is, all frames must be joined together using INSTANCE and A-KIND-OF links. Therefore, the representation tree (actually a network that is treelike) has as its root the most general object (frame), and its leaves are the lowest level instances of whatever the network is representing. For example, if one were representing car models, the root frame might be all automobiles; below that, frames encoding General Motors, Ford, and Toyota cars; and at the bottom of the tree there would be Celicas, Skylarks, Mustangs, and so forth. The A-KIND-OF links point backward, so that Buicks are A-KIND-OF General Motors car. Unless otherwise specified, Buicks would inherit all the properties that are common to General Motors cars. This type of representation is very helpful in forming and storing generalizations made about objects or events.

Summary. GUS uses frames as a way of representing data on airline flight schedules. It also makes use of framed knowledge to guide its goal-oriented processing. Frame representation schemes are an improvement over those using simple semantic nets. They allow for grouping data, much like partitioned semantic networks. Furthermore, most systems employing frames allow for them to be structured in a hierarchical manner so that categorization and inheritance dependencies can be established.

KRL, FRL and KLONE are languages that are based on frame or framelike representations. They all offer ways for describing inheritance, matching one frame against another, and various other functions. KLONE is the newest and most successful of these. It provides a consistent set of semantics for linking together frames, and thus solves one of the problems that has plagued semantic network schemes.

The use of frames linked together into hierarchical structures is a representation that lends itself to generalization processing. INSTANCE and A-KIND-OF links correspond to specialization and generalization, respectively. Many representation/generalization schemes use this basic formalism in constructing complex network descriptions of physical objects.

OPUS—Physical Object Representation Schemes

SHRDLU addressed the problem of representing small numbers of blocklike objects. An obvious extension of this is to encode information intelligently about large numbers of arbitrarily complex real-world objects. This section describes several methodologies for doing so.

Physical object representation schemes for NLP seem to fall into three major groups. The first group consists of those schemes that are mainly concerned with representing the way in which objects are used. That is, the functionality of a physical object or the way humans think of an object while performing a task involving it (Grosz, 1977). The second group is formed by those schemes that strive to encode some fundamental properties (*e.g.*, melting point or density) of physical objects. The remaining group includes those systems that seek to represent physical objects from a visual perspective, and are therefore useful for describing an object's structure. These groups are not necessarily distinct, in that some representation schemes can be members of more than one group. To give a better idea of what these groups are, one example system from each group will be examined.

Object Primitives (Lehnert, 1978) are an excellent example of a physical object representation scheme that is a member of the first group. This representation scheme was designed to be an extension of CD. Each of the seven primitives stands for a basic attribute of an object. By combining several of these attributes together, any object can be described. For example, an ice cube tray might have the Object Primitive representation taken from (Lehnert, 1978):

```
[Ice Cube Tray
    (a SOURCE with
        ⟨output = Ice Cubes⟩)
    (a CONSUMER with
        ⟨input = Water⟩)]
```

Here the SOURCE and the CONSUMER are two of the seven possible Object Primitives. Notice that no attempt is made to encode the physical form of an ice cube tray However, the functional features of an ice cube tray are represented by this scheme in a manner that is consistent with other CD-forms.

The primary purpose of OPUS (Lehnert and Burstein, 1979) was to read sentences about physical objects and convert them into Object Primitive representations. OPUS can be classified as an expectation-based parser that uses its knowledge about physical objects to aid in understanding input text.

The program "understands" physical objects in an everyday type environment. The representation scheme concentrates on how objects are to be used and allows utilitarian inferences to be made. For example, the sentence:

John opened the bottle and poured the wine

would be represented by a structure that includes such inferenced facts as:

- A cap was removed from the bottle.
- Wine was in the bottle
- Wine was emptied from the bottle

This idea fits in well with the original concept in CD that ACT representation is central to understanding and that various connectives allow for merging ACTS into complex events. The work that Wendy Lehnert did to extend CD was to define seven Object Primitives that function, in object representation, much like Schank's ACTS, which deal with human-oriented events.

An example of a scheme from the second class of physical object representation schemes is the work Gordon Novak (Novak, 1977) did to develop a canonical physical object representation system for use in a program called ISAAC. This program reads and solves elementary physics problems stated in English. Although this is a NLP application program, the representation for the objects being described in the problems is fundamental in the sense that only the physical properties of the object are encoded. Thus, for example, a dog standing on an inclined plane might be represented by a point mass; the fact that the animal is a dog has no significance in this context.

This scheme is canonical because many different objects are reduced to the same representation that contains all the information to classify these objects uniquely. Canonical representation is typical of physical object representation schemes that fall into this second class. Schemes in this class are generally very useful in specific domains, but are not too applicable to everyday type situations. The Object Primitives scheme is canonical in the sense that an ice cube tray has only one purpose (and therefore only one representation). However, it is qualitatively different from ISAAC's representation scheme because Object Primitives does not try to capture fundamental physical properties of an object.

An important sub-class of these schemes has received considerable attention recently. The term qualitative physics (de Kleer and Brown, 1983; Forbus, 1981; Hayes, 1979) is used to refer to the notion of understanding real-world physics for AI purposes. This implies that qualitative physics is simpler than classical physics and that it can function well in commonsense reasoning processes. Qualitative physics differs from other schemes that fall into this second class in that qualitative physics schemes are intended to be applicable to a wide range of situations.

Representations that relate to visual processes constitute the third class of object encoding systems. A program written by Stephen Kosslyn and Steven Shwartz (1977) attempts to simulate how people use visual data. Their program models only a few aspects of visual processing. It is able to search an input image for various sub-parts and identify their position relative to other parts, regardless of the scale or, to some extent, the angle of view. Running in reverse, the program is also able to construct well proportioned images by using its knowledge of how parts can interconnect. This type of ability may be useful in NLP systems that need a structural description of an object.

There has been a rather large amount of research relating to physical object perception in recent years. Both experimental psychology and robotic vision processing are concerned with how humans recognize real-world objects. Much of this work is based on the idea that scenes are decomposed into sets of primitive elements with relational elements holding an image together. Some strong evidence that this kind of processing takes place in children has been uncovered (Hayes, 1978). Vision research spans a wide range of image representation levels (see Cohen and Feigenbaum, 1982, for an overview). At the lowest level, scenes are usually encoded on a point-by-point basis, while the higher levels may approach abstractions characteristic of schemes used for natural language processing. Kosslyn and Shwartz's model of vision processing fits somewhere in the lower to middle range of these schemes.

Summary. OPUS is primarily concerned with the way objects are used in everyday-type settings. It is a fairly simple system designed to test a physical object representation scheme that serves as an extension to CD.

Most physical object representation schemes for NLP have one particular specialty OPUS offers a system, Object Primitives, that mates with CD but lacks the ability to capture detail of the structure of objects. Other systems, like Kosslyn and Shwartz's, allow for great detail but miss out on the higher level abstractions, such as how physical objects are used. Encoding an object's purpose for use in a task-oriented environment is also a shortcoming of most current systems (OPUS and Barbara Grosz's task domain are notable exceptions). To understand complex physical objects fully, a need exists for processing techniques from each of the three classes: visual, utilitarian, and fundamental physical property.

IPP—Generalization and Memory

Assuming that the representation problems for a single complex physical object have been solved, we are now faced with the problem of organizing many such descriptions in an intelligent manner IPP (Lebowitz, 1980; Lebowitz, 1983b; Lebowitz, 1983c) and similar programs demonstrate how generalization can be used to achieve this end.

One common feature that most of the preceding programs (including MARGIE, GUS and OPUS) have is their use of frames[3] as knowledge structures. IPP is no exception. The frame structures used in IPP are forms of MOPs (Memory Organizational Packets) (Schank, 1980; Schank, 1982). MOPs are very high-level representational structures that organize scenes, scripts, and supplemental data into a coherent picture of an event. In this sense, MOPs work much like plans, but are more powerful and allow for dynamic script building. That is, the scripts that a MOP employs need not be a permanent part of the MOP. They can be modified, deleted, or repositioned within the MOP in order to reflect a better understanding of what the MOP is encoding. The dynamic nature of MOPs is an important element in a understanding system that uses them. This ability to restructure memory dynamically is the principal difference between MOPs and simple frames or partitioned semantic nets. By allowing for a representation scheme that can reorganize its own data, MOPs go far beyond the capabilities of static frame-based processing techniques.

[3]The term *frames* is used here to include any representation scheme which groups data into logical blocks and provides for individual access to the slots within these blocks It should be noted that the frames used in IPP are equivalent to those used in MARGIE or GUS in only the broadest sense

IPP uses MOPs as long-term memory representations of stories it reads about terrorism. Its approach is to scan stories from wire services and newspapers and understand them in terms of what information it has gathered from previous stories. The use of MOPs residing in memory in understanding the current input text is one of the important features of this program. IPP recognizes similarities and differences between events stored with MOPs it has in memory and then uses this observational data to build other MOPs that can be used as stereotypical knowledge. This process is a form of generalization.

To exemplify this type of generalization, consider the following taken from (Lebowitz, 1980):

UPI, 4 April 1980, Northern Ireland

"Terrorists believed to be from the Irish Republican Army murdered a part-time policeman... "

UPI, 7 June 1980, Northern Ireland

"The outlawed Irish Republican Army shot dead a part-time soldier in front of his 11-year-old son in a village store Sunday."

From these stories, IPP would made the generalization:

"Terrorist killings in Northern Ireland are carried out by members of the Irish Republican Army "

This generalization is made possible by a comparison of MOP slot fillers. The stereotypical MOP for a terrorist killing event has slots for place and actor, among others such as victim, method, and the like. The program assumes that all facts it knows about are relevant to compare After forming this generalization, IPP will use it to make inferences while reading other stories. Thus, if a new story about a terrorist act in Northern Ireland came across the UPI wire, and no mention of who committed the act was made, then IPP would assume that the Irish Republican Army was responsible. This sort of assumption is an example of default processing mentioned in the context of GUS, but carried out at a higher level of representation and dynamically.

To get a better idea of what MOPs can represent, consider the MOP skeleton (adapted from (Schank, 1982)), as shown in Table 1.

Here we see that the M-AIRPLANE MOP is composed of several scenes, which in turn contain scripts, which are complex CD descriptions of a simple activity. That is, scenes are at a higher level of representation than are scripts, and MOPs are at a still higher level. This diagram shows only what the DRIVE-TO-AIRPORT scene expands to. All the other scenes have some script representation as well. Although MOPs are a form of frame, they are far removed from something as simple as the date frame exemplified in the GUS description.

IPP correctly reads and understands hundreds of separate stories. The strong performance of this program is partially due to the fact that it reads only a limited domain

LEVEL OF REPRESENTATION	CONTENT OF REPRESENTATION
MOP	M-AIRPLANE
scene	(PLAN TRIP)
scene	(GET MONEY)
scene	(CALL AIRLINE)
scene	(GET TICKETS)
scene	(DRIVE TO AIRPORT)
script	FIND KEYS
script	PLAN ROUTE
script	LOAD LUGGAGE
script	etc.
•	•
•	•
•	•
•	•

Table 1

of stories. By using a small number of stereotypical MOPs that are initially input by the programmer, the generalization process is made somewhat easier. Only a relatively small number of similarities and/or differences among MOPs need be analyzed.

Lebowitz's work is not the only recent research into using generalization processes in conjunction with natural language understanding systems. CYRUS (Kolodner, 1980), a program developed concurrently with IPP, uses a similar generalization process in order to understand events concerning the activities of individuals (Cyrus Vance was the prototype) They differ in the way that they make use of knowledge gained through generalization. IPP uses its inferred knowledge in order to help itself in understanding further input text, while CYRUS answers user questions by employing this knowledge to help it reconstruct episodes in memory. These reconstructed episodes can be thought of as a re-creation of the mental state that the understanding system had while reading the original text.

Recent work by Kathleen McCoy, on a program called ENHANCE (McCoy, 1982) uses generalization as a way to restructure an existing data base. It subdivides entity classes in a data base according to a set of world knowledge axioms. These sub-classes form a structured hierarchy that is tailored to a particular use by the information contained within the axioms. The enhanced data base is then used by a text-generation program to provide intelligent responses to user queries. Thus, the work done by the generation program is simplified because most of the inferencing it needs to perform has already been pre-computed by ENHANCE.

Much work has been done in psychology in human cognitive modeling (see Kintsch, 1977 for an overview). As a consequence of this work, and others', many different ways of thinking about generalization have emerged. Some researchers prefer to think that all learning is in some way generalization, while others reserve the term *generalization* for a specific cognitive process, such as building stereotypes from a limited number of examples. Concept building and rule learning (Stolfo, 1980) are phrases that are often used to describe generalization processes (Mitchell, 1982 and Michalski, 1983 provide useful classifications of learning research).

Rule learning is the term that Tom Mitchell applies to his notion of version spaces (Mitchell, 1977). Version spaces refers to a representation/generalization method for finding the set of all possible rules that can account for the outcome of some particular action given the results of this action. They are used in a program called Meta-DENDRAL (Buchanan and Mitchell, 1978) which learns rules for use in the production system that DENDRAL (Lindsay *et al.*, 1980) uses. Although this program does not do natural language processing, it uses a dual form of generalization based on the version space method. It can produce production rules that are as general as possible, but still fully account for the observed data, or it can produce very specific rules, or both. This type of multi-level generalization ability seems potentially quite useful in NLP applications, but has yet to be implemented.

Generalizations based on high-level representations, such as those that MOPs encode, differ from learning driven by simple semantic nets. Winston's ARCH program could learn the concept of an arch by analyzing several correct and erroneous examples. It did this by studying the form of the semantic net that represented each example. IPP makes its generalizations by using the content of MOPs. This form versus content distinction is not clear-cut. Both semantic nets and MOPs use links to encode knowledge, and both use nodes (frames) to hold data. The difference lies in the realization that MOPs encode their low-level knowledge in frame slots and their high-level knowledge as links, while semantic nets store all their data as links.

Knowledge gained through generalization is certainly of this high-level type. IPP uses this knowledge as a way of structuring its memory. That is, the act of forming generalizations actually results in a different overall memory structure (only if a new concept is created) Furthermore, the system can use its newly acquired knowledge to help it understand additional input during the parsing process. This type of representation/generalization integration is extremely powerful as the basis for a NLP program that needs to deal with varied levels of representation.

Summary. IPP, CYRUS, and ENHANCE represent recent developments in using generalization as an active organizational mechanism for knowledge. IPP can read hundreds of stories about terrorism and understand them in terms of the previous knowledge it has acquired. The use of MOPs, along with the ability to structure them dynamically, is the key to this learning process.

The MOP form of knowledge representation is very versatile. Many levels of description can be encoded within a hierarchy of conceptual frames. This ability seems to be a necessity for a physical object understanding system that hopes to handle complex objects. Complex physical objects are often described by a series of part, sub-part relations. Thus, a representation scheme would need to encode the whole object, its major components, the parts of the major components, and so forth.

The problems that arise in static frame-based representation schemes, having to do with their inability to reorganize the data that they encode easily, have largely been solved by dynamic MOP-based systems. IPP and CYRUS have demonstrated the usefulness of integrating generalization with representation to form adaptable understanding programs. This integration is a consequence of the use of generalization processes as a way of structuring data.

MOPs and generalization offer a viable approach for building representation/generalization systems that seek to understand knowledge in a complex domain.

RESEARCHER—A synthesis

IPP demonstrated how a generalization-based memory can be used to organize a large number of event representations into a unified structure. The events used were nonstructured frame (MOP) descriptions of terrorism stories. They did not have sub-events, sub-sub-events, and the like. Unlike these events, complex physical object descriptions are hierarchically structured. RESEARCHER (Lebowitz, 1983d; Lebowitz, 1983a) integrates representation and generalization in a similar fashion (as IPP did) to form a robust understanding system for complex, hierarchically structured object descriptions provided by the patent abstracts it reads.

RESEARCHER, which is still under development, functions by parsing patent abstracts into a representation structure based upon *memettes*. Memettes are a type of frame similar to MOPs, but are specifically designed to be used in building hierarchical structures. Each memette represents a part of a complex object at some level of detail. That is, a single memette can represent an entire object (a disc drive, for example), or it can be used to encode the description of a unitary object (such as a particular screw in the disc drive). A memette that represents an object that contains other objects as parts is called composite.

The physical object representation scheme that RESEARCHER uses is based on two principles: physical objects can be primarily represented as a hierarchical structure, and this hierarchical structure is augmented by relations connecting arbitrary nodes in the hierarchy. For example, an automobile can be thought of as a hierarchy of components. That is, it has a body, a chassis and an engine; the engine has a carburetor, a crankcase, and so forth. Furthermore, the parts are related by various positional references (*e.g.*, the body is on top of the chassis).

RESEARCHER uses a canonical, CD-like scheme for specifying the inter-component physical relations. Each relation used in the parts hierarchy is described as a combination of various property-value pairs. Five primitive properties, used in combinations, suffice to reduce natural language relation phrases into a closed set. Table 2 (see Wasserman and Lebowitz, 1983, for a full account of this scheme) shows these properties and some typical words that are strongly associated with each. Many words and phrases often require two of these five for an accurate description. For example, the phrase "on top of" would need both the contact and location properties in its encoding. This scheme is an example of a combination of all three types of physical object representation approaches.

Using this component/relation scheme, RESEARCHER parses patent abstracts into memette structures. The memette frame slots filled by the parser include: TYPE—either *unitary* or *composite*; STRUCTURE—a list of relations, if composite, or a description of the object's shape, if unitary; and COMPONENTS—a list of the memette's parts. The following text is taken from a patent abstract about an enclosed disc drive. This text and its representation are taken, in part, from (Lebowitz, 1983a). Disc drive patents form RESEARCHER's initial domain.

PROPERTY	DESCRIPTION	VALUE(S)
•*distance*	used for relations that refer to disjoint objects (*e.g.*, near, remote)	a single integer from 0 to 10. 0—close, 10—far
•*contact*	describes the degree to which objects are in contact with each other. (*e.g.*, touching, affixed)	a single integer from −10 to +10. −10 = strongly forced together +10 = touching, but being forced apart
•*location*	indicates in which direction an object is located relative to another. (*e.g.*, above, left)	a 2D or 3D angular identification along with a reference frame indication.
•*orientation*	describes the relative orientation of two objects. (*e g.*, parallel, perpendicular)	a 2D or 3D angular identification.
•*enclosure*	used for relations which describe objects, where one is either fully or partially enclosed by another (*e.g.*, encircled, cornered)	*full* or *partial* plus a shape description of the interface between the enclosed and the enclosing objects.

Table 2.

Enclosed Disc Drive Having Combination Filter Assembly: A combination filter system for an enclosed disc drive in which a breather filter is provided in a central position in the disc drive cover and a recirculating air filter is concentrically positioned about the breather filter

A possible memette structure for this patent is:

```
(NAME   enclosed-disc-drive-with-filter
 TYPE   composite
 COMPONENTS   (enclosure disc-drive)
 STRUCTURE   ((SURROUNDS enclosure disc-drive)))
(NAME   enclosure
TYPE.  composite
 COMPONENTS   (cover case)
 STRUCTURE   ((ON-TOP-OF cover case)))
(NAME.  case
 TYPE   unitary
 STRUCTURE   (box open-on-top))
(NAME   disc-drive
 TYPE   composite
 STRUCTURE   unknown)
(NAME   cover
 TYPE   composite
 STRUCTURE ((SURROUNDS[centrally] cover breather-filter)
        (SURROUNDS[centrally] recirculating-air-filter
        breather-filter)))
(NAME   breather-filter
 TYPE   unknown)
(NAME.  recirculating-air-filter
 TYPE:  unknown)
```

In order to integrate generalization with representation, each memette contains an additional slot that allows it to be connected to other memettes forming a generalization hierarchy. The VARIANT-OF slot is essentially an IS-A link that allows for inheritance of information. The hierarchy created by the use of this slot allows for generalizations to be made at all levels in the component hierarchy. Consider the two representations of similar enclosed disc drives (also taken from [Lebowitz, 1983a]) shown in Figure 1.

Note that the generalized *enclosure#* has a *cover#* on-top-of something and that the generalized *enclosed-disc-drive#* has both a *disc-drive#* and an *enclosure#* . Thus generalizations have been made at the top level of the component hierarchy (*i.e.*, *enclosed-disc-drive#*) and at lower levels (*i.e.*, the *enclosure#*). By organizing all its data in this way, RESEARCHER can act as an intelligent information system.

The justification for this process of generalizing at all levels is best explained by Herbert Simon's idea of near-decomposability (Simon, 1981). A nearly decomposable system is one in which the interaction among the components that make up the system is weaker than the glue the keeps any one component intact. The contention is that systems can evolve in complexity by making use of this property and that a hierarchy is the natural form into which a complex system usually develops. Thus, sub-parts of any hierarchy become stable as the system grows. This indicates that stable components are important and should be recognized as being so by an intelligent understander of such systems. The understanding of hierarchies is discussed further in (Wasserman, 1984).

RESEARCHER is not the only system that has tried to represent component hierarchies within a generalization hierarchy. NETL (Fahlman, 1979) uses both PART-OF and IS-A links in representing knowledge in a highly parallel computation system. Although the interaction between the component and generalization hierarchies is apparent in NETL, it is not used to advantage in the encoding scheme.

Another program (unnamed) (Hayes, 1977) employed a categorization hierarchy that classified animal body-part hierarchies. Thus, a generalization (IS-A) hierarchy was used to classify PART-OF hierarchies. This work, although somewhat similar to RESEARCHER's methods in that it combined generalization and representation in the same functional way, required a human expert to implement the knowledge structures and modify them as needed.

A few observations have become clear while working on RESEARCHER. A hierarchy understanding system that is to be used for real-world knowledge acquisition about physical objects and be truly intelligent needs to have the ability to automatically build representations (no human intervention); dynamically reorganize memory to better reflect learned knowledge; make use of the near-decomposability of hierarchies to store information in a compact form; recognize and exploit the interrelationship of the representation language with the generalization method use primitives of human cognition.

Summary. RESEARCHER carries the idea of generalization-based memory into the domain of physical object understanding. Frames are shared in two orthogonal hierarchies: the components hierarchy and the generalization hierarchy. This permits objects to be represented concisely and organized according to what they have in common.

The scheme used to encode relations among objects is based on semantic primitives that serve to reduce natural language relation expressions into a closed class, in much the same way as CD does for actions.

Conclusion

The six programs presented above by no means completely span all the NLP programs that have contributed to the progress made in knowledge representation and generalization. They do, however, form a representative set of programs that demonstrate the kind of research into physical object understanding and generalization systems that has taken place in the past ten years or so.

The large number of programs that are intended to investigate the benefits of some particular knowledge structure is, of course, necessary. Obviously, one of the first consid-

```
                    enclosed-disc-drive1
                       /            |
     --  disc-drive1 --     enclosure1 -----
     |      |     |    |  /   on-top-of      \
  motor#    |   disc#  | cover# -------> support-member#
       spindle#   r/w-head#
```

```
                    enclosed-disc-drive2
                       /            |
     --  disc-drive2 --     enclosure2 ------------
     |      |     |    |  /   on-top-of  \    /    \
  motor#    |   disc#  | cover# ----> base# /        \
        spindle#  r/w-head#                /   surr    \
                                     b-filter# ----> r-filter#
```

RESEARCHER generalizes what these two instances have in common to arrive at the structure:

```
                    enclosed-disc-drive#
                       /            |
     --  disc-drive# --     enclosure# --------
     |      |     |    |  /   on-top-of        \
  motor#    |   disc#  | cover# ---------> <       >
       spindle#   r/w-head#
```

Figure 1.

erations in any AI system is how to represent information. Thus, many researchers concentrate on developing a good representation system, often with the intent of using it in a full natural language comprehension program at some later time.

This argument goes a long way in explaining the dearth of programs that make use of a generalization process. Only a few systems, such as IPP, RESEARCHER, CYRUS, NETL, and ENHANCE, focus attention on the use of generalization as an understanding mechanism. It seems that using generalization as the basis of, instead of as an add-on to, of a NLP program is a good way to proceed.

The brief history of NLP programs presented here has demonstrated that in a fairly short time great progress has been made. The next ten years should see rapid growth, particularly in the area of applying generalization principles to natural language processing programs.

References

Barr, A. & Feigenbaum, E. A. (1981) *The Handbook of Artificial Intelligence.* Los Altos: William Kaufman, Inc.

Bobrow, D. & Collins, A. (1975) *Representation and Understanding: Studies in Cognitive Science.* New York: Academic Press.

Bobrow, D. G. & Winograd, T. (1977) An Overview of KRL, a Knowledge Representation Language. *Cognitive Science* 1:3-46.

Bobrow, D. G. & Winograd, T (1977) *Experience with KRL-0 one cycle of a knowledge representation language. IJCAI-5*:213-222.

Bobrow D. G *et al.* (1977) GUS, A Frame-Driven Dialog System *Artificial Intelligence* 8:155-173

Brachman, R J. (1979) *Taxonomy, descriptions and individuals in natural language processing* Association for Computational Linguistics: 33-38

Buchanan, B. G & Mitchell, T M. (1978) Model-directed learning of production rules. In D. A. Waterman & F. Hayes-Roth (Eds), *Pattern-Directed Inference* New York: Academic Press.

Charniak, E. (1977) Ms Malaprop, a language comprehension program. *IJCAI-5*:1-7.

Chomsky, N (1965) *Aspects of the Theory of Syntax* Cambridge: MIT Press

Cohen, P. R. & Feigenbaum, E. A. (1982) Vision. In P. R. Cohen & E. A. Feigenbaum (Eds.), *The Handbook of Artificial Intelligence* Los Altos: William Kaufman Inc

Cullingford, R (1978) Script application: Computer understanding of newspaper stories Tech. Rep. 116, Department of Computer Science. Yale University.

de Kleer, J & Brown, S (1983) *The origin, form and logic of qualitative physical laws. IJCAI-8*:1158-1169.

Fahlman, S. E. (1979) *NETL, a System for Representing and Using Real-world Knowledge.* Cambridge, : MIT Press

Fillmore, C (1968) The case for case In E. Bach & R. Harms (Eds.), *Universals in Linguistic Theory,* New York: Holt, Rinehart and Winston.

Findler, N. V. (1979) *Associative Networks* New York: Academic Press

Forbus, K. D. (1981) *Qualitative reasoning about physical processes. IJCAI-7*:326-330.

Grosz, B (1977) The representation and use of focus in dialogue understanding. Tech. Rep. 151, Stanford Research Institute

Hayes, P J. (1977) *On semantic nets, frames and associations. IJCAI-5*:99-107

Hayes, J. (1978) Children's Visual Descriptions. *Cognitive Science* 2:1-15.

Hayes, P. J. (1979) The naive physics manifesto. In D. Michie (Ed.), *Expert Systems in the Microelectronic Age.* Edinburgh: Edinburgh University Press.

Hendrix, G G. (1979) Encoding knowledge in partitioned networks. In N. V. Findler (Ed), *Associative Networks*, New York: Academic Press.

Kintsch, W. (1977) *Memory and Cognition.* New York: John Wiley and Sons.

Kolodner, J. L. (1980) Retrieval and organizational strategies in conceptual memory: A computer model. Tech Rep 187, Department of Computer Science. Yale University

Kosslyn, S. M. & Shwartz, S. P. (1977) A Simulation of Visual Imagery. *Cognitive Science* 1:265-295.

Kuipers, B. J (1975) A frame for frames: Representing knowledge for recognition In D. Bobrow & A Collins (Eds.), *Representation and Understanding: Studies in Cognitive Science.* New York: Academic Press.

Lebowitz, M (1980) Generalization and memory in an integrated understanding system Tech Rep. 186, Department of Computer Science. Yale University

Lebowitz, M (1983) RESEARCHER: An overview *AAAI-83 Proceedings*, Washington, D.C., 232-235.

Lebowitz, M. (1983) Memory-based Parsing. *Artificial Intelligence* 21:285-326.

Lebowitz, M (1983) Generalization from Natural Language Text *Cognitive Science* 7(1):1-40.

Lebowitz, M. (1983) Intelligent information systems ACM SIGIR, Washington, DC

Lehnert, W G (1978) Representing physical objects in memory Tech Rep. 131, Computer Science Department. Yale University

Lehnert, W. G. & Burstein, M H. (1979) The role of object primitives in natural language processing. Tech. Rep 162, Computer Science Department. Yale University

Lindsay, R , Buchanan, B. G , Feigenbaum, E A. & Lederberg J. (1980) *DENDRAL* New York: McGraw-Hill

McCoy, K. F. (1982) Augmenting a data base knowledge representation for natural language generation Association for Computational Linguistics, Toronto, Ontario, Canada.

Michalski, R. S. (1983) A Theory and Methodology of Inductive Learning. *Artificial Intelligence* 20:111-161

Miller, G A (1975) Comments on lexical analysis *Theoretical Issues in Natural Language Processing* 1:34-37.

Minsky, M (1975) A framework for representing knowledge In P. H Winston (Ed), *The Psychology of Computer Vision*, New York: McGraw-Hill.

Mitchell, T M (1977) *Version spaces: A candidate elimination approach to rule learning IJCAI-5*:305-310

Mitchell, T. M (1982) Generalization as Search *Artificial Intelligence* 18:203-226

Novak, G. S (1977) *Representations of knowledge in a program for solving physics problems IJCAI-5*:286-291

Quillian, M R (1968) Semantic memory In M. Minsky (Ed), *Semantic Information Processing*, Cambridge: MIT Press.

Riesbeck, C K & Schank, R C (1976) Comprehension by computer: Expectation-based analysis of sentences in context In W J. M Levelt & G. B. Flores d'Arcais (Eds.), *Studies in the Perception of Language*, Chichester, England: John Wiley and Sons. Also Tech. Rep. 78, Computer Science Department. Yale University

Roberts, R. B & Goldstein, I. P. (1977) The FRL manual. Tech Rep. 409, MIT AI Laboratory.

Rosch, E. *et al* (1976) Basic Objects in Natural Categories. *Cognitive Psychology* 8(3):382-437

Schank, R. C. (1972) Conceptual Dependency: A theory of natural language understanding. *Cognitive Psychology* 3(4):532-631.

Schank, R C. (1975) *Conceptual Information Processing.* Amsterdam: North Holland.

Schank, R. C. (1980) Language and Memory *Cognitive Science* 4(3):243-284.

Schank, R. C. (1982) *Dynamic Memory: A Theory of Reminding and Learning in Computers and People.* New York: Cambridge University Press.

Schank, R. C. & Abelson, R. P. (1977) *Scripts, Plans, Goals and Understanding* Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Schank, R & Burstein, M. (1982) *Modeling memory for language understanding.* Tech. Rep 220, Department of Computer Science. Yale University.

Schank, R. C & Riesbeck, C K (1981) *Inside Computer Understanding* Hillsdale, New Jersey: Lawrence Erlbaum Associates

Simon, H. A. (1981) The architecture of complexity In H. A Simon (Ed.), *The Sciences of the Artificial.* Cambridge: MIT Press

Small, S. (1980) Word expert parsing: A theory of distributed word-based natural language understanding. Tech. Rep. TR-954, Department of Computer Science University of Maryland

Stolfo S. (1980) Learning control of production systems Tech. Rept. 79-11-1, Department of Computer Science. Columbia University

Tennant, H (1981) *Natural Language Processing An Introduction to an Emerging Technology.* New York: Petrocelli Books, Inc.

Thorne, J. , Bratley, P & Dewar, H (1968) The syntactic analysis of English by machine. In D Michie (Ed), *Machine Intelligence 3*, New York: American Elsevier Publishing Company.

Wasserman, K. (1984) Understanding Hierarchically Structured Objects Tech. Rept 84-5-3, Department of Computer Science. Columbia University

Wasserman, K & Lebowitz, M (1983) Representing Complex Physical Objects Cognition and Brain Theory 6(3): 333-352

Wilensky, R (1978) Understanding goal-based stories Tech Rep. 140, Department of Computer Science Yale University.

Wilks, Y (1973) An artificial intelligence approach to machine translation. In R. C. Schank & K Colby (Eds), *Computer Models of Thought and Language*, San Francisco: W. H. Freeman Co

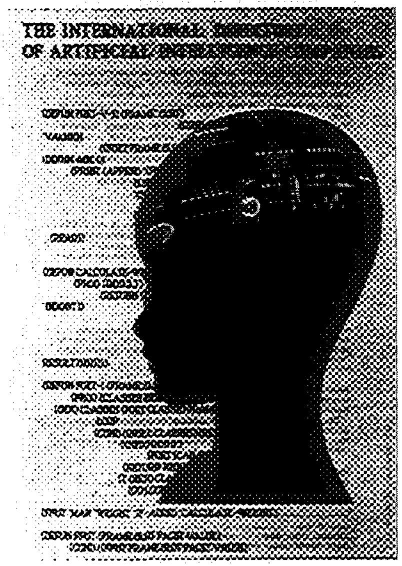
Wilks, Y. (1974) Natural language understanding systems within the AI paradigm: A survey and some comparisons Tech. Rep. 237, AI Laboratory, Stanford University

Wilks, Y (1975) An Intelligent Analyzer and Understander of English. Communications of the Association for Computer Machinery 18: 264-274

Winograd, T. (1972) *Understanding Natural Language* New York: Academic Press

Winograd, T (1975) Frame representation and the declarative/ procedural controversy In D. Bobrow & A Collins (Eds), *Representation and Understanding: Studies in Cognitive Science* New York: Academic Press

The first Directory of Artificial Intelligence Companies in the World!


THE INTERNATIONAL DIRECTORY
OF ARTIFICIAL INTELLIGENCE COMPANIES

The Directory represents a valuable source of information in the Artificial Intelligence business field. All the most important AI Companies are included together with their products, services, customers, staff, address, etc.
More than 85 companies and 150 products for Artificial Intelligence . The Directory is a quick reference to the AI world which tells you who does what, where and how (and in most cases for how much...).The Directory will save you an enormous amount of time because all you wanted to know about the AI business field is already there.
180 pages on real Artificial Intelligence.

CONTENTS:Foreword by prof. Bernard Meltzer - How to read the Directory - Artificial Intelligence: what are its Applications - <u>List of the main AI Companies in the world, with reference to their products, activities and plans</u> - List of Universities and Research Centres in USA, Europe and Japan - AI books and journals - AI associations and conferences - Glossary of AI terms.

Mail to: Artificial Intelligence Software srl
Casella Postale 198, I-45100 Rovigo (Italy)
Please send: The International Directory of Artificial Intelligence Companies. ($ 200 +$5 postage)*
() Payment enclosed () Institutional purchase order
Note:Orders from individuals must be prepaid.

NAME ______________________

ADDRESS ______________________

______________________ **Zip Code** ________

*** For orders placed on the forms provided at AAAI-84 and ECAI-84 Conferences, the reduced price indicated there will apply.**

Winston, P. H (1977) *Artificial Intelligence.* Reading, Massachusetts: Addison Wesley.

Woods, W. A. (1970) Transition Network Grammars for Natural Language Analysis Communications of the ACM 13: 591-606.

Woods, W. A. (1975) What's in a link: Foundations for semantic networks. In D Bobrow & A. Collins (Eds.), *Representation and Understanding: Studies in Cognitive Science.* New York: Academic Press.

Woods, W. A. , Kaplan, R & Nash-Webber, B. (1972) The lunar sciences natural language information system. Tech. Rep 2378, Bolt Beranek and Newman, Inc.

Bibliographic Notes

Several of the references listed are surveys of work done in NLP and related areas.

A good overview of NLP is (Tennant, 1981). This book covers the history and recent developments in all NLP sub-fields. Many examples are provided by means of case studies of individual programs. Further descriptions of several of the systems mentioned in this article can be found.

The Handbook of Artificial Intelligence, Volume 1 (Barr and Feigenbaum, 1981) is largely devoted to NLP and knowledge representation It also provides case studies of various programs. Although not quite as readable as Tennant's book, it does offer excellent references

Chapter 6 of (Kintsch, 1977) is an interesting survey of psychological research in language comprehension

Much of the work mentioned in this article was done by the AI project at Yale University. A good overview of the IPP, OPUS, PAM, SAM, ELI, and CYRUS programs can be found in (Schank & Burstein, 1982). Also a brief description of MOPs is given A detailed description of the earlier programs (PAM, SAM, and ELI) can be found in (Schank and Riesbeck, 1981).

Certain books and conference proceedings are particularly rich in articles pertaining to the issues raised in this paper.

The proceedings from *IJCAI-5* contain many papers on representation and generalization. These include: KRL (Bobrow & Winograd, 1977b), ISAAC (Novak, 1977) and version spaces (Mitchell, 1977). Other papers not referred to here, but of interest, can also be found.

Representation and Understanding: Studies in Cognitive Science (Bobrow and Collins, 1975), is the name of the book containing (Kuipers, 1975; Winograd, 1975; Woods, 1975). These papers and others give an excellent in-depth discussion of knowledge representation. In particular the semantic network and frame formalisms are explored.

Several well written papers about representational issues can be found in *Associative Networks* (Findler, 1979) Most of the works contained here are concerned with semantic networks, but not exclusively Many of the authors referenced in this paper have contributed sections of *Associative Networks*, including (Hendrix, 1979)