5th AAAI-Sponsored Knowledge Acquisition for Knowledge-Based Systems Workshop
Banff, Canada, November, 1989.

Verification of Frame and Semantic Network Knowledge Bases

Daniel E. O'Leary

Graduate School of Business
University of Southern California
Los Angeles, California 90089-1421
213-743-4095

ABSTRACT

Virtually all previous research in verification of knowledge bases has concentrated on rule-based systems. Since most systems and shells developed to-date have concentrated on knowledge representation using rules, this seems appropriate. However, increasingly, there are systems being built using other forms of knowledge representation, such as frames and semantic nets. As a result, there is a need to develop verification approaches for these types of knowledge representations. Verification of frames and semantic networks is investigated from a domain independent approach that focuses on the characteristics of the knowledge representation. Examples are used to illustrate each approach. Implementation considerations also are discussed.

1. INTRODUCTION

Verification was defined by Adrion et al. (1982) as "the demonstration of the consistency, completeness and correctness of the software." The purpose of this paper is to develop approaches to the verification of knowledge represented as either frames or semantic networks.

In knowledge-based systems, many verification efforts are aimed directly at verifying the knowledge base portion of the software. This results from both the design and development processes, and the software used with expert systems. Typically, from a design and development perspective, the knowledge base is treated as an independent entity. In addition, the existence of expert systems shells allows the user to assume that the inference engine produces appropriate inferences, so that the focus can be on the development and verification of a knowledge base.

If assumptions are made about the knowledge base (e.g., assume it is in the form of rules), then the pursuit of consistency, completeness and correctness take greater structure. The more restrictive the assumptions on the knowledge representation or domain, the more structured the verification efforts can be made. Thus, verification can be, to a large extent, a function of specific knowledge representation technology on which the system is based or the domain in which the system resides.

For example, there has been substantial research on verification of rule-based systems. Those efforts have focused on the structure of the knowledge contained in the rules (Nazareth, 1988 and Nguyen et al., 1985 and 1987) and the structure of the weights on those rules (O'Leary, 1990).

1.1 Use of Frames and Semantic Networks

Although most of the expert or knowledge-based systems that have been developed likely are rule-based, increasingly systems are being developed using alternative knowledge representation schemes. For example, frames and semantic networks (nets) are receiving greater use in many application systems (e.g., Willingham and Ribar, 1988). In part this is because the systems that are being built are more complex, so that alternative forms of knowledge representation are necessary. Since systems increasingly employ other forms of knowledge representation, there is interest in establishing verification of other types of knowledge-based systems.

Since most systems that employ frames or semantic networks do not use shells that embed verification efforts, the verification often is the hands of the developer. In addition, as greater effort is made to develop shells that employ alternative knowledge representations there is a need to incorporate verification devices in those shells. Unfortunately, there is little literature established on the verification of frames or semantic networks. Thus, the purpose of this paper is to elicit approaches to assist in the verification of frame and semantic network knowledge bases.

1.2 Domain Dependent and Independent Approaches

There are at least two ways to characterize the verification process: domain dependent and domain independent (e.g., Nazareth, 1989). This paper is aimed primarily at domain independent approaches.

Domain dependent verification attempts to exploit what is known about the domain, in the form of meta-knowledge, to assist in the verification process. Such meta-knowledge is either embedded in the system or accessed by exploiting human's knowledge of the domain. Some of the first verification approaches were of this nature (Davis, 1976).

Domain independent verification uses knowledge about the particular form of knowledge representation to assist in the verification process, but does not employ any knowledge about the domain. For example, in the case of rules, typically it is inappropriate to have cycles in the knowledge base. Domain independent approaches use this knowledge about the knowledge representation to assist in the verification process.

Outline of this Paper

This paper proceeds as follows. Section 2 summarizes some of the characteristics of frames and semantic nets and provides definitions used later in the paper. Section 3 investigates domain dependent verification of frames and semantic networks. Section 4 analyzes domain independent verification of frames and semantic networks. Section 6 provides a brief summary of the paper.

2. FRAMES AND SEMANTIC NETS

Although rules probably are the most frequently used form of knowledge representation, frames and semantic networks are among the most used of other forms of knowledge representation. This section provides a brief summary of some of the primary characteristics of frames and semantic nets.

2.1 Frames

As noted by Minsky, the developer of the theory of frames,
in 1975,

> A frame is a data-structure for representing a stereotyped situation like being in a certain kind of living room or going to a child's birthday. Attached to each frame are several kinds of information. Some of this information is about how to use the frame. Some is about what one can expect to happen next. Some is about what to do if these confirmations are not confirmed.
>
> We can think of a frame as a network of nodes and relations. The 'top' levels of a frame are fixed and represent things that are always true about the supposed situation. The lower levels have many terminals -- 'slots' that must be filled by specific instances or data. Each terminal can specify conditions its assignments must meet. (The assignments themselves are usually smaller 'subframes.') Simple conditions are specified by markers that might require terminal assignment to be a person, an object of sufficient value, or a pointer to a subframe of a certain type. More complex conditions can specify relations among the things assigned to several terminals.

As an example of a typical frame, Winston (1979) (See figure 1) displays a frame that includes title information of the frame, action information, actor information, object information, source information, destination information, results storage (that lead to a state change frame) and subprocesses information. In each case, except the title, there can be a number of slots to contain information about, e.g., subprocesses. As a result, slots may have considerable information attached to them.

Frames also are hierarchical, so that diagrams of frame-based systems take on a tree-like structure (figure 2) or an acyclic network structure (figure 3). At each level of the tree the frames are likely to be of the same basic design.

Structurally, the "top" frame can be referred to as the root frame. While, the "bottom" frames can be referred to as the terminal frames. Frames between the top and bottom frames can be referred to as intermediate frames. In figure 2, the root frame is the room frame and the terminal frames are the picture frame, the window frame and the door frame.

In some cases the frames lead to other frames, as in figure 1. In some knowledge bases those frames establish parallel structures. For example, in figure 3 each of the group frames leads to two object frames, which in turn lead to an object frame.

2.2 Semantic Networks

Originally semantic nets were designed as a representation structure of the meanings of words. In semantic networks, knowledge is represented as a set of nodes and directed arcs, where the arcs are used to represent relationships between the nodes and the nodes represent objects and characteristics of those objects. Example semantic nets are located in figures 4 and 5.

In semantic networks, there are three types of nodes: root nodes (no arcs leading in -- only arcs leading out); intermediate nodes (must arcs leading in and out); and terminal nodes (only arcs leading in -- no arcs leading out). In some cases there may be little or no parallel structure (figure 4), while in other cases there may be substantial parallel structure (figure 5). Structurally, semantic nets are either trees (figure 4) or acyclic networks (figure 5). Network forms of semantic nets would seldom, if ever, be designed to allow a cycle to occur. Further, in some cases, it may be possible to set up an a priori precedence among types of arcs that would occur. For example, it may be possible to state that an "isa" arc always proceeds a "color" arc. In the examples, no such relationships are established.

3. DOMAIN DEPENDENT VERIFICATION

Domain dependent approaches assume that there is some meta- knowledge available that allows the system to examine resulting knowledge for correctness, completeness and consistency. For example, in the case of figure 2, such domain knowledge could include feasible information for different rooms in a house.

Consider the case of living rooms. If some slot information were to include a stove or a toilet then the verification should ascertain that there was an error, since these are seldom found in living rooms. Similarly, if the "picture" frame (or a "picture frame") was empty then verification could suggest that there was an error of omission, since most living rooms have pictures in them. As noted by Nazareth (1989, p. 263), "verification is not performed, per se, but is achieved through debugging of erroneous system recommendations."

3.1 Integration with Knowledge Acquisition

Since domain dependent verification employs knowledge about the domain, it can be treated as a part of the knowledge acquisition process. Domain dependent approaches can be embedded in the knowledge acquisition processes and tools that support knowledge acquisition. In addition, as these systems accumulate knowledge about the domain, they can use that knowledge to verify new knowledge solicited as part of the process.

For example, consider the case of automated knowledge acquisition, where the expert uses the system and the system assists in the acquisition of that knowledge. For verification purposes, the user can function as one source of the meta-knowledge or the meta-knowledge can be embedded in the system or both. Further, the knowledge acquisition approach may be generic or domain dependent.

Broad based opportunities exist for the use of domain knowledge in verification, including domain knowledge, problem type knowledge (diagnostic), promulgated problem solution methods (smart questionnaires), etc. Virtually any knowledge or decision structure that has a known structure can be used to determine if the ascertained structure is similar to the expected structure. These choices are illustrated in figure 6.

Source of Verification Meta-knowledge

Human	System	/Knowledge /Acquisition
A	B	Domain Independent
C	D	Domain Dependent

Figure 6

Meta-knowledge and Verification

A comprehensive summary of many different machine-based knowledge acquisition approaches is provided in Boose (1989). Most of those systems are generic, so that they can be used for many different domains. For example, AQUINAS and KITTEN use hierarchically-structured repertory grid interviewing. Most of those systems would thus fall into either category A or B. Other systems such as ID3 are machine learning-based systems, and would fall in category B.

In both types of systems (A and B), the underlying methodological approach could be used in the verification process. For example, in the case of ID3, if the system is unable to classify each of the cases then that can indicate inconsistency in the data and knowledge. Similar constraints resulting from problem methodology can be developed from other approaches, such as decision trees.

A few of the systems discussed by Boose are domain dependent, falling into categories C and D. For example, LEAP uses apprenticeship learning to learn steps in VLSI design, MUM is concerned with medical problems and STUDENT acquires knowledge for the statistical consulting domain. These systems contain meta-knowledge that potentially could be useful for domain dependent verification. In addition, some systems make assumptions as to the type of problem to be solved (e.g., diagnostic problems), thus, providing additional structure for verification.

In any of A through D, the system can be adapted to verify based on the knowledge acquired, either by constantly relating it back to the newly acquired knowledge or by comparing to existing knowledge. In such comparisons either the user's or the system's domain knowledge can provide the basis of comparison. If the knowledge is there -- use it.

3.2 Limitations

Domain dependent approaches have some limitations. First, with a domain dependent approach, the meta-knowledge of such a system also requires verification. Even in the case of human users, potential problems such as inconsistency can allow errors to permeate the system. Second, if the meta knowledge required is not stable, then the verification portion of the system could require frequent updating. This could lead to a substantial system cost. Such, domain dependent approaches are likely to be particularly useful in large or ongoing systems, where such meta knowledge may be stable and where such a system could receive frequent use. These factors could mitigate system costs. Third, even though the system may have domain expertise, compared to the human expert the system's knowledge may be shallow. As a result, domain dependent approaches may provide limited return.

4. DOMAIN INDEPENDENT VERIFICATION

Domain independent verification uses knowledge about frames and semantic nets, as forms of knowledge representation in general, to assist in the verification process. Accordingly, the purpose of this section is to elicit that kind of knowledge that can be helpful in verification efforts.

It is important to note that the tests generated to verify a system are not guarantees that the system is one hundred percent correct. Instead, they become ways to determine when there are ascertainable errors of the type delineated here.

4.1 Consistency

Consistency refers, in part, to the names or labels used in the knowledge representation. Since humans have a tendency to make errors (e.g., in typing) or call the same thing by different names, procedures need to be adopted to ensure that the names and and contents are consistent. Consistency also can refer to consistent implementation of parallel structures.

Frames. In frames, consistency is a concern with the names given to the frames, frame slots and contents. One approach to eliminate errors deriving from inconsistency is to require the development of different lists for frame names, slot names and slot contents. System construction and use of frames would then require all choices to be from those lists. This would eliminate errors caused by alternative names, spellings, etc. Thus, part of the knowledge acquisition process is settling on such details.

For example, as noted in figure 2, the frame name is "room frame." This approach could require the construction of a frame by that name only if the frame name were given before the construction of the frame. In addition, the system would constrain use of that name to a single frame. Further, construction of, or reference to a frame would require choosing that frame's name from a list.

Consistency also refers to parallel treatment of parallel structures. If the knowledge base employs a set of parallel structures, as illustrated in figure 3, then the system can request that the user elicit parallelism structures prior to entering knowledge bases. When the system is aware of those parallel structures, their correctness can be monitored.

Allowing parallel structures also can facilitate capture of knowledge by the system. Rather than requiring a piece-by-piece structuring of the network of frames, allowing copying of existing parts of the network make the job of knowledge elicitation easier.

Semantic Networks. In semantic nets, consistency is an issue in the names given to arcs and nodes, e.g., "isa" and "chair." In order to eliminate multiple names being assigned to the same object, e.g., rather than calling a "chair" a "stool," the system can require that a list of eligible node and arc names be developed before those names are entered into the knowledge base. As in the case of the frames, construction of the network would require use of a name from one of those lists. Further, in a manner similar to that discussed for frames, parallel structure can be accounted for by the system when it is aware of its existence.

4.2 Redundancy

Redundancy refers to the introduction of potential undesirable duplication of knowledge into the knowledge base.

Frames. Redundancy can occur in at least four different ways, redundant frames, redundant slots within a frame, redundant content within a frame and redundant connections with other frames. There are at least two approaches that could find use: preventing and detecting redundancies.

A preventive approach would allow a single use of a frame name. The system would not allow further use of any frame name. Within a frame, the same approach could be used to eliminate redundant frame slot names. In order to prevent redundant connections, the system could allow at most one link hierarchically above and below the frame.

Another approach is to build a system that can detect potential redundancies. The existence of frame redundancies could be determined by comparing the content of all frames from the same hierarchical levels (e.g., as in the figures "group frames"), thus, limiting the number of frames that must be compared. Similarly, within a given frame, redundant slots and contents can be determined by comparing names or contents of slots. Finally, redundancy in links with other frames can be accomplished by examining uniqueness of links to and from frames.

Semantic Nets. Redundancy in semantic nets can occur in at least two ways, redundancy of arcs and redundancy of nodes. As a default, the designer of semantic net systems or shells could set a default of one arc with the same name per node. To the extent that those relationships can be specified, they can be used to assist in the verification process by finding those arcs that do not meet that conditions of the structure placed on them.

For example, in figure 4, depending on the context, it is unlikely that "chair" or any other object will occur in more than one node. Similarly, in many instances the user may wish to limit the use of a particular arc label from a given node to a single (or two or ...) occurrences. For example, a single "isa" may be allowed per node.

4.3 Completeness

A system may violate completeness if there is missing knowledge. The issue then becomes one of determining how to identify missing knowledge. In some situations, the nature of the knowledge representation provides a basis to assert incompleteness. For example, a rule of the form "if a then ----" is incomplete, since there is no conclusion.

Frames. In the case of frames structural incompleteness can occur if there is a missing frame, or a missing slot within the frame or if there is a missing link between frames.

In order to determine if there are missing frames or slots, the list of frame names, slot names and contents are elicited by the system as suggested above, can indicate a lack of completeness. First, if there is an unused frame name, slot name, or content, then that may indicate that the developer "forgot" or "neglected" to include that frame. Second, if there is a frame name, slot name or content that is used, but is not on the list of feasible names then that also can indicate a lack of completeness. Incomplete, since the use of a name not on a preestablished list is likely to indicate that the lists are incomplete to begin with.

A missing link between frames could be determined using a number of different approaches. If there is no path to a frame and the frame is not a "root" frame, then the knowledge base is not complete. For example, in figure 2 if there was no arc from the room frame to the wall frame then that would be an indication that the knowledge representation was incomplete.

In a similar way, statements can be made about other types of frames such as terminal frames and intermediate frames. If there is no path from the frame and the frame is not a "terminal" frame then the knowledge base is not complete. If the frame is an "intermediate" frame then there should be a path to another frame and from another frame.

Parallel structure also can be used to establish completeness. If the developer knows that there is parallel structure between sets of frames then a comparison between those sets of frames and connections will yield a determination as to completeness.

Semantic Networks. In the case of semantic networks structural incompleteness can occur if there is a node that is not connected by any arc and if there is an arc that leads to no node. For example, in figure 4, if the "isa" arc leads from chair and there is no "furniture" node, then the net is in error. In either case, those conditions indicate either incompleteness or incorrectness.

Further, similar to frames, "root" nodes, "intermediate" nodes and "terminal" nodes establish expectations. If a node is a root node then it should have no incoming arcs and it should only have outgoing arcs. For example, the root node in figure 2, "room frame," has only arcs to other nodes. If a node is an intermediate node then it should have both incoming and outgoing arcs. Similarly, if a node is a terminal node, it should only have incoming nodes and no outgoing nodes.

In addition, there is another approach to assessing completeness. If a list of eligible arcs and nodes is established then at least two cases can indicate a lack of completeness. First, if there is an unused arc or node name then that can indicate that the designer has forgotten to include the arc or node. Second, if there is a name that is used that is not on the list of feasible names, then that also can indicate a lack of completeness, similar to the case of frames, as discussed above.

4.4 Correct Structure

Much of the analysis of correctness of knowledge representation is domain based. However, assuming domain independence, it is still possible to assess the correctness of the structure of the knowledge representation, by drawing on certain graph theory concepts, such as whether the net is a tree or an acyclic graph.

Frames. Typically, relationships between frames employ either a tree structure or acyclic network structure, depending on how the systems fill their slots. Thus, in the first case, verification would consist of determining if the set of relationships between frames define a tree (easily ascertained using network theory). In the second case, verification would consist of determining if the network contains any cycles. This last approach is similar to those used with rule-based systems to elicit cycles, if they exist. Some frame-based packages allow cycles in the frame networks by only testing if the frame has one node hierarchically above and below. Such a test is only a myopic test of whether a graph is acyclic.

There also is a structure within each frame and many such frames employ the same structure. As a result, another important verification approach is to check the structure within frames that should be structured the same: do each have the same named slots, do each have the same number of slots, etc. This approach uses information that is solicited from the user in order to assist in the verification effort. This approach is part of the knowledge acquisition effort, since it can force the user to find symmetry and structure, in the knowledge, to the extent feasible.

Finally, the structure can be used to ascertain when a frame is inappropriately connected to another frame. First, it would be a very unusual situation to have a frame connected to itself. Second, an important verification characteristic of a frame is to collect the planned number of frames that frame is connected to, hierarchically from above and below. Thus, if more than that number of frames are connected in actuality, then there is an error. Again, this approach would try to ensure that the user examined the nature of the knowledge for symmetry.

Semantic Networks. As with frames, semantic networks are likely to employ either a tree approach or an acyclic network approach. Thus, similar verification procedures to frames would be used. Further, it would be unlikely that a node would be connected to itself by an arc. Any such occurrence likely would indicate that there is an error in the knowledge representation.

5. SUMMARY

This paper has argued that although there have been a number of efforts to establish verification processes for rule-based knowledge bases, there has been limited investigation of frame and semantic network knowledge bases. As a result, when building a frame or semantic network-based system or shell, the verification process is not clear from the literature. The thrust of this paper is to mitigate that gap. Thus, this paper provided a number of different approaches to verification of frames and semantic networks. Those approaches were designed to examine the knowledge base for consistency, redundancy, completeness and structure.

These include that network representations of frame-based knowledge bases be employed for a variety of tests, that frame-based systems force the user to established to the system information about parallelism and relationships between frames (e.g., number of frames with inputs to and outputs from, that lists of frame names and slots be used to mitigate errors in such knowledge bases and that frames be established as either root frames, terminal frames or intermediate frames. Similar approaches were developed for semantic networks.

Although this paper has focused on frames and semantic networks, this same approach can be extended to other forms of knowledge representation where the same effort is made to exploit the structure and content of the knowledge representation for verification purposes.

The verification process is critical to knowledge acquisition. At one level, the verification process is necessary to ensure that the knowledge that is represented is correct, complete and consistent. Otherwise when the system is compared to the user there could be errors inhibiting the performance of the system. In addition, this paper has argued that the verification process is an integral part of the knowledge acquisition process. Verification tests were identified that forced the user to examine the knowledge for structural characteristics and symmetry. Such analysis of existing knowledge could lead to additional knowledge.

REFERENCES

Adrion, W., Branstad, M., and Cherniavsky, J., (1982) "Validation, Verification, and Testing of Computer Software," ACM Computing Surveys, Vol. 14, No. 2, pp. 159-192.

Boose, J., (1989) "A Survey of Knowledge Acquisition Techniques and Tools," Knowledge Acquisition, Volume i, pp. 3-37.

Boose, J. and Bradshaw, J., (1987) "Expertise Transfer and Complex Problems: Using AQUINAS as a Knowledge Acquisition Workbench for Knowledge-based Systems," International Journal of Man-Machine Studies, Vol. 26, No. 1, pp. 3-28.

Davis, R., (1976) Use of Meta Knowledge in the Construction and Maintenance of Large Knowledge Bases, Ph.D. Dissertation, Stanford University,.

Gaines, B., "An Overview of Knowledge Acquisition and Transfer," International Journal of Man-Machine Studies, Vol. 26, No. 4, pp. 453-472.

Minsky, M., (1975) "A Framework for Representing Knowledge," in The Psychology of Computer Vision, P. Winston (ed.), McGraw-Hill, New York.

Nazareth, D., (1989) "Issues in the Verification of Knowledge in Rule-based Systems," International Journal of Man-Machine Studies, Vol. 30, pp. 255-271.

Nguyen, T., Perkins, W., Laffey, T., and Pecora, D., (1985) "Checking Expert System Knowledge Bases for Consistency and Completeness," Proceedings of the International Joint Conference For Artificial Intelligence, pp. 375-378.

Nguyen, T., Perkins, W., Laffey, T., and Pecora, D., (1987) "Knowledge-based Verification," AI Magazine, Vol. 8, No. 2.

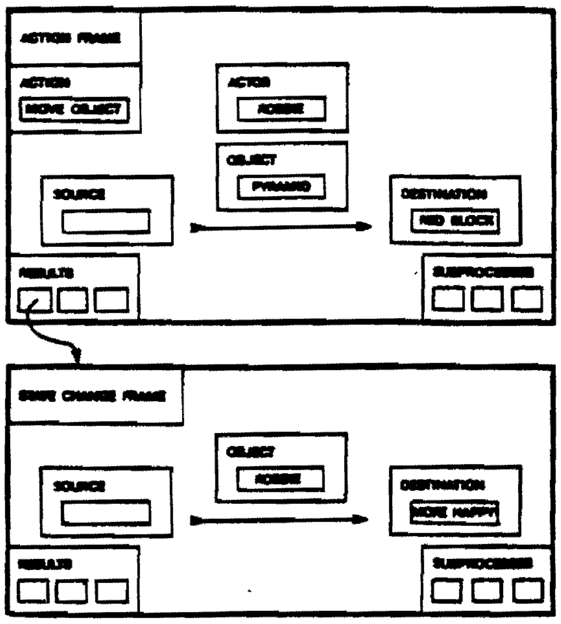
O'Leary, D., (1990) "Soliciting Weights or Probabilities from Experts for Rule-based Expert Systems," International Journal of Man-Machine Studies.

Rich, E., (1983) Artificial Intelligence, McGraw-Hill, New York.

Willingham, J. and Ribar, G., (1988) "Development of an Expert Audit System for Loan Loss Evaluation," Auditor Productivity in the Year 2000, Arthur Young.

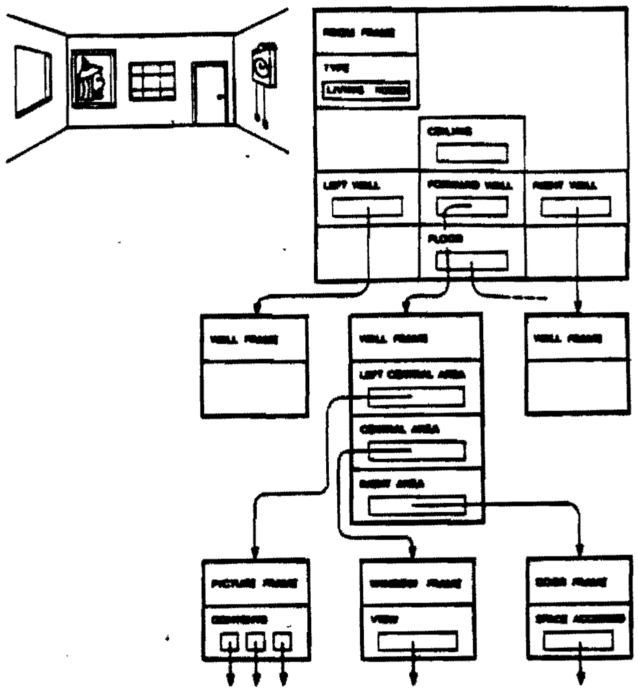
Winston, P., (1979) Artificial Intelligence, Addison-Wesley, Reading, Massachusetts.

Figure 1: Sample Frame



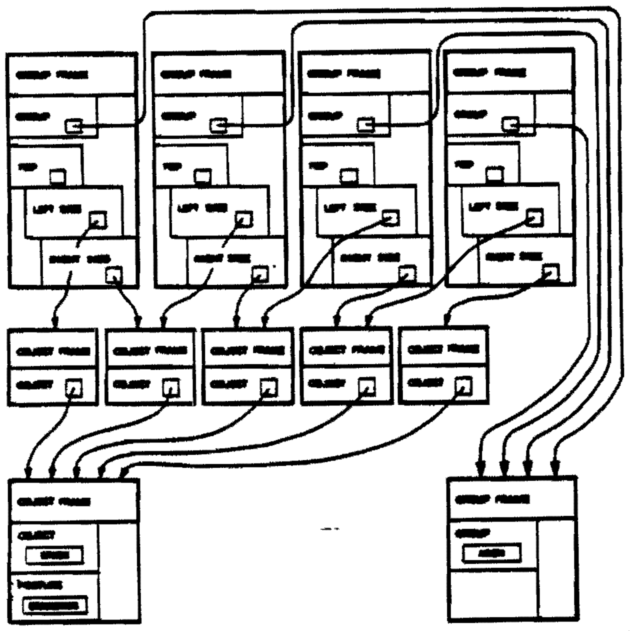
Source: Winston [1979]

Figure 2: Tree Structure of Frames



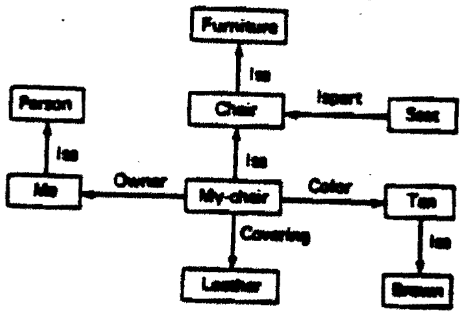
Source: Winston [1979]

Figure 3: Network Structure of
Frames



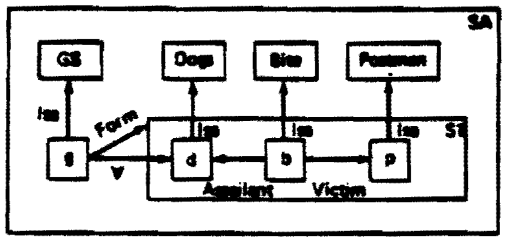
Source: Winston [1979]

Figure 4: Sample Semantic Net



Source: Rich [1983]

Figure 5: Parallel Structure
in a Semantic Net



Source: Rich [1983]

Proceedings of the 5TH BANFF KNOWLEDGE ACQUISITION FOR KNOWLEDGE-BASED SYSTEMS WORKSHOP

Banff Conference Centre, Banff, Alberta, Canada
November 4-9, 1990

In Cooperation with the
American Association for Artificial Intelligence

Workshop Co-Chairs:
John H. Boose, Boeing Advanced Technology Center
Brian R. Gaines, University of Calgary